Christian Payer, BSc

Separation of Arteries and Veins in Pulmonary CT Images

MASTER'S THESIS

to achieve the university degree of
Diplom-Ingenieur

Master's degree programme
Computer Science

submitted to

Graz University of Technology

Supervisor

Prof. Dr. Horst Bischof
Institute for Computer Graphics and Vision

Advisors

Dr. Martin Urschler
Ludwig Boltzmann Institute for Clinical Forensic Imaging

Dr. Zoltán Bálint
Ludwig Boltzmann Institute for Lung Vascular Research

Graz, Austria, April 2015

Abstract

Automated computer-aided analysis of lung vessels has shown to yield promising results for non-invasive diagnosis of lung diseases. In order to detect vascular changes which affect arteries and veins differently, an algorithm capable of identifying these two compartments is needed. We propose a fully automatic algorithm that separates arteries and veins in thoracic computed tomography (CT) images based on two integer programs. The first extracts multiple distinct subtrees inside a graph of vessel paths. The second labels each vessel tree as either artery or vein by maximizing both the contact surface in their generalized Voronoi diagram, and a measure based on their closeness to accompanying bronchi. We evaluate the performance of our automatic algorithm on 10 manual segmentations of arterial and venous trees from patients with and without pulmonary vascular disease, achieving an average voxel based overlap of 94.1% (range: 85.0% – 98.7%), outperforming a recent state-of-the-art interactive method. We show the possible clinical use of this artery-vein separation algorithm by quantifying the tortuosity of vessels, whereby the tortuosity of arteries can better distinguish between patients with and without pulmonary hypertension than the tortuosity of veins.

Kurzfassung

Automatische computergestützte Analyse von Lungengefäßen zeigt vielversprechende Ergebnisse bei der nicht invasiven Diagnose von Lungenkrankheiten. Um unterschiedliche Veränderungen in Arterien und Venen zu erkennen sind Algorithmen nötig, die diese voneinander unterscheiden können. Wir stellen einen vollautomatischen Algorithmus vor, der Arterien und Venen in Computertomographie (CT) Bildern des Thorax basierend auf zwei ganzzahligen Optimierungsfunktionen trennt. Die erste Optimierungsfunktion extrahiert mehrere unterschiedliche Teilbäume aus einem Graph von Gefäßpfaden. Die zweite Optimierungsfunktion markiert jeden Teilbaum entweder als Arterie oder als Vene, indem zwei Maße basierend auf den Grenzflächen ihrer generalisierten Voronoi Diagramme und auf ihrer Nähe zu nebenläufigen Bronchien verwendet werden. Die Leistungsfähigkeit unseres automatischen Algorithmus wird mit 10 manuellen Segmentierungen von Arterien und Venen gemessen, die für CT Bilder von Patienten mit und ohne Lungengefäßkrankheiten erstellt worden sind. Wir erreichen eine mittlere voxelbasierte Überschneidung von 94.1% (Bereich: 85.0% – 98.7%) und übertreffen damit eine interaktive Methode am aktuellen Stand der Technik. Wir zeigen eine mögliche klinische Anwendung von Arterien- und Venentrennung, indem wir die Gewundenheit von Blutgefäßen messen, wobei die Gewundenheit von Arterien besser dazu geeignet ist zwischen Patienten mit und ohne pulmonaler Hypertonie zu unterscheiden als die Gewundenheit von Venen.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

The text document uploaded to TUGRAZonline is identical to the presented master's thesis.

Place	Date	Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

Ort	Datum	Unterschrift

Acknowledgments

First and foremost I would like to thank Dr. Martin Urschler for his guidance throughout the creation of the thesis. Without his hints and ideas, the outcome of the thesis would not be possible. I would also like to thank all members of the Ludwig Boltzmann Institute for Lung Vascular Research, especially Dr. Zoltán Bálint and DI Michael Pienn, who gave me a lot of support and great suggestions and introduced me to the medical part of the thesis. Also, I would like to thank Prof. Andrea Olschewski for the opportunity to work at this institute among people of diverse professions. I also thank my fellow students and friends for the fruitful discussions and coffee breaks. Lastly, I thank my family for their financial and moral support during my whole studies.

Contents

List of Figures

List of Tables

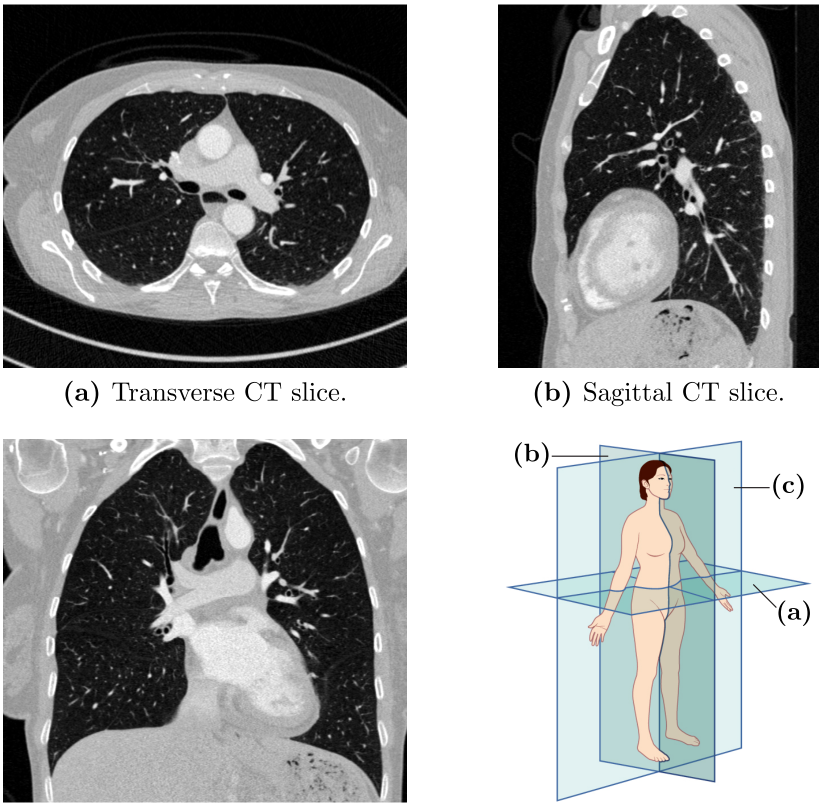
1 Introduction

Contents

1.1 Motivation

Medical imaging technologies like Computed Tomography (CT), Magnetic Resonance Imaging (MRI) and ultrasound are well established tools helping in the diagnosis of various diseases. Such imaging technologies allow non-invasive and three-dimensional (3D) insights into parts of the human body with great detail. Although *CT* examinations expose the patient to a considerable amount of radiation, the fast acquisition time of a few seconds and the uncomplicated procedure have made *CT* a standard tool in many diagnostic processes. Due to the varying X-ray attenuation of different materials, like air, water, bones, and other tissue, physicians are able to differentiate anatomical structures in *CT*-volumes, as can be seen in Figure 1.1. Furthermore, by using specific contrast agents, the visibility of various structures like blood vessels or nodules can be increased. Recent improvements in *CT*, like Dual-Energy Computed Tomography (DECT), dose reduction or spatial resolutions in sub-millimeter range, have further increased the value of *CT* as a diagnostic tool. This leads to a growing amount of data the physicians need to analyze. Therefore, many Computer Aided Diagnosis (CAD) systems are being created to assist experts in the diagnostic processes.

Methods of *CAD* cover preprocessing, pattern recognition, classification, segmentation and other image processing tools. Due to its excellent capabilities of visualization and quantification, segmentation is one of the most extensively used tools in *CAD*. Its task is to partition images or volumes into regions of interest, in order to locate specific objects or their boundaries. The objects of interest usually share certain characteristics and are



(a) Transverse CT slice.

(b) Sagittal CT slice.

(c) Coronal CT slice.

(d) Planes of the body.[1]

Figure 1.1: Example for different slices of a thoracic *CT* image. Image (a) shows a transverse, (b) a sagittal, and (c) a coronal slice of a thoracic *CT* image. Image (d) shows the orientation of the slices with respect to the human body.

generated by analyzing local or global image features and anatomical properties. As vessels are located throughout the whole human body, substantial research has been performed for their computer-aided segmentation. Especially the investigation of the pulmonary vessels, which are responsible for oxygen uptake and carbon dioxide release, is of great interest in clinical practice. Applications include detection of early stage pulmonary nodules [31], pulmonary embolism [27], or Pulmonary Hypertension (PH) [25].

PH is a life threatening chronic disorder of the pulmonary circulation. Although, *PH* is a rare disease with a prevalence of about 15 – 50 cases per million population, its prevalence for certain at-risk groups is substantially higher [35]. The main symptoms include dyspnoea on exertion, reduced exercise capacity, thoracic discomfort or pain. Despite the improvement of therapies in the recent years, the prognosis for *PH* is still very bad with one, three, and five year survival being 91%, 74%, and 65%, respectively [4]. *PH* is defined

[1] Planes of Body with modified labels, from OpenStax College, June 2013 via Anatomy and Physiology `http://cnx.org/content/col11496/latest/`, CC BY 4.0 License. © 2013 Rice University

with a Mean Pulmonary Artery Pressure (mPAP) ≥ 25 mmHg at rest and is grouped into various classes [16]. The gold standard for measuring the *mPAP* and therefore diagnosing *PH* is Right-Heart Catheterization (RHC). As *RHC* is an invasive technique, it is not suitable for regularly monitoring the progression of *PH*, whereas other imaging-based clinical tools could be used for this task.

Due to the increased pressure in the pulmonary vessels of patients with *PH*, the morphology of their lung vasculature is affected. Helmberger et al. [15] showed that analyzing the structure of pulmonary vessels in *CT* images by calculating various quantitative readouts can differentiate between patients with and without *PH*. Examples for quantitative readouts are the number of segments or branching points, the fractal dimension, or the tortuosity of vessels. As these readouts are highly dependent on the segmentations for which they are calculated, a robust vessel segmentation of the thoracic *CT* images is crucial for obtaining high diagnostic quality.

Due to the large amount of vessels in the lungs, automatic segmentations are preferable over manual ones. As the structure of the lung vasculature is complex and highly variable, automatic segmentation is difficult. Moreover, vessels in pulmonary *CT* images have often a similar attenuation range (measured in Hounsfield Units (HUs)) as other structures in close proximity, like bronchus walls, nodules, and tissue. Therefore, simple attenuation-based segmentation of the vessel trees does not yield satisfactory results and more sophisticated features need to be used [39]. As vessel structures are locally shaped like tubes, many methods for enhancing tubular structures have been presented in the past, e.g., [13], [41], [7], [26], [36], [22]. Those filters estimate the certainty for each voxel of the image of being the center of a vessel with specific radius and direction. With these enhanced vascular structures the segmentation or centerline extraction of vessels can be performed more robustly. Several methods have been presented that use the vessel-enhanced images as input for binarization [7], for weighted path calculation [15], for level-set evolution [26], for vessel traversal [45], and more. Excellent overviews of vessel extraction methods can be found in [19], [23], and [39].

With such robust vessel segmentation methods, quantitative readouts of the segmentations may be used for diagnosing diseases, like chronic obstructive pulmonary disease [10], [29], [28], or the aforementioned *PH* [15]. As some classes of *PH* manifest itself differently in arteries and veins [16], additional distinction between them and independent readout calculation could further increase the diagnostic value. Therefore, a separation of arterial and venous trees is of interest.

Although pulmonary arteries and veins are connected at their capillaries and thus form complex graph structures, the small capillaries are not visible in *CT* images. As a result, the vessel graphs are visible as multiple disconnected trees, as illustrated in Figure 1.2. These arterial and venous trees are intertwined and often run in parallel or in close proximity to each other. Also, the borders between adjacent vessels are often not visible, due to the partial volume effect. Therefore, crossings and proximities look similar to vessel branches and their detection may be mixed up when just analyzed locally, as

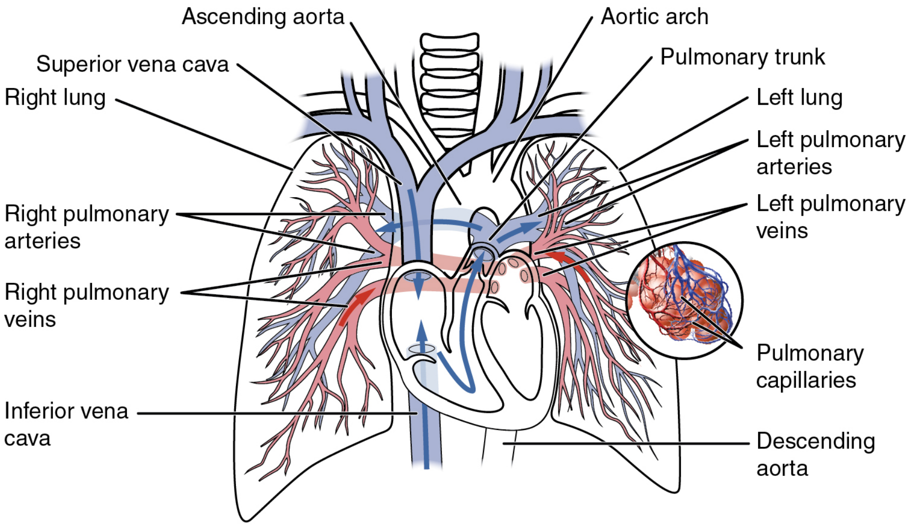
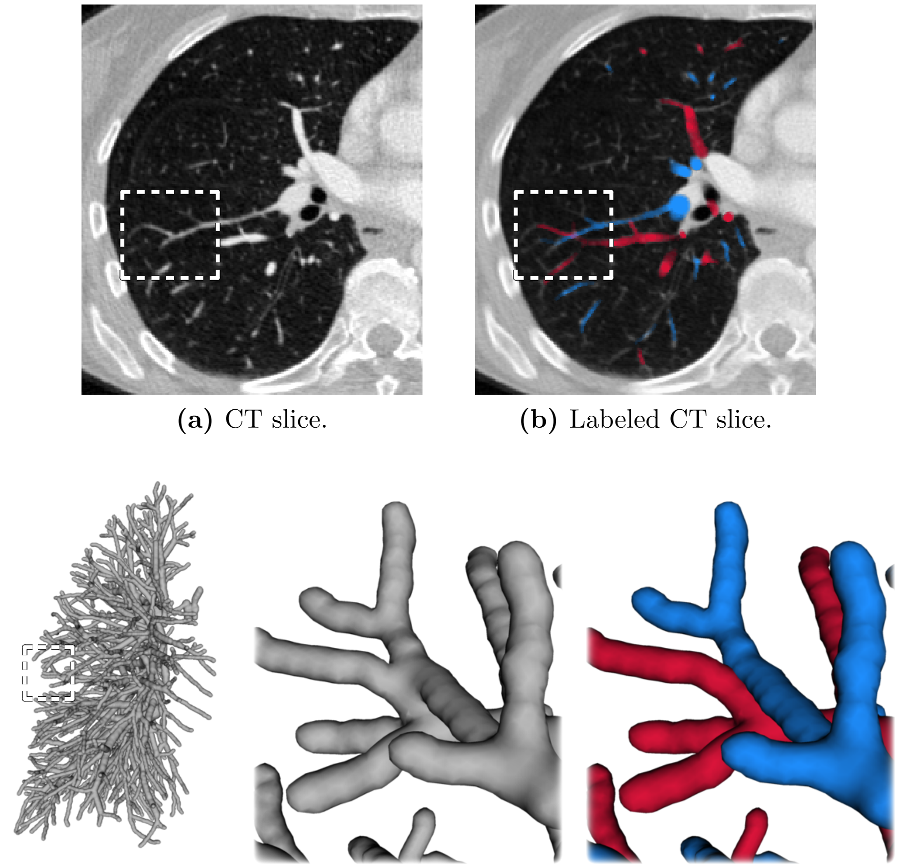


Figure 1.2: Visualization of pulmonary circulation with arterial and venous trees.[2]

can be seen in Figure 1.3. Such errors do not change the vessel segmentation itself, but the extracted structure of the tree. Thus in vessel segmentation such errors do not carry much weight, in contrast to artery-vein (A/V) separation, where the structure of the trees and correct connections are far more important. Furthermore, arteries and veins are not differentiable by other local features, like their attenuation in *CT* images. Even with contrast-enhanced *CT*, which could lead to different attenuation for arteries and veins, *HU*-based separation is not robust enough, as the propagation of the contrast material inside the lung is too unpredictable [20].

As arteries and veins are not differentiable when just analyzed locally, more global anatomical properties have to be taken into account for a robust separation. Such anatomical features may be the vessel's connection to the heart, i.e., arteries start at the main pulmonary artery, whereas veins start at the left atrium. Another feature of pulmonary vessels is that pulmonary arteries usually run in parallel and in close proximity to bronchi, while veins are farther away. Furthermore, pulmonary arteries and veins start both in the heart, while being anatomically connected to each other in the capillaries, which spread over the lungs. Also, the amount of blood that flows from the heart into the pulmonary arteries is the same as the amount of blood that flows from the veins back into the heart. This leads to a uniform distribution of arteries and veins throughout the whole lungs, i.e., there are no large regions with just arteries and no veins, or conversely. It has to be noted, that such anatomic features vary from person to person, which makes the task of automatic A/V separation even harder.

[2] Pulmonary Circulation, from OpenStax College, June 2013 via Anatomy and Physiology `http://cnx.org/content/col11496/latest/`, CC BY 4.0 License. © 2013 Rice University.



(a) CT slice.

(b) Labeled CT slice.

(c) Rendering of a vessel segmentation.

(d) Zoomed in vessel segmentation.

(e) Zoomed in A/V segmentation.

Figure 1.3: Example of a thoracic *CT* image, where arteries (blue) and veins (red) are close to each other. The vessel branches in the white box of the *CT* slice (a) appear as being directly connected, whereas the A/V separation in (b) shows that the branches belong to two distinct vessels. The 3D rendering of the vessel segmentation (c) and its zoom-in (d) visualize these merged branches in 3D, whereas the A/V separation (e) can tell them apart.

1.2 Contribution and Structure of the Thesis

Some approaches of A/V separation in thoracic *CT* images have been presented in the past (see Chapter 2), whereas most of them need a certain amount of user interaction. We propose a novel fully automatic method that generates a segmentation of vessel trees inside the lungs and labels them afterwards as either arteries or veins. Different to other algorithms, we do not create a full vascular tree segmentation and separate it afterwards, but create separated vascular subtrees in the first place. This idea is inspired by the works of Türetken et al. [46] and Robben et al. [37], who propose to use integer programming for segmentation and labeling in order to extract connected vessels. We contribute to this integer programming optimization by extending it with an automatic vessel root detection for an arbitrary number of vessel subtrees. As a result of the subtree extraction, we are able to label the subtrees without any further need of modifying them. The final A/V labeling is based on anatomical properties of vessels in the lungs. We use for this task a measure of the vessel's proximity to bronchi, which is based on the work of Bülow et al. [6], and a novel measure that quantifies the uniform distribution of vessels in the lung. Our contribution to the labeling lies in the combination of these two anatomical measures and their integer-programming-based optimization.

The proposed algorithm starts with generating a lung segmentation, which is based on [15]. All subsequent steps are performed for the left and right lung independently. Afterwards, vascular structures are enhanced with a multi-scale tubularity filter [22] for the later vessel extraction. In order to generate vessel paths, regularly spaced local maxima are extracted from this vessel-enhanced images and connected with four-dimensional tubular paths [3]. A subsequent integer program groups these tubular paths in order to represent distinct vascular subtrees, which is based on [46]. A main contribution of our work is that we do not need declared root nodes for this subtree calculation, but generate them implicitly by minimizing a constrained objective function. As each of these distinct subtrees is either an artery or a vein, they have to be labeled accordingly.

Another contribution is the novel integer program based A/V labeling step that uses two anatomical properties of pulmonary vessels to perform this labeling task. The first anatomical property is the uniform distribution of arteries and veins throughout the lungs, where our proposed A/V separation method is the first one that makes use of it. We quantify this uniform distribution by maximizing the contact surface of neighboring arterial and venous regions of a Generalized Voronoi Diagram (GVD). We use the proximity of pulmonary arteries and bronchi as a second anatomical property and measure it with an "arterialness" value that is based on the method described in [6]. With these two anatomical properties, the A/V labeling of the extracted subtrees is performed, which results in the final A/V separation. A more detailed description of the algorithm is given in Chapter 3.

We evaluate the proposed method by comparing it with a recent state-of-the-art interactive algorithm and calculating voxel-based agreement with 10 manually labeled datasets. This evaluation shows good performance of the proposed fully automatic algorithm with a mean agreement of 94.1% outperforming the state-of-the-art interactive algorithm that achieves a mean agreement of 91.9%. The segmentation results are shown in Chapter 4.

To show medical applicability of our algorithm, we calculate quantitative readouts of the generated arterial and venous trees. We use previously described readouts of [5] and [15] and correlate them to clinical parameters of the patients. These results are presented and discussed in Chapter 5.

Finally, we summarize and give a conclusion of the thesis in Chapter 6.

2 Related Work

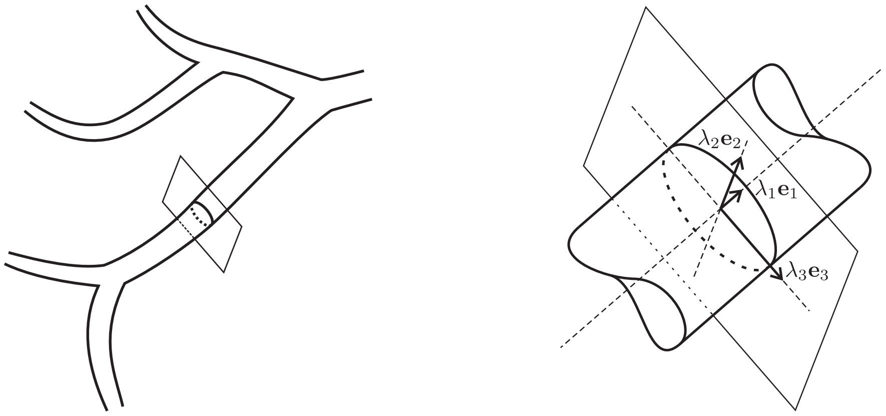
Contents

In this chapter related work on artery-vein (A/V) separation is discussed. As most A/V separation methods are dependent on a good vessel segmentation, a short overview of vessel segmentation methods is given as well (see Section 2.1). Methods for A/V separation are described in detail in Section 2.2, whereas the motivation for our method is given in Section 2.3.

2.1 Vessel Segmentation

Vessel segmentation has gained much interest in the past. Excellent overviews can be found in Kirbas and Quek [19], Lesage et al. [23], and Rudyanto et al. [39]. In numerous medical imaging technologies vessels are observed, which may be quantified or segmented for Computer Aided Diagnosis (CAD). One of the main physical property of vessels is, that they have locally a tube-like shape, except for branching points, as visualized in Figure 2.1. Therefore, numerous filters have been developed that enhance tube-like structures as a pre-processing step for vessel segmentation ([13], [41], [7], [26], [36], [22]). The methods of Frangi et al. [13], Sato et al. [41], and Pock et al. [36] use an eigenvalue and eigenvector analysis of the Hessian matrix to estimate the direction of the local tubes. In local tube-like structures, the eigenvalues of the Hessian have the property that one eigenvalue is close to zero while the other two are much larger with approximately equal value (see Figure 2.1b).

Another method that enhances tubular structures is from Law and Chung [22]. They enhance image regions by projecting the image gradient along some axes and measure



(a) Schematic vessel.

(b) Local tube with eigenvectors of Hessian.

Figure 2.1: Schematic representation of a vessel branch and its local tube-like structure (a). The right image (b) shows the normal plane of the vessel and the eigenvectors of the Hessian. The images are adapted from [2].

the amount of the projected gradient that flows in or out of a local spherical region. Afterwards an eigenvalue and eigenvector analysis of this oriented flux is performed to get the directions of the tube.

Cañero and Radeva [7] and Manniesing et al. [26] use the results of such a vessel enhancement filter as input for denoising. Their method of vessel enhancing diffusion smooths the image in the direction of a vessel, while it blurs regions outside of vessels. This way, the edges at the vessel borders are preserved, while other regions get blurred.

All vessel enhancement filters have in common that they highly depend on the size of their internal filter or structuring element. As a consequence, only vessels within a small radius range can be enhanced in one step. To overcome this limitation, multi-scale analysis is performed to obtain a broader range of vessels. This leads to multiple result images, which can be combined by taking a normalized maximum value over all scales, as seen in Figure 2.2.

Such so-called ”tubularity” or vessel-enhanced images are used as an input for creating a vessel segmentation or for tracking vessel trees. Various methods like binarization [7], weighted path calculation [15], level-set evolution [26], vessel traversal [45] are used to perform these tasks. For a more detailed list see [19], [23], and [39].

Most methods either create a vessel segmentation and afterwards calculate their centerlines, or the other way around. An approach that combines these steps is from Li and Yezzi [24]. They represent vessels as four-dimensional (4D) paths, where the first three dimensions are the spatial dimensions of the centerline and the fourth represents the radius of a specific point on the path. By calculating a 4D path from one point to another, the segmentation can be obtained by taking the envelope of spheres with the corresponding

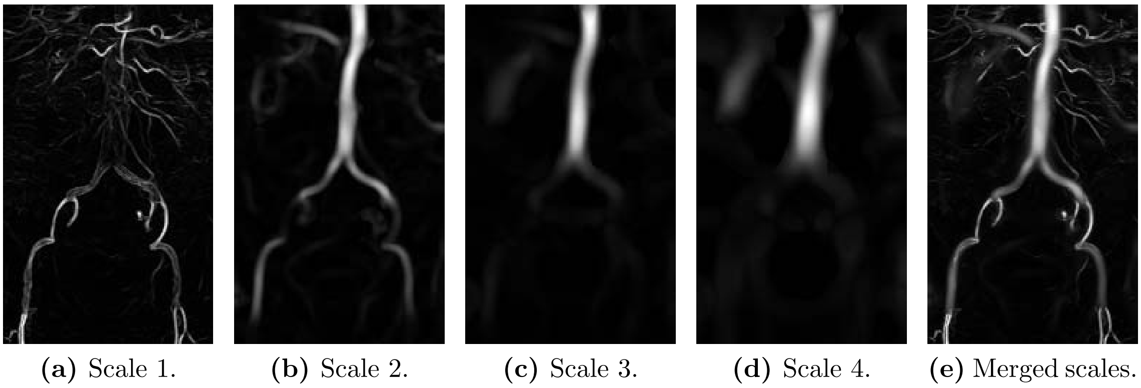


Figure 2.2: Visualization of a multi-scale vessel enhancing approach. The first four images ((a), (b), (c), (d)) show vessel-enhanced images for various scales, while the last image (e) shows the final multi-scale image based on a selection procedure. The images are taken from [13].

radii on all points of the path. As the method presented by Li and Yezzi [24] just calculates vessel paths between two points, Türetken et al. [46] extended the method to a full path extraction scheme that generates whole graphs of vessels. Some parts of their methods are used by our proposed algorithm and will be presented in more detail throughout Chapter 3.

2.2 Artery/Vein Separation

Although not being as extensively discussed as vessel segmentation, A/V separation still has found some interest in various medical applications. Substantial research has been done using two-dimensional (2D) images, especially in color images of the retina, e.g., Lau et al. [21], Hu et al. [17], and Joshi et al. [18]. Although a few elements of A/V separation in 2D images are similar to 3D images, e.g., analysis of the structural connections of vessel segments, there are major differences. In 2D images, branching points are not distinguishable from vessel crossings, when just observed in their local neighborhood. Furthermore, parts of different vessels can be on top of each other, which leads to just one visible vessel segment that belongs to more than one vessel, as visualized in Figure 2.3. Also, other features like the color can help to distinguish arteries and veins when using 2D retina images. Such information is not available in 3D Computed Tomography (CT) images. As we want to separate arteries and veins in *CT* images, we focus on 3D methods of A/V separation in this section.

One application of 3D A/V separation is hepatic *CT*, e.g., Foley et al. [11], Selle et al. [42], and Francis et al. [12]. Due to the relatively slow blood flow in the liver, injecting contrast agents and scanning multiple times in a row support separation of the different vessel types, as shown in Figure 2.4. Therefore, the different vessel types can

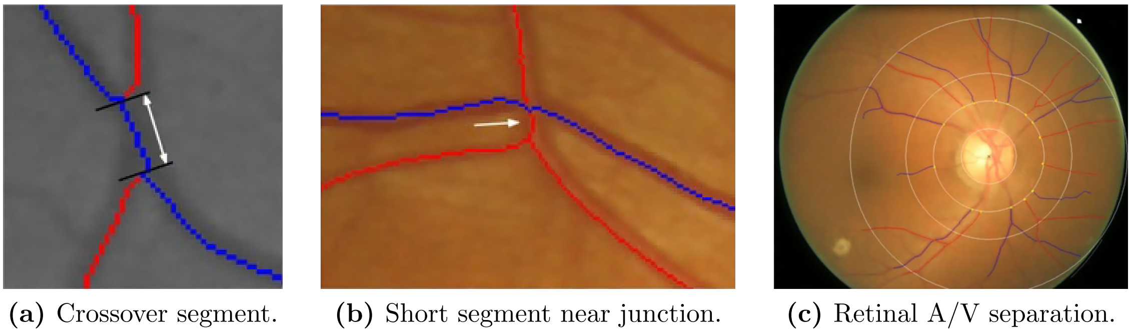

(a) Crossover segment. (b) Short segment near junction. (c) Retinal A/V separation.

Figure 2.3: Examples of A/V separation in retinal images. Overlapping vessel segments (a), as well as short segments near junctions (b) cause problems in A/V separation. The last image (c) shows a result of A/V separation in retinal images. The images are taken from [21].

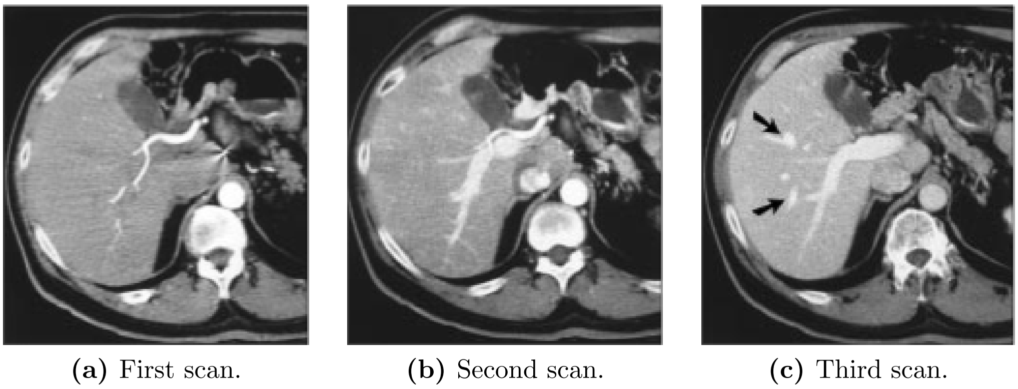

(a) First scan. (b) Second scan. (c) Third scan.

Figure 2.4: Examples of multiple scans for hepatic *CT*. As the blood flow in the liver vessels is relatively slow, multiple *CT* scans can be used in order to segment the vasculature. The images (a), (b), and (c) show the contrast medium at different points in time and are taken from [11].

be segmented in the specific scan, in which they are visible the best. The information of all images together can be used for creating the full vasculature of the liver. Moreover, the vessels trees in the liver are not as intertwined as in the lungs, which enables more straightforward segmentation methods [42].

In thoracic *CT*, arteries and veins have similar attenuation properties and even when using contrast agents, the attenuation difference of arteries and veins cannot be determined reliably. The timing of the scan when the contrast material is either completely in the arteries or completely in the veins is very hard to estimate and differs from patient to patient. Additionally, scanning multiple times in a row is not practicable due to the high radiation dose of a thoracic scan. Therefore, more complex features than attenuation difference have to be taken into account for A/V separation.

Most of A/V separation methods in pulmonary *CT* images use structural information, either for tracking vessels from given seed points, or for a voxel-wise distinction of arteries and veins. Some methods also incorporate other anatomical features, like proximity of arteries and bronchi to separate arteries and veins. In the following we analyze and give an overview of various A/V separation methods grouped accordingly:

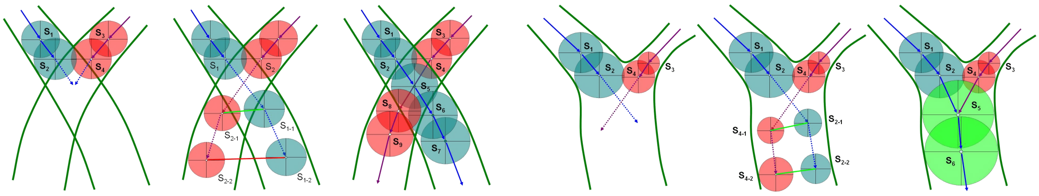
- methods using generative models for vessel tracking
- methods using voxel-based labeling
- methods using additional anatomical features

2.2.1 Methods Using Generative Models for Vessel Tracking

The methods described in this section use vessel tracking techniques by generating models of geometric primitives and fitting them to vessel segmentations.

The first method presented in detail is from Park et al. [33]. They create an A/V separation by connecting blown up spheres inside the vessel segmentation. The first step of their algorithm is to make a multilevel segmentation of the thorax to extract lungs and blood vessels. As bones outside the lung have similar attenuation compared to vessels and therefore could influence the vessel segmentation, they are removed beforehand by using smooth surfaces. Those smooth surfaces stick to the outer border of the lung segmentation and remove the ribcage. The next step is to extract the graph representation of the pulmonary trunk. To get a better separation of the pulmonary trunk close to the heart regions, morphological operations, like dilation, are performed. Afterwards, Park et al. calculate a distance transformation inside those binary regions. On each local maximum of the distance transform, spheres with radius of the corresponding distance values are created to represent the vascular structures. As the spheres generated by local maxima are not very dense and may not be overlapping, new spheres have to be generated until all spheres are connected. Two rules are applied to generate these new spheres. First, if two neighboring spheres are overlapping, a new sphere that maximizes the distance as well as the radius between them will be generated. Second, if a sphere is isolated, a new sphere with the maximum radius within a defined distance to the isolated sphere will be generated. By analyzing the graph representation of the connected spheres, the trunk of the pulmonary artery can be found.

Afterwards, the pulmonary vessel graph is extracted by generating spheres in the same way as explained before with the following additions: On dead ends, new spheres are generated in the direction of the dead end until the end of the vessel segmentation is reached. Branches are detected by analyzing spheres generated from different starting points, as visualized in Figure 2.5. If the direct connection for at least one pair of spheres generated from different start points is outside the vessel segmentation, the branches are just touching each other and will not be merged (see Figure 2.5a). If the direct connections of the spheres lie completely inside the vessel segmentation, they are merged to one branch



(a) Tracking step.

(b) Branching step.

Figure 2.5: The images show the spheres generated for vessels next to each other (a) and for vessel branches (b) for the method of Park et al. [33]. The images are taken from [33].

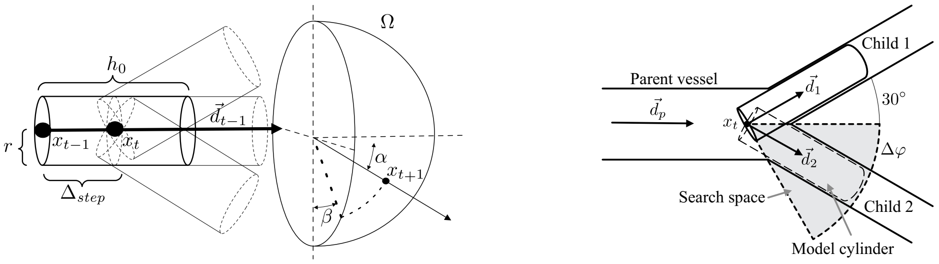
(see Figure 2.5b). When all possible spheres inside the vessel segmentation are generated, vessels are labeled as arteries, if they are connected to the pulmonary artery, or as veins, if they are not.

As Park et al. [33] did not provide a comparable quality measures of their method, further analysis is needed in order to evaluate its practical benefit.

Another method that uses a similar approach as Park et al. [33] is from Wala et al. [48], whereas they fit cylinders instead of spheres into the vessel segmentation. Although they just segment the first vessel branch starting in the main pulmonary artery, their method shows some promising ideas for vessel tracking.

The algorithm uses a threshold based vessel segmentation as input for a vessel tracker. Due to the similar intensity values of airway walls and vessels, they could affect the vessel tracker. Therefore, they are removed beforehand by subtracting a slightly dilated airway segmentation from the *CT* image. The next step of the algorithm is to automatically find the seed point, i.e., the pulmonary artery, for the vessel tracker, which is done by constructing and analyzing regions of interest around airway branches. These regions of interest are defined by cylinders around the airway. They are thresholded at -400 Hounsfield Unit (HU) and morphologically opened to remove unwanted structures. Afterwards, a connected component analysis is performed to get possible candidates for the pulmonary artery. Every object that is too close to the airway center is removed, as it may contain areas of the heart. The remaining object, that is closest to the airway center, is defined as the pulmonary artery. Its centroid is set as the seed point for the vessel tracker.

Starting from this seed point, the vessel tracker fits cylinders into the vessel segmentation, as shown in Figure 2.6. Various cylinders in a defined direction and radius range are generated at the end point of the previous one (see Figure 2.6a), while the cylinder with the best overlap with the vessel segmentation is taken as the next one. The vessel tracker appends new cylinders until it either fails to find a strong match (= cylinder with at least 50% overlap) or a bifurcation is detected. Bifurcations are detected by generating two cylinders that satisfy a bifurcation model based on the direction between themselves and the direction of the parent cylinder (see Figure 2.6b). If a vessel segment is too short



(a) Tracking step. **(b)** Branching step.

Figure 2.6: Cylinders created for vessel tracking (a) and branching detection (b) from the method of Wala et al. [48]. The images are taken from [48].

or leaking into the nearby mediastinum, the vessel tracker stops and the leaked cylinders are removed. Finally, all created cylinders are combined to obtain the full segmentation of the tracked vessel.

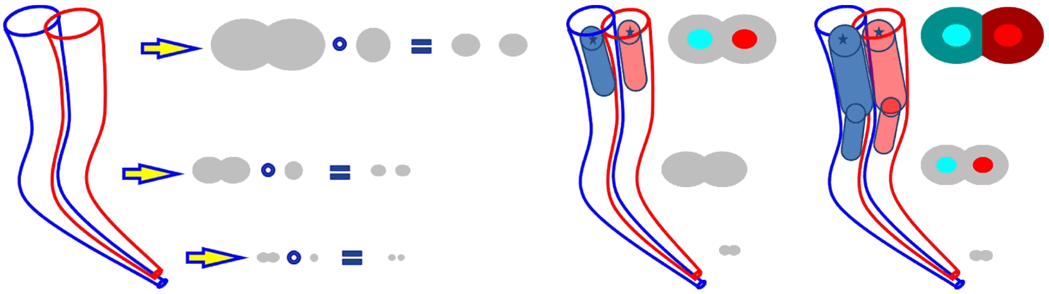
The vessel tracker of Wala et al. [48] shows promising results for the segmented pulmonary arteries, whereas a major drawback is that they only generate one branch of the pulmonary artery and no veins at all. Nevertheless, it could be used to track vessels starting from manual seed points in order to generate more complete vessel trees.

2.2.2 Methods Using Voxel-Based Labeling

The methods described in this section label arteries and veins voxel-wise by incorporating local or global connections.

Saha et al. [40] create a voxel-wise A/V separation by using manual seed points and the connections between them. They propose a method that separates two objects with almost equal intensity values by morphological opening with differently sized kernels, as shown in Figure 2.7. As this is the case for pulmonary arteries and veins in *CT* images, the task is now to find the size of the local kernel depending on the size of the vessel and to trace continuity of locally separated regions.

The A/V separation starts with a fuzzy vessel segmentation presented in Shikata et al. [44] and two sets of manual seed points (one set for arteries and one for veins). The segmentation is called fuzzy, as some voxels of the actual vessels may be missing and some voxels outside the actual vessels may be present. Then they calculate a fuzzy distance transform that can cope with disconnected voxels. The fuzzy distance from one point to another is the length of the shortest paths between them on the fuzzy vessel segmentation. Afterwards, the local scale, which is the local maximum of the fuzzy distance transform value, is calculated for every point. To reduce the effects of spatial variations in the objects scale, the fuzzy distance transform values are normalized by dividing them by their local scale. When using the normalized value, the local minima stay the same, while



(a) Different sizes of morphological operators along vessels, that are needed to separate arteries and veins.

(b) Paths leaking into each others classes.

(c) Stop leaking with morhophological reconstruction.

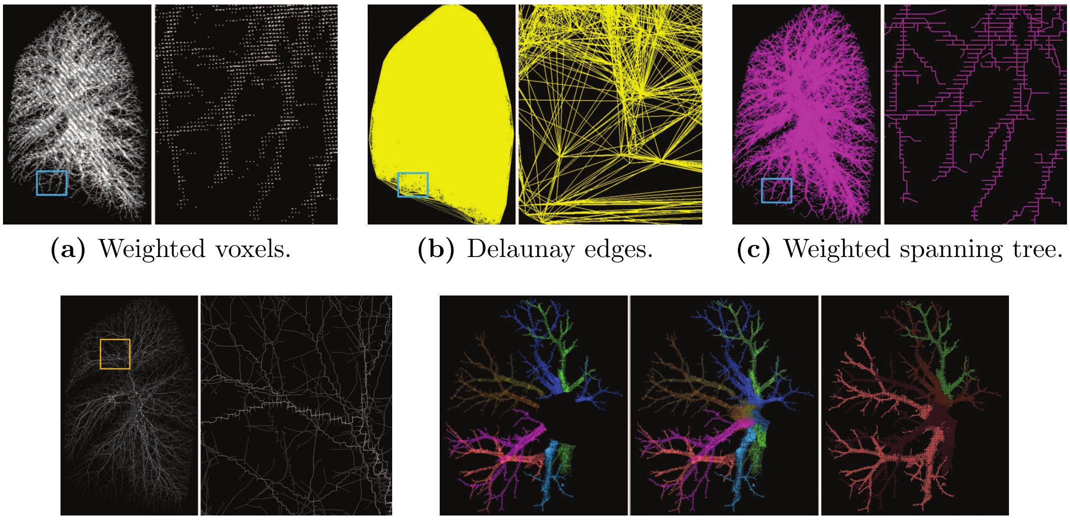
Figure 2.7: Schematic description of A/V separation proposed by Saha et al. [40]. The left image (a) shows the need of different sizes of morphological operators in order to separate arteries and veins. The next images show paths leaking to the wrong classes (b) and the solution by morphological reconstruction (c). The images are taken from Saha et al. [40].

the number of needed iterations is reduced. Another value needed from the algorithm is the fuzzy morphoconnectivity between two points. The fuzzy morphoconnectivity is the size of the erosion operator that disconnects the two points.

With the fuzzy distance transform and the fuzzy morphoconnectivity, the regions of the two isointensity objects can be calculated. An object region of a class is defined by all points that have a greater morphoconnectivity value to the seed points of its class to the seed points of the other class. As the regions defined for each class do not fill the whole vessel segmentation, the space around them must be filled without compromising their separation (see Figure 2.7b). This is done by adding unclassified points of the vessel segmentation to the class, whose nearest point is closer than the nearest point from the other class. At this stage of the algorithm, one iteration is finished and the fuzzy distance transform values are updated so that they do not contain paths from one class to the other. The process of calculating morphoconnectivity, separating the classes and connecting points to the classes is repeated until no change in the classes is detected anymore.

With enough seed points for each class, the algorithm can achieve a separation accuracy of 100%. As the algorithm usually performs better with more seed points, an efficient user interface is implemented by Gao et al. [14]. A major advantage of this algorithm is that it can correct erroneous vessel segmentations with its fuzzy approach, while the necessary user intervention can be considered as a drawback.

Another method that also tries to separate arteries and veins for every voxel of a vessel segmentation is from Park et al. [34]. The goal of their method is to find voxel-based trees with minimal construction energy inside the vessel segmentation. The steps of the algorithm are shown in Figure 2.8.



(a) Weighted voxels.

(b) Delaunay edges.

(c) Weighted spanning tree.

(d) Tree depth edges.

(e) Cut, regrown and merged subtrees.

Figure 2.8: The images show steps of the algorithm of Park et al. [34]. At first, voxels of a vessel segmentation are weighted (a) and connected with Delaunay edges (b). Afterwards, a weighted spanning tree (c) on these edges is cut by their depth (d) to create various subtrees that are merged again (e). The images are taken from [34].

The vessel segmentation used as input is created by region growing with adaptive thresholding followed by a filtering based on the eigenvalues of the Hessian. The algorithm generates weights for every voxel which are based on the normalized intensity of the *CT* image, the normalized distance to the boundary of the vessel, and the Laplacian of this distance (see Figure 2.8a). With these vessel weights, a weighted Delaunay triangulation is created to generate edges between the voxels (see Figure 2.8b). The Delaunay edges and the voxels of the vessel segmentation define a graph from which a tree with minimal construction energy is extracted. This shortest path tree is calculated with Dijkstra's algorithm [9] starting from the most central voxel of the *CT* image, where the edge costs are calculated as follows: For every voxel a local direction has to be estimated. Instead of using the Hessian-based vessel orientation, a direction that is based on the covariance matrix of position differences between the voxel and its neighbors is used. The weight is calculated by taking the dot product of the calculated direction of the voxel and the direction of the edge itself.

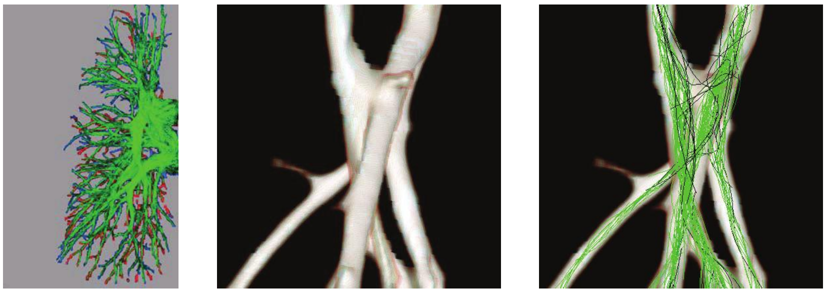
This leads to a single tree that contains arteries as well as veins, as visualized in Figure 2.8c. Next, the tree is cut at several positions in order to create distinct subtrees. This is done by cutting edges whose children have a higher depth than a user-defined constant value (see Figure 2.8d). As the cutting depth is hard to estimate, Park et al. suggest to set this cutting depth manually by an interacting user in order to improve the results. Now, the branches of arteries and veins should be separated quite well and can be merged again, until only two classes remain. The automatic merging of these subtrees is performed by generating cylinders on the root node of each branch and calculating their mutual overlap. If the overlap is high enough, the branches are merged into the same class, as shown in Figure 2.8e. As this step is very critical for the final outcome of the algorithm, the merging criterion is set to be very conservative in order to merge only branches that belong together with high certainty. The remaining branches must be labeled by hand as the correct class in order to generate the final A/V segmentation.

The connecting graph approach and subtree calculation of Park et al. [34] look promising, although there is a significant amount of user interaction needed. Another drawback is that, even though most of the regions where vessels are touching each other are in the proximal region of the lung, they also exist in the more distal regions, which is not handled at all by the algorithm. For evaluation purposes and comparison with our proposed method we reimplemented the method of Park et al. [34], as described in Chapter 4.

Another promising method is from Kitamura et al. [20], who formulate the vessel segmentation as well as the vessel separation as a voxel labeling problem. Their algorithm consists of three steps: automatic root detection for the arterial and venous trees, vessel region extraction, and A/V separation.

The main pulmonary artery and the left atrium are used as roots of the vessel trees. They are segmented using a landmark detector that is trained for the main pulmonary artery and the left atrium, respectively. The next step is to generate a vessel segmentation that is based on a graph-cut method with unary and pairwise terms. Foreground seeds are set automatically around voxels, where the roots have been detected as well as voxels with a high tubularity value of a multi-scale vessel detector. Bronchus voxels, as well as voxels, that have a lower intensity value than the root nodes are set automatically as background seeds. The pairwise terms of the graph-cut method smooth the labeling depending on the gradient values of the image. This vessel segmentation is then used as an input for A/V separation. Arteries and veins are separated by minimizing an energy function that is composed of unary, pairwise and higher-order terms. The unary terms are set around the root positions to enforce their correct labeling, whereas the pairwise terms enforce smooth labeling depending on the image gradients. The higher order terms are more complicated and explained in the following.

For every voxel in the vessel segmentation a shortest paths tree, which covers paths with length of 15 voxels, is constructed, as shown in Figure 2.9. From this shortest path tree, the path with the minimal cost is extracted, where the edge costs are based on image intensities. Weights of these minimum cost paths are calculated in order to use them as the higher order potentials of the energy function. These weights are composed of several features, e.g., the length of the path, the straight-line distance between the endpoints, the total curvature along the path, the maximum curvature, the maximum intensity derivative, and the variance of the intensity derivative. Due to the complexity of finding correct



(a) Vessel segmentation.

(b) Zoomed in vessel segmentation.

(c) Weighted local shortest paths.

Figure 2.9: The images show the weighted local shortest paths of a vessel segmentation (a). The paths of the zoom-in (b) are shown in (c), where better paths are drawn in brighter green. The images are taken from Kitamura et al. [20].

weightings for all these features, they are generated with a machine learning algorithm, where manually prepared reference segmentations are used as training data. With the unary, pairwise and higher-order potentials, the energy function is minimized resulting in the final A/V separation.

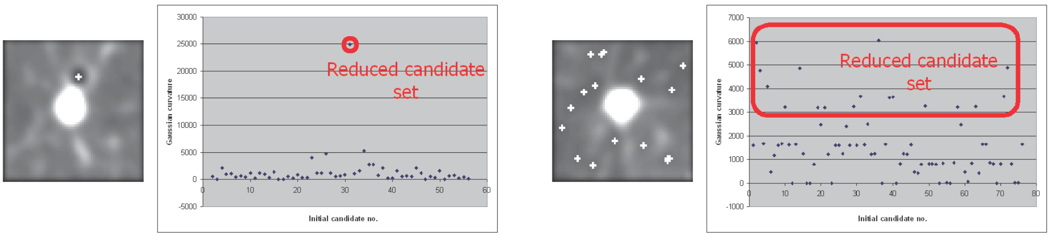
The approach of minimizing the energy functional provides promising results, as described by Kitamura et al. [20]. An advantage of their method is that it is fully automatic, although the manual reference segmentations that are needed for training could be seen as a drawback. The main cause of error of the algorithm is around the hilum of the lung, where arteries and veins are very close to each other. Unfortunately, errors in these regions propagate to more proximal regions of the vessel trees.

2.2.3 Methods Using Additional Anatomical Features

The methods described in this section use additional anatomical features for A/V separation. Although the vasculature inside the lungs is very variable, there are some usually fixed anatomical properties. One property is that arteries run along bronchi, while veins run in between their branches.

Bülow et al. [6] use this fact to perform an automatic A/V separation by using their proposed measure of ”arterialness”. The separation algorithm needs a full vessel segmentation with the corresponding tree structure as an input. The vessel segmentation is based on a fast marching approach with initial seed points. Various voxel-, segment- and tree-based criteria are evaluated to generate the whole tree structure and segmentation result.

Starting with this vessel segmentation, the individual branches are evaluated as being arteries or veins by assuming that arteries are close and in parallel to bronchi. As automatic airway segmentations are usually not deep enough to contain the bronchi of interest, Bülow et al. [6] suggest another method of checking the proximity of bronchi to vessels. For every centerline point on every vessel in a certain radius range, they search for bronchus points neighboring the vessel. The restriction to a certain range of radii comes from the observation that bronchi accompanying vessels with small radii are not visible on the *CT* images, while the assumption that only arteries run along bronchi does not hold for vessels with too big radii.



(a) Bronchus point candidates for artery.

(b) Bronchus point candidates for vein.

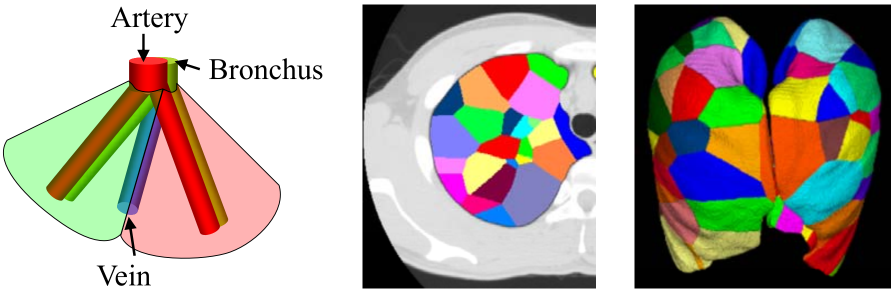
Figure 2.10: Example images of planes perpendicular to an artery (a) and to a vein (b) with the reduced bronchus candidate set of the method of Bülow et al. [6]. The images are taken from [6].

For every remaining vessel centerline point a local plane, orthogonal to the vessel direction, is extracted. The next step is to find possible bronchus candidates inside this plane. The bronchus should be visible as a dark blob with a nearly Gaussian profile. Therefore, all local gray value minima are extracted and the Gaussian curvature of those points is calculated, as visualized in Figure 2.10. Afterwards the points are filtered with their Gaussian curvature by removing points below the half of the maximum calculated Gaussian curvature. As a result of a too low maximum value of the Gaussian curvature, too many points are remaining (see Figure 2.10b) and the vessel point is classified as a vein. If the maximum is high enough and just a few points are remaining, the coorientation ($O = |\mathbf{e}_{bronchus} \cdot \mathbf{e}_{vessel}|^2$) of the bronchus point direction and the vessel direction is calculated. The maximum coorientation of all remaining bronchus candidates is taken as the final measure of arterialness of this vessel point. To reduce errors near branching points, the arterialness value is calculated only for the first half of each vessel segment.

The final labeling of arteries and veins is done by setting vessel segments whose mean arterialness value is greater than a certain threshold to arteries. All unlabeled subtrees of labeled vessel segments receive the same label as their parent segments.

The results obtained with the arterialness measure of Bülow et al. [6] look promising, while the fixed vessel segmentation is a drawback. In our proposed method, we use another arterialness measure that is calculated similarly to theirs, which will be described in detail in Chapter 3.

Another approach that uses anatomical properties of arteries and veins is from Mekada et al. [30]. Like Bülow et al. [6] they make use of the property that arteries and bronchi run in parallel. They extend it with the observation that arteries and bronchi run along the center of each lung segment, while veins run between bronchi near the border of each lung segment, as depicted in Figure 2.11a. They calculate two distances for each vessel voxel that are based on a 3D Generalized Voronoi Diagram (GVD).



(a) Schematic relation of arteries, veins and bronchi.

(b) Slice of the generalized Voronoi diagram.

(c) Rendering of the generalized Voronoi diagram.

Figure 2.11: The left image (a) shows the schematic relation of arteries, veins and bronchi as proposed by Mekada et al. [30]. The other two images show a slice (b) and a 3D rendering (c) of a *GVD* of bronchi segments that is used for calculating the distance of arteries and veins to their next bronchus. The images are taken from [30].

They start from a given vessel segmentation that is based on region growing with spherical structured elements. The seed points of the region growing must simultaneously satisfy the following three conditions: the point must be inside a given lung segmentation, it must have a value higher than -400 *HU*, and there must not be another point with more than -400 *HU* within a distance of 4 mm. The vessels near the hilum are excluded from the analysis because the anatomical properties for differentiation do not apply to vessels around the hilum.

For all vessel branches two features are calculated: the distance from the bronchus region to the vessel segment (Db) and the distance between the nearest border of the lung segment to the vessel (Dv). For calculating those two features an airway segmentation is needed, while an additional segmentation of the lung segments is needed as well for calculating Dv. As the lung segments cannot be seen directly in the *CT* images, they have to be approximated otherwise. The approximation is done by calculating the 3D *GVD* of the bronchial branches (see Figure 2.11b and 2.11c). This *GVD* calculates the nearest bronchial branch for each point inside the lung segmentation. With the airway segmentation and the *GVD*, Db as well as Dv can be calculated for each vessel point. Db should be smaller for arteries than for veins, as arteries run along bronchi, while Dv should be smaller for veins, as they run near the borders of a lung segment. Those two distances are measured for all vessel segments which are classified accordingly. For robustness, the segments are also grouped by their connection to the pulmonary artery.

Although the method of Mekada et al. [30] provides good results, a better vessel segmentation method could improve the separation results, as most errors are located where arteries and veins are in close proximity to each other. Furthermore, the *GVD* is a promising tool for grouping voxels into regions depending on their nearest seed points. In our proposed method we use the *GVD* for approximating the distribution of vessel trees, as described in Chapter 3.

2.3 Motivation for our Proposed Method

Most of the described A/V separation methods use a full vessel segmentation as input. As these segmentations are often constrained to be tree-like, they may contain spurious branches, while missing real ones. We therefore propose a method based on the works of Türetken et al. [46] that does not create a full segmentation of the vessel tree, but extracts multiple distinct subtrees that have a low probability of containing spurious branches. This implies that each subtree is well separated and either an artery or a vein. Due to their good separation, we can then focus on labeling them without further need of modification, like merging or pruning.

Additionally, most A/V separation methods contain a significant amount of user interaction. For example, Park et al. [34] also extract distinct subtrees and label them afterwards, but their automatic merging step still needs user interaction. Moreover, robust automatic seed point detection for either tracking (see Section 2.2.1) or voxel-based labeling (see Section 2.2.2) is hard to achieve. Kitamura et al. [20] proposed the only method that we know of, which is able to detect seed points for both arteries and veins automatically. Due to their machine learning approach and the necessary large amount of manual annotations, we have decided not to use it, but instead explore how well a low-level image processing method performs. We choose to use other anatomical features of the lung vasculature and adapted the arterialness measure from Bülow et al. [6]. One drawback of this arterialness measure is that although vessels with high arterialness values are most probably arteries, low arterialness values do not ensure that the vessel is a vein. Therefore, this feature alone is not sufficient for labeling all of the extracted subtrees and we propose to combine it with a novel anatomical measure that enforces a uniform distribution of arteries and veins throughout the lungs. We calculate the distribution of the vessels by quantifying the contact surface of regions of a *GVD* that is generated with the extracted vessel subtrees. By combining these two measures, we are able to label each distinct subtree as being an artery or a vein and to create the final fully automatic A/V segmentation.

3 Method

Contents

In this chapter the proposed algorithm is explained in detail. As the algorithm makes use of integer programming, a short introduction to this optimization technique is given beforehand in Section 3.1. The outline of the proposed algorithm is shown in Figure 3.1. It expects a contrast-enhanced thoracic Computed Tomography (CT) image as input. After a lung segmentation based on Helmberger et al. [15] (see Section 3.2), subsequent processing is performed for left and right lung independently. Next, an overcomplete four-dimensional (4D) graph of vessel paths is calculated, that contains the actual sought-after connections, but also spurious ones (see Section 3.3). Afterwards, multiple distinct subtrees are extracted from this graph by solving an integer program that identifies anatomical meaningful vessel paths and connects them to tree-like structures (see Section 3.4). The final A/V labeling of these distinct subtrees is calculated by a second integer program that exploits two anatomical features of vessels in the lung, i.e., the uniform distribution of arteries and veins throughout the lungs, and the proximity of arteries and bronchi (see Section 3.5).

3.1 Integer Programming

As the following parts of the proposed algorithm use integer programming for obtaining the results, a short introduction to integer programming is given in this section. Inte-

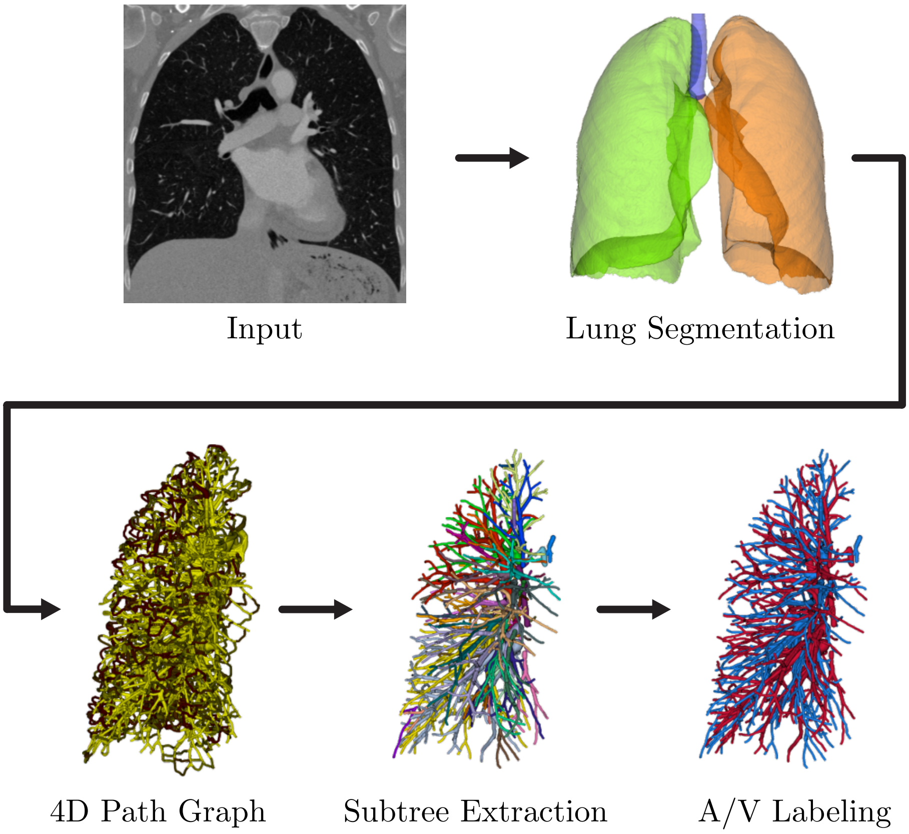


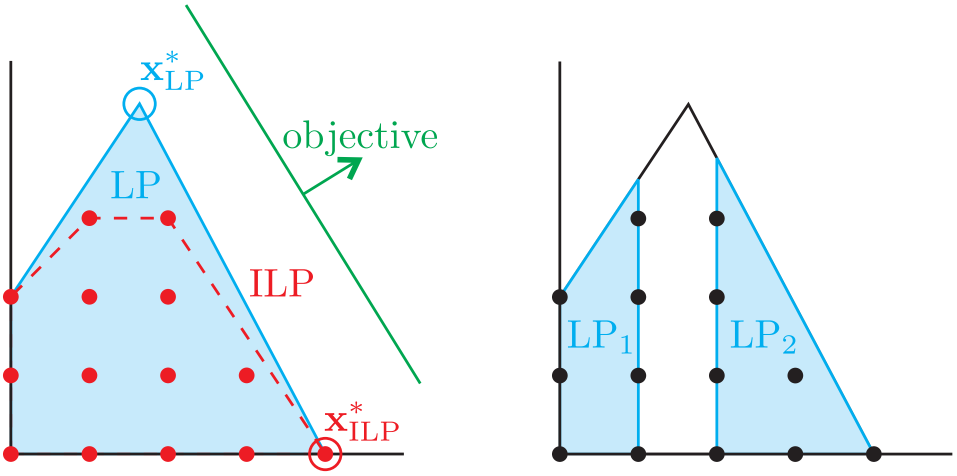
Figure 3.1: Flow chart of the proposed algorithm. The algorithm uses thoracic contrast-enhanced Dual-Energy Computed Tomography (DECT) images as input. A lung segmentation is created and the A/V separation pipeline consisting of three steps, namely the 4D path graph, the subtree extraction, and the final A/V labeling, is applied to both sides of the lung independently.

ger programming aims to maximize a constrained objective function where the objective function as well as the constraints contain variables that are restricted to integers [49].

Integer programs can be grouped depending on the structure of the objective function and the constraints. If the objective function contains additional quadratic terms, they are called ”integer quadratic programs”, if the constraints contain quadratic terms, they are called ”quadratically constrained”. Additionally, if some of the variables are real-valued, they are called ”mixed-integer programs”.

The most basic case of integer programs has a linear objective function with linear constraints. A general overview of integer linear programming is given in Section 3.1.1. One commonly used method for solving integer programs is the so-called ”branch-and-bound” algorithm, that is explained in Section 3.1.2. As the integer programs used by our proposed method have quadratic objective functions, integer quadratic programming is introduced in Section 3.1.3.

3.1.1 Linear Programming and Integer Linear Programming



(a) Integer linear programming.

(b) Branching step.

Figure 3.2: The integer linear program ILP and the linear program relaxation LP are shown in (a), where the feasible space of the linear program is shown in blue. The objective function is shown in green. The optimal solutions of the linear program $\mathbf{x}^*_{\mathrm{LP}}$ and the linear integer program $\mathbf{x}^*_{\mathrm{LP}}$ are far apart. The second image (b) shows the first branching step of the branch-and-bound algorithm that splits the first linear program relaxation LP into the two new ones LP_1 and LP_2.

Integer linear programming is highly related to linear programming, as shown in Figure 3.2. As solving a linear integer program makes use of solving the corresponding linear program, we first define linear programs. The canonical form of a linear program is defined as

$$\max_{\mathbf{x}} \ \mathbf{c}^T\mathbf{x} \tag{3.1}$$

subject to:

$$A\mathbf{x} \leq \mathbf{b}, \tag{3.2}$$

with $\mathbf{x} = (x_1, \ldots, x_n)^T$ being the column vector of the variables $x_i \in \mathbb{R}$ to optimize and $\mathbf{c} = (c_1, \ldots, c_n)^T$ being the column vector of the factors $c_i \in \mathbb{R}$ for the linear terms. The column vector $\mathbf{b} = (b_1, \ldots, b_m)^T$ with $b_i \in \mathbb{R}$ and the matrix $A = (a_{ij})$ with $a_{ij} \in \mathbb{R}$ are used for defining the linear constraints. Objective functions that need to be minimized instead of maximized can be converted to the canonical form (3.1) by negating their linear factors c_i. Equality (=) or greater-equal ($\geq$) constraints can be converted to the canonical form (3.2) by including additional lower-equal ($\leq$) constraints or negating the corresponding factors of the linear constraints, respectively. The aim is to find the optimal solution $\mathbf{x}^*$ that maximizes the objective function while being feasible at the same time. A certain $\mathbf{x}$ is called feasible, if it is in the space of feasible solutions, i.e., it does not violate any constraints. The optimal solutions of linear programs always lie on intersections of multiple constraints and many efficient algorithms exist to solve linear programs [47].

The only change in the definition of integer linear programs as compared to linear programs (see Equation (3.1) and (3.2)) is that the variables x_i are constrained to be integral, i.e., $x_i \in \mathbb{Z}_0^+$. As the optimal solution is not guaranteed to lie on an intersection of constraints, in contrast to linear programs, integer linear programming is much harder to solve and was proven to be NP-hard. Therefore, various methods for solving integer programs have been proposed in the past, where branch-and-bound and its derived methods have shown to lead to good results.

3.1.2 Branch-and-Bound

The branch-and-bound algorithm uses a divide-and-conquer method for solving integer programs. At first, the algorithm drops the integral constraints of the vector $\mathbf{x}$ to create a linear program. Afterwards, the solution for the corresponding linear program of this so-called "LP-relaxation" step is calculated with solvers for linear programming. Since the result of the linear program is less constrained than the result of its corresponding integer program, it serves as an upper bound for the integral solution, i.e., the solution of the integer program is guaranteed not to be greater than the solution of the associated LP-relaxation. As the result of the LP-relaxation usually does not have integral coordinates, as depicted in Figure 3.2a, additional constraints have to be added in order to restrict the solution to integers. This is done in the branching step of the algorithm, where two constraints along integral coordinates of one variable are added, spliting the current linear program into two new ones. These two constraints bind a real-valued variable x_i to its next integers, so that the solution of the current linear program becomes infeasible (see Figure 3.2a). For example, if the solution for $x_i = 1.65$, two new linear programs are created where the first has the additional constraint $x_i \leq 1$ and the second has the additional constraint $x_i \geq 2$. These linear programs are then solved and split up again to new linear programs in the same way as before until a feasible integral solution is found. This procedure creates a search tree, where the integral solutions are in the leaf nodes.

This tree has to be enumerated completely in order to generate the optimal solution. The enumeration should be performed in depth-first instead of breadth-first search for several reasons [47]. The sooner a feasible integral solution is found, the better, as it serves as an intermediate result and as a lower bound for the optimal solution. With the best-so-far integral solution, all branches can be pruned, where the solution of the linear program relaxation, which serves as an upper bound in this branch, is lower. Furthermore, the current solution gap, i.e., the difference between the current solution and the upper bound, can be calculated. This can be used as a quality measure of the intermediate result. If the lower bound and the upper bound are the same, the current best-so-far solution is the optimal one and the search tree does not need to be enumerated any further. Another reason to prefer depth-first search is that it favors linear program solvers. By moving one step deeper in the search tree, each subsequent linear program is obtained from the preceding one by simply adding another constraint. This helps most linear program solvers, as they do not need to be started again from scratch, but can refine their previous results.

In practice, the plain branch-and-bound algorithm is not used without modifications. Additional heuristics, improvements in enumerating the search tree, or other techniques, like cutting planes, are used for improving performance and reducing runtime of the branch-and-bound algorithm [49].

3.1.3 Integer Quadratic Programming

The integer programs that are used by the proposed method use quadratic objective functions with linear constraints that are defined as

$$\max_{\mathbf{x}} \ \mathbf{c}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T G\mathbf{x} \tag{3.3}$$

subject to:

$$A\mathbf{x} \leq \mathbf{b}, \tag{3.4}$$

with $\mathbf{x}$, $\mathbf{c}$, $\mathbf{b}$, and A having the same definition as for integer linear programs (see Section 3.1.1). The additional matrix $G = (g_{ij})$ defines the factors $g_{ij} \in \mathbb{R}$ for the quadratic terms of the objective function.

The difference to linear integer programs is that the objective function contains quadratic terms. Quadratic integer programs can be solved with branch-and-bound as well, but instead of an IP-relaxation step, a QP-relaxation step is used that converts the integer quadratic program to a real-valued quadratic program. Although quadratic programs are more complex than linear ones, there exist several efficient algorithms for solving them [47].

3.2 Lung Segmentation

The proposed algorithm for A/V separation starts with calculating a lung segmentation in order to restrict the vessel extraction to the lung regions. Furthermore, a separation in left and right lung is of interest, as the proposed algorithm creates the A/V segmentation for both sides of the lungs independently. The lung segmentation and separation in left and right lungs are based on the method presented in Helmberger et al. [15].

As a first step, the lung region is binarized using Otsu's method [32] in order to separate bright (tissue) and dark (air, lungs) regions in the *CT* image. The largest connected dark region inside the bright region specifies an initial segmentation, where trachea, left and right lungs are merged. The first branching point of the trachea is defined as the starting point of left and right lung. To separate the lungs, the shortest path to the trachea is calculated for every voxel inside the segmentation. The label of a voxel is dependent on the direction from which the path emerges to the defined starting point. The cost function of the shortest path algorithm is based on image gradients of neighboring voxels as well as intensity values of the lungs and the airways. Finally, to remove holes arising from vessels or other bright structures inside the lung segmentation, morphological closing is applied to both lungs.

All subsequent steps of the proposed algorithm are calculated for each side of the lungs independently.

3.3 4D Path Graph

In order to prepare extraction of arterial and venous vessel trees, an overcomplete graph G of 4D paths, that contains spurious as well as the sought after connections, is generated. As proposed by Li and Yezzi [24], a 3D vessel surface can be seen as a 4D curve along the centerline of a vessel, where the first three dimensions represent the spatial coordinates and the fourth represents its radius. The segmentation of the vessel can be calculated by taking the envelope of the list of spheres traversing this 4D curve, as depicted in Figure 3.3. An advantage of representing a vessel as a 4D path is that it expresses the vessel as a centerline and as a segmentation at the same time.

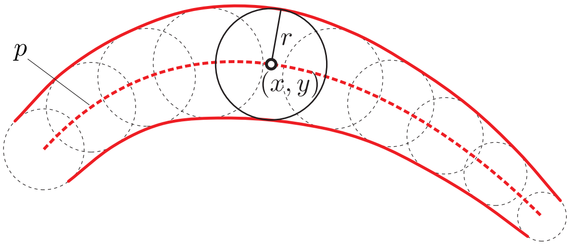


Figure 3.3: A 2D vessel represented as a 3D curve (x, y, r) along its centerline. By taking the envelope of the circles traversing this path, a segmentation can be constructed. For better visualization, a 2D vessel is shown, while the extension to 3D vessels using spheres instead of circles is straightforward.

The single parts of the 4D path graph calculation, which are based on the work of Türetken et al. [46], are shown in Figure 3.4. The points of the graph G are extracted from a vessel-enhanced image (see Section 3.3.1) and connected in their local neighborhood (see Section 3.3.2). The 4D paths between two neighboring points in the graph G are calculated by minimizing a geodesic distance, that is based on the vessel-enhanced image (see Section 3.3.3).

3.3.1 4D Vessel Enhancement

Vessels can be locally approximated by differently sized and oriented tubes. Therefore, multiple filters that enhance tube-like structures have been presented in the past. In

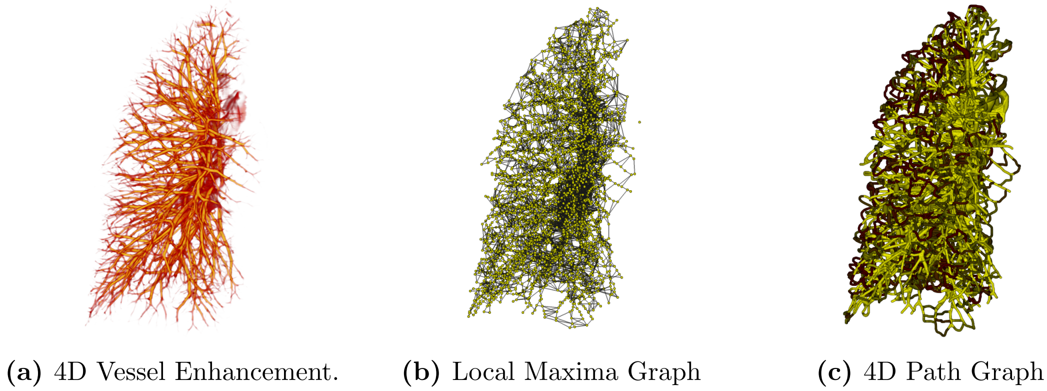

(a) 4D Vessel Enhancement. **(b)** Local Maxima Graph **(c)** 4D Path Graph

Figure 3.4: The steps of calculating the 4D Path Graph. At first, a 4D tubularity filter enhances vascular structures. In (a) a maximum intensity projection over the radius is performed to reduce the 4D image to a 3D image for visualization purpose. In the next step, local maxima are extracted from this vessel-enhanced image (b). The last step is to connect neighboring local maxima by 4D tubular paths to create the final 4D path graph (c).

our method we use the tubularity filter Optimally-Oriented Flux (OOF) for this task, as proposed by Law and Chung [22], with an implementation from Benmansour et al. [3].

The oriented flux in a particular direction projects the image gradient on the boundary surface of a sphere with a certain position and radius to its center. The oriented flux along direction $\mathbf{d}$ of a sphere S_r with position $\mathbf{x}$ and radius r is defined as

$$f\left(\mathbf{d},\mathbf{x},r\right) = \int_{\partial S_r} \left(\left(\nabla\left(G_\sigma * I\right)\left(\mathbf{x}+r\mathbf{n}\right)\cdot\mathbf{d}\right)\mathbf{d}\right)\cdot\mathbf{n}\,dA, \tag{3.5}$$

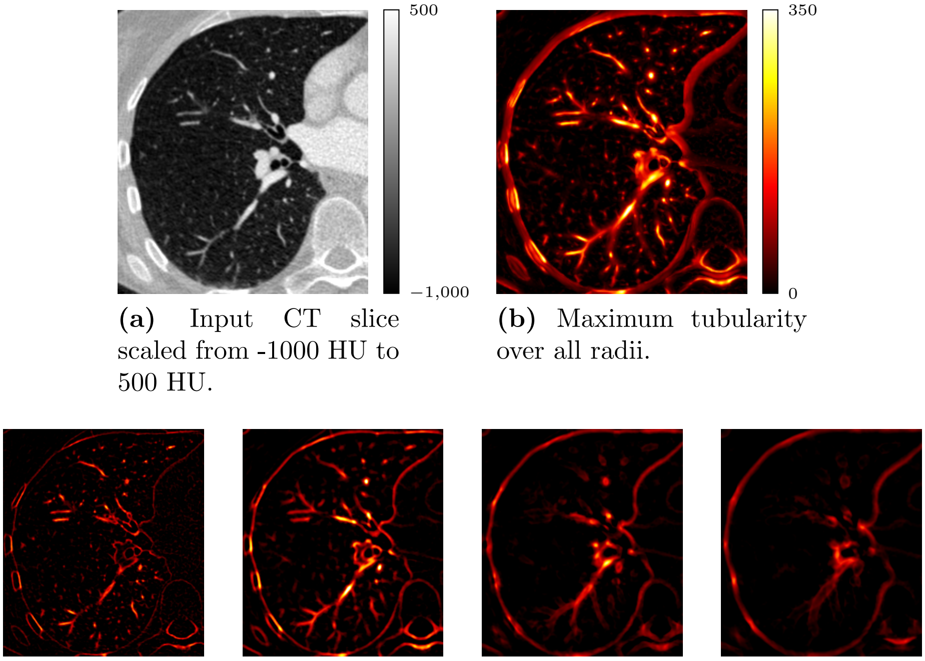
where $\nabla\left(G_\sigma * I\right)(\cdot)$ is the gradient of the smoothed input image $I(\cdot)$ that has been smoothed with the Gaussian function $G_\sigma(\cdot)$ and sufficiently small σ, $\mathbf{n}$ is the outward unit normal of ∂S_r, and dA is the infinitesimal area on ∂S_r. The optimally oriented flux aims to find the direction of the projection axis $\mathbf{d}$ that minimizes the oriented flux, which is defined as

$$\arg\min_{\mathbf{d}} f\left(\mathbf{d},\mathbf{x},r\right). \tag{3.6}$$

This projection axis $\mathbf{d}$ also estimates the direction of the tube, i.e., in our case the direction of the vessel. Using the convergence theorem, it can be shown that $f\left(\mathbf{d},\mathbf{x},r\right)$ is a quadratic form on $\mathbf{d}$ and can be calculated with a convolution

$$f\left(\mathbf{d},\mathbf{x},r\right) = \mathbf{d}^T\left\{I * F_r(\mathbf{x})\right\}\mathbf{d}, \tag{3.7}$$

where $F_r(\mathbf{x})$ is the oriented flux filter [3]. The minimum of $f\left(\mathbf{d},\mathbf{x},r\right)$ can be calculated by differentiating $f\left(\mathbf{d},\mathbf{x},r\right)$ with respect to $\mathbf{d}$ and solving a generalized eigenvalue problem [3].



(a) Input CT slice scaled from -1000 HU to 500 HU.

(b) Maximum tubularity over all radii.

(c) $r = 0.5$

(d) $r = 2$

(e) $r = 3.5$

(f) $r = 5$

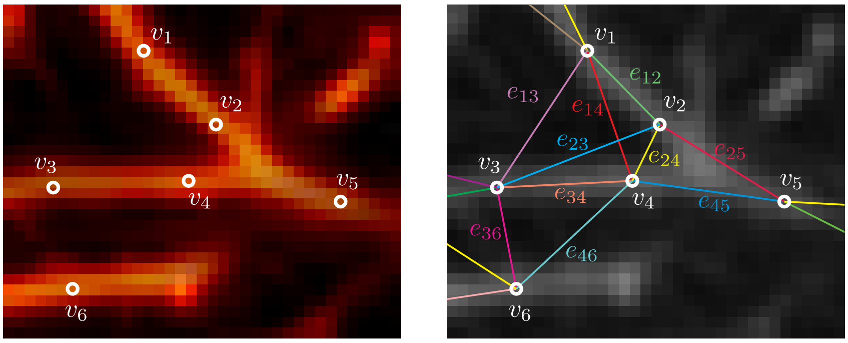
Figure 3.5: Visualization (a) shows a slice of the input *CT* image and (b) its 4D vessel enhancement result. The images in the bottom row ((c), (d), (e), and (f)) show the vessel enhancement output for various radii. The final 4D vessel enhancement image is composed of the 3D vessel enhancement images for all computed radii.

As the thickness of vessels in the lungs is varying, the *OOF* has to be calculated for spheres with different radii. By calculating the *OOF* for multiple spheres within a discrete radius range, each of the resulting three dimensional *OOF* output images can be merged to the final 4D multi-scale *OOF* result. An example of this multi-scale analysis for a single *CT* slice is shown in Figure 3.5.

The result of this multi-scale analysis is a 4D vessel-enhanced image $\mathcal{T}(\cdot)$, where the first three dimensions correspond to the three spatial locations and the fourth dimension corresponds to the radius of the vessel. Particularly, the filter estimates for each 4D coordinate the certainty, that a vessel with a specific radius centers a specific position, as well as its direction. The result of this filter will be used for the subsequent 4D path calculation. A 3D maximum intensity projection over multiple radii is shown in Figure 3.4a.

3.3.2 Local Maxima Graph Calculation

The next step is to generate a local maxima graph $G = (V, E)$ of the 4D vessel-enhanced image $\mathcal{T}(\cdot)$. The purpose of this step is to extract a large number of points that lie with high certainty on the centerline of vessels, and afterwards connect them in their local neighborhood.



(a) Local maxima.

(b) Neighbor edges.

Figure 3.6: Example of a small local maxima graph extracted from the tubularity image. The left image (a) shows the regularly spaced local maxima v_i of a tubularity image, whereas the right image (b) shows the edges e_{ij} of neighboring points.

Each vertex $v_i \in V$ of the local maxima graph G corresponds to a point $\mathbf{p}$ that is a local maxima in the enhanced image, as visualized in Figure 3.6a. Points are added to the set V in descending order of their tubularity values of the vessel-enhanced image. To prevent points from being too close to each other, all points within a certain Euclidean distance d_{maxima} to previously added points are invalidated, i.e., are no future candidates for extraction. Then, the point with the next largest tubularity value, that has not been invalidated yet, is added and its local neighborhood is invalidated as well. This process is iterated as long as all points above a certain threshold in the vessel-enhanced image have been examined.

Afterwards, the edges E of the graph G have to be calculated, as visualized in Figure 3.6b. Every point of V is connected with every other point of V, whose Euclidean distance is below a certain distance value $d_{neighbor}$. An example 3D rendering of the full local maxima graph is shown in Figure 3.4b.

The resulting undirected graph $G = (V, E)$ consists of the extracted maxima $\mathbf{p}_i$ as vertices $v_i \in V$, where $\forall v_i, v_j \in V$ and $i \neq j : \|\mathbf{p}_i, \mathbf{p}_j\| \geq d_{maxima}$ and the connection of neighboring points as edges E, where $\forall e_{ij} \in E :\ v_i,\ v_j \in V$ and $\|\mathbf{p}_i, \mathbf{p}_j\| \leq d_{neighbor}$.

3.3.3 4D Path Calculation

After generating the local neighborhood graph G, 4D vessel paths p_{ij} between connected vertices $v_i \in V$ and $v_j \in V$ are calculated. These paths are constructed to minimize an energy functional of the previously calculated 4D vessel-enhanced image, as proposed by Benmansour et al. [3]. An example procedure of extracting a single path is shown in

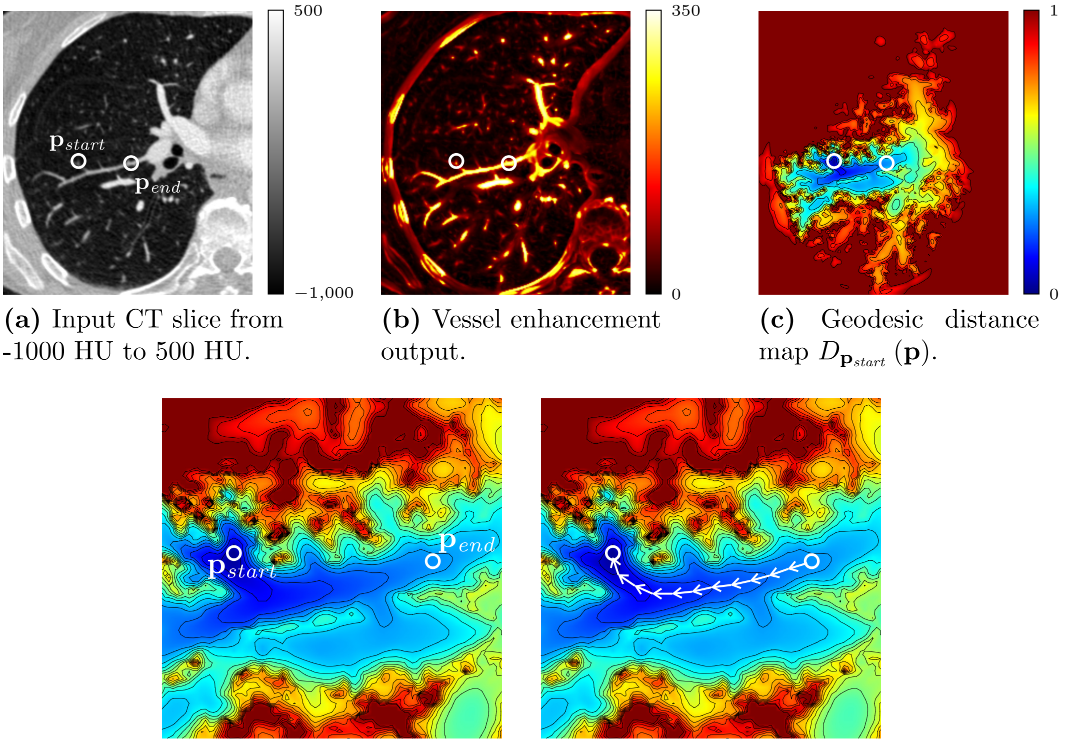
Figure 3.7. The energy functional for a path ρ parametrized over s is defined as

$$E(\rho) = \int \mathcal{T}(\rho(s))\, ds, \tag{3.8}$$

with $\mathcal{T}(\cdot)$ being the 4D tubularity image. The geodesic distance d from a point $\mathbf{p}_{start}$ to the point $\mathbf{p}_{end}$ is defined as

$$d(\mathbf{p}_{start}, \mathbf{p}_{end}) = \min_{\rho \in \mathcal{P}_{\mathbf{p}_{start}, \mathbf{p}_{end}}} \left\{ \int \mathcal{T}(\rho(s))\, ds \right\}, \tag{3.9}$$

where $\mathcal{P}_{\mathbf{p}_{start}, \mathbf{p}_{end}}$ is the set of all paths starting from point $\mathbf{p}_{start}$ and ending in point $\mathbf{p}_{end}$.



(a) Input CT slice from -1000 HU to 500 HU.

(b) Vessel enhancement output.

(c) Geodesic distance map $D_{\mathbf{p}_{start}}(\mathbf{p})$.

(d) Zoom-in of geodesic distance map.

(e) Gradient descent path from $\mathbf{p}_{end}$ to $\mathbf{p}_{start}$.

Figure 3.7: Example of extracting a path from $\mathbf{p}_{start}$ to $\mathbf{p}_{end}$ as shown in (a). The fast marching algorithm uses a vessel-enhanced image (b) to create a geodesic distance map (c), from which a zoom-in is shown in (d). The final path of (e) is calculated with a gradient descent on the distance map. For better visualization, these images show a 2D path extraction procedure in a single slice that uses a 2D vessel enhancement. In the actual proposed algorithm, all extracted paths are calculated with the full 4D vessel enhancement image and are therefore four-dimensional.

The distance map $D_{\mathbf{p}_{start}}(\mathbf{p})$ from the fixed start-point $\mathbf{p}_{start}$ to an arbitrary point $\mathbf{p}$ inside the image can be seen as a map of arrival times of a front propagating from the source $\mathbf{p}_{start}$ with velocity $\frac{1}{\mathcal{T}(\cdot)}$. This map satisfies the eikonal equation

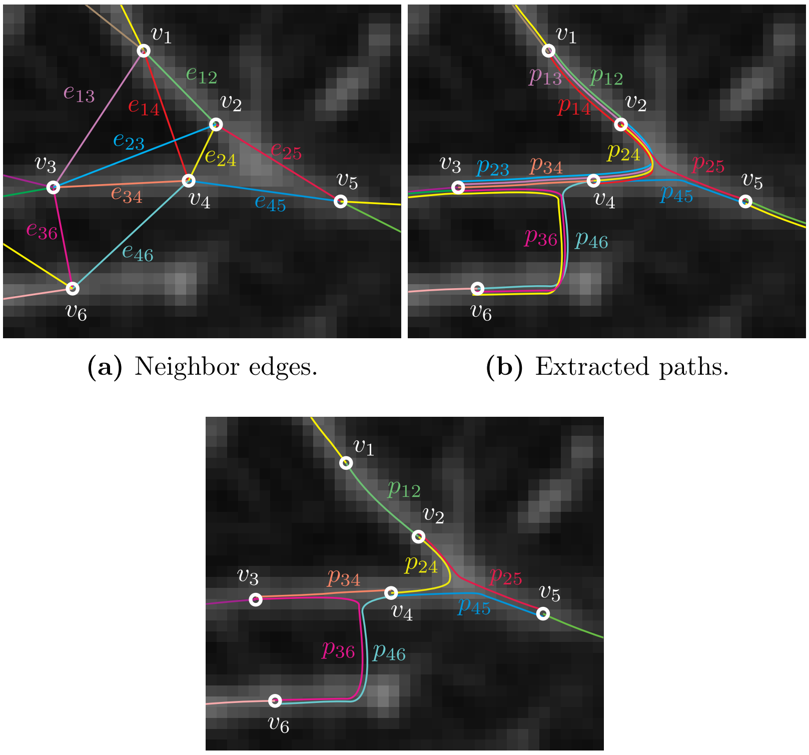
$$\left\|\nabla D_{\mathbf{p}_{start}}\left(\mathbf{p}\right)\right\| = \mathcal{T}\left(\mathbf{p}\right), \tag{3.10}$$

with $D_{\mathbf{p}_{start}}\left(\mathbf{p}_{start}\right) = 0$ for all $\mathbf{p}$ in the image domain [3]. The solution of this eikonal equation can be calculated with the fast marching algorithm [43] using the 4D tubularity image as a speed function, as visualized in Figure 3.7c.

As $D_{\mathbf{p}_{start}}$ has exactly one global minimum, which is the start-point $\mathbf{p}_{start}$ where $D_{\mathbf{p}_{start}}\left(\mathbf{p}_{start}\right) = 0$, the shortest path from any point $\mathbf{p}$ to the start-point can be calculated by a gradient descent method (see Figure 3.7e). This leads to a gradient descent path that starts in point $\mathbf{p}_{end}$ and ends in point $\mathbf{p}_{start}$. Furthermore, the points of the paths do not have to be on integer coordinates, but may be on subvoxel coordinates. The step size of the gradient descent method is set to a fixed value, to get a discrete path p that is composed of a list of points with a certain distance between them. Every point of the gradient descent path, except the end-point, has this fixed distance to the previous one. The final discrete path p from point $\mathbf{p}_{start}$ to $\mathbf{p}_{end}$ is the path that the gradient descent method took, but in reverse order. Its geodesic length $l\left(p\right)$ is the geodesic distance $d\left(\mathbf{p}_{start}, \mathbf{p}_{end}\right)$ from its start to its end point.

This path calculation procedure is repeated in order to create a 4D path p_{ij} for every edge $e_{ij} \in E$ in the local neighborhood graph G, as depicted in Figure 3.8a and 3.8b. Many of those paths are highly overlapping, because the maximum distance of vertices considered as neighbors is set relatively large in order not to miss any real vessel paths. This leads to many paths running along the same part of a vessel and containing other vertices of the graph that cannot be filtered before knowing the paths themselves. As the extracted vascular tree structures will be invalid, if multiple paths run along the same centerline, a filtering step is performed, that removes every path that encloses any other path entirely. This does not delete information of the full vessel segmentation itself, because the removed paths are completely composed of other paths that still remain in graph G (see Figure 3.8c). Furthermore, the subsequent steps of the proposed algorithm make a distinction between the opposing edges e_{ij} and e_{ji} and their corresponding paths p_{ij} and p_{ji}. Therefore, the edge e_{ji} is added to the graph G and its path p_{ji} is generated by traversing path p_{ij} in reverse order.

After this step, a final directed overcomplete vascular graph that contains meaningful as well as spurious branches has been calculated and is ready for subtree extraction. An example 3D visualization of the full 4D path graph is shown in Figure 3.4c. Note that only the paths p_{ij} and not their opposing ones p_{ji} are visualized, as they run along the same points, but in opposite directions.



(a) Neighbor edges.

(b) Extracted paths.

(c) Filtered paths.

Figure 3.8: Example of a small local maxima graph (a) and the extracted paths between all neighboring vertices (b). The filtering step removes all paths that fully enclose any other path and leads to the remaining ones (c). Only the centerlines of the paths are shown. For better visualization, the images show 2D (x, y) paths instead of the actual 4D (x, y, z, r) ones. Additionally, the paths are slightly shifted in order to make them visible.

3.4 Subtree Extraction

In order to identify anatomically meaningful vessel trees and prepare the input for A/V separation, we extract a set of connected subtrees from the overcomplete maxima graph G, using an optimization procedure based on integer programming, similar to Türetken et al. [46]. The proposed integer program calculates a configuration of binary variables that minimizes a constrained objective function (see Section 3.4.1) of weights of adjoining vessel paths (see Section 3.4.2). In contrast to the method of Türetken et al., we extract multiple trees at a time and do not need explicitly declared root nodes. The root nodes are calculated implicitly by minimizing the objective function, as explained in Section 3.4.3. Finally, the subtrees are resampled and the actual branching points of the vessels are calculated (Section 3.4.4). An example output of the subtree extraction is shown in Figure 3.9.

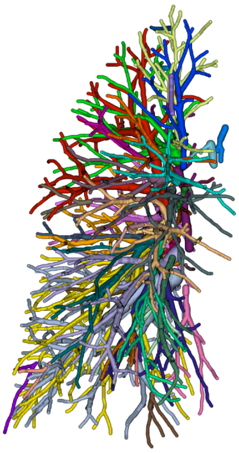


Figure 3.9: Result of the subtree extraction where every subtree is visualized in a different color.

3.4.1 Objective Function

The quadratic objective function used for subtree extraction incorporates two binary variables for each directed edge e_{ij} in the local maxima graph G. The first variable, t_{ij}, is set to 1, if and only if the edge e_{ij} is part of an extracted subtree. The second variable, r_{ij}, is set to 1, if and only if the edge e_{ij} is the root edge of a subtree, i.e., the edge with no predecessors. As not every configuration of variables creates feasible tree-like structures, the objective function has to be restricted by several constraints. The objective function is defined as

$$\operatorname*{arg\,min}_{t,r} \sum_{e_{ij},e_{jk}\in E} w_{ijk}\; t_{ij}\; t_{jk} \;+\; \sigma \sum_{e_{ij}\in E} r_{ij} \tag{3.11}$$

subject to:

$$\textstyle\sum_{e_{hi}\in E} t_{hi} + r_{ij} \geq t_{ij} \qquad \forall e_{ij}\in E, \tag{3.12}$$

$$\textstyle\sum_{e_{hi}\in E} t_{hi} + r_{ij} \leq 1 \qquad \forall e_{ij}\in E, \tag{3.13}$$

$$t_{ij} \geq r_{ij} \qquad \forall e_{ij}\in E, \tag{3.14}$$

$$t_{ij} + t_{ji} \leq 1 \qquad \forall e_{ij}\in E, \tag{3.15}$$

where $t_{ij}, r_{ij} \in \{0,1\}$, $w_{ijk}\in\mathbb{R}$, and $\sigma\in\mathbb{R}_0^+$.

The first term of Equation (3.11) can be interpreted as follows: t_{ij} is either active ($t_{ij} = 1$), which means that the corresponding edge e_{ij} is part of the resulting treelike structures, or inactive ($t_{ij} = 0$), if it is not.

For every adjoining edge pair $\langle e_{ij}, e_{jk}\rangle$, with e_{ij} and e_{jk} connected at the vertex v_j (see Figure 3.6), an edge pair weight w_{ijk} is calculated. We use weights of edge pairs instead of weights of single edges to easily incorporate the geometric relation of adjoining edge paths as well as the weights of single edge paths into one combined weight. So every edge pair weight, where both edges e_{ij} and e_{jk} are part of the final vessel structure is added to the sum. This means, that edge pair weights of correct edges should be negative, while edge pair weights of incorrect edges should be positive. A more detailed description of the edge pair weight can be found in Section 3.4.2.

The second term of Equation (3.11) limits the number of distinct subtrees by employing a constant root-penalizing factor σ. As every subtree has exactly one root edge with $r_{ij} = 1$, which is modeled in the constraints, this constant value σ increases the objective function for every created subtree. This term is needed to prevent the creation of too many subtrees, as well as to facilitate more branches. A more detailed explanation of the root detection is given in Section 3.4.3.

The objective function has to be constrained to allow only tree-like structures in the results. The first constraint (3.12) assures that, if at least one preceding edge e_{hi} is active or the current edge e_{ij} is a root edge, the current edge may (but does not need to) be active as well. The second constraint (3.13) assures that at most one preceding edge e_{hi} is active, or the edge is a root edge, or neither of both. The third constraint (3.14) assures that each root edge is an active edge as well. The combination of those three constraints enforces tree-like structures, by ensuring that each active edge either has exactly one predecessor edge, or is a root edge of a subtree. The fourth constraint (3.15) guarantees that if an edge e_{ij} is active, its opposing edge e_{ji} is not active at the same time.

3.4.2 Edge Pair Weight

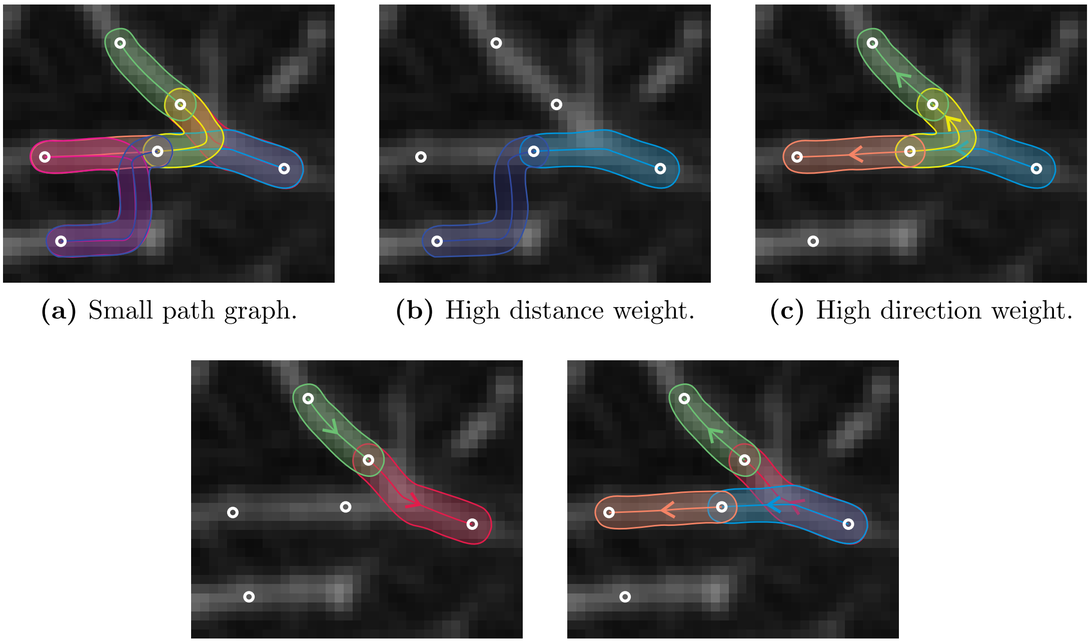
The choice of the edge pair weight that is used in the objective function (3.11) is critical for the outcome of the integer program, as it needs to distinguish between physically reasonable and unreasonable vessel paths (see Figure 3.10). As mentioned in the previous section, we use a combined weight of adjoining edge pairs $\langle e_{ij}, e_{jk} \rangle$ to incorporate a geometric relationship of adjoining vessel paths. It has to be noted that the edges of the 4D path graph are directed, i.e., e_{ij} and e_{ji} and their corresponding paths p_{ij} and p_{ji} are opposing, which is crucial for correct edge pair weight calculation. Moreover, edge pairs can only be calculated for adjoining edge pairs, i.e., the end-point of the first edge has to be the start-point of the second one.

The edge pair weight of the edges e_{ij} and e_{jk} with their corresponding paths p_{ij} and p_{jk} is defined as

$$w_{ijk} = \alpha\, w_{ijk}^{distance} + \beta\, w_{ijk}^{direction} + \gamma\, w_{ijk}^{radius} + \delta, \tag{3.16}$$

with $\alpha, \beta, \gamma, w_{ijk}^{distance}, w_{ijk}^{direction}, w_{ijk}^{radius} \in \mathbb{R}_0^+$, and $\delta \in \mathbb{R}^-$.

As the objective function (3.11) will be minimized, weights of physically reasonable paths need to be negative. Otherwise the trivial solution, i.e., no path extracted, would always minimize the objective function. The only term of Equation (3.16) that may be negative is δ, all other terms are positive. The choice of δ is crucial for getting good weights, as its value has to be just as negative to make weights of reasonable paths slightly negative,



(a) Small path graph. **(b)** High distance weight. **(c)** High direction weight.

(d) High radius weight. **(e)** Extracted subtree.

Figure 3.10: Example of a small path graph. The edge pair weights should be defined such that minimizing the objective function (3.11) of graph (a) leads to subtree (e). The images (b), (c) and (d) show example paths pairs with high distance weight, high direction weight, and high radius weight, respectively. For better visualization, the images show 3D (x, y, r) instead of 4D (x, y, z, r) paths, where the radius coordinate r is visualized as the border of the vessel.

while keeping weights of unreasonable ones positive. It is constant for all edge pairs and used as a negative base value. The direction weight $w_{ijk}^{direction}$, the radius weight w_{ijk}^{radius}, and the distance weight $w_{ijk}^{distance}$ range between 0 and 1 and penalize unreasonable paths by making their total edge pair weight w_{ijk} positive. The constant factors α, β, and γ shift the influence of the single weights in the total sum. We will show in Section 4.1.2, how the values for α, β, γ, and δ are set.

The first weight of Equation (3.16), the distance weight $w_{ijk}^{distance}$ penalizes paths with a large geodesic distance, as visualized in Figure 3.10b, and is defined as

$$w_{ijk}^{distance} = \max\left(l\left(p_{ij}\right), l\left(p_{jk}\right)\right), \tag{3.17}$$

where $l\left(p_{ij}\right)$ and $l\left(p_{jk}\right)$ are the geodesic lengths of paths p_{ij} and p_{jk} that were calculated by the path extraction algorithm (see Section 3.3.3). These values represent a minimal geodesic distance on the 4D tubularity image between the two end-points of a path, i.e., the geodesic length of the path itself. The distance weight of the edge pair is the maximum value of the geodesic lengths of the two adjoining paths.

As the calculated geodesic distance does not incorporate the direction of paths at all, path pairs with unusual directions or direction changes, as shown in Figure 3.10c, do not get worse weights than path pairs with reasonable directions. Therefore, $w_{ijk}^{direction}$, the second weight of Equation (3.16), takes the geometric relationship of path pairs into account and penalizes path pairs with unusual directions. It is defined as

$$w_{ijk}^{direction} = 1 - \frac{1}{(n-1)(m-1)} \sum_{i=1}^{n-1} \sum_{j=1}^{m-1} \mathbf{u}_i \cdot \mathbf{v}_j, \tag{3.18}$$

where $\mathbf{u}_i$ is the series of $n-1$ normalized direction vectors between the n points of path p_{ij} and where $\mathbf{v}_i$ is the series of $m-1$ normalized direction vectors between the m points of path p_{jk}. Equation (3.18) quantifies the orientation similarity of two paths by calculating the mean of all dot products of directions of the first path with the second path. In the best case, i.e., when both paths are collinear straight lines, the mean of all dot products is equal to 1. The more differences in direction are present, the lower the mean of dot products will be. As the direction weight should be 0 in the best case, the final weight is calculated by subtracting this mean from 1.

The last weight of Equation (3.16), the radius weight w_{ijk}^{radius}, helps detecting the correct direction of a path pair. In the lungs, vessel radii are usually decreasing with distance from the heart. Therefore, the radius weight penalizes path pairs, where the radius is increasing, as visualized in Figure 3.10d. It is defined as

$$w_{ijk}^{radius} = \max\left(1 - \frac{r^{start}(p_{ij})}{r^{end}(p_{jk})}, 0\right), \tag{3.19}$$

where $r^{start}(p_{ij})$ is the radius of the start-point of path p_{ij} and $r^{end}(p_{jk})$ is the radius of the end point of path p_{jk}. When the radius is increasing from start to end, their ratio is greater than 1, otherwise it is less than 1. To give higher weights to path pairs with increasing radii, this ratio is subtracted from 1. As we do not want to prefer path pairs that have decreasing radii, which would be negative after this subtraction, we set the minimum of w_{ijk}^{radius} to 0. In the best case, i.e., path pairs have equal or decreasing radii, the radius weight equals 0, while in any other case the weight becomes positive.

3.4.3 Subtree Root Detection

As mentioned before, the roots of the subtrees are calculated implicitly by minimizing the objective function (3.11). The objective function combined with the constraints assures, that only tree-like structures are extracted from the graph G, while every of these tree-like structures has exactly one root node. Although the extraction of the subtrees and their root detection is done simultaneously by minimizing the objective function, for easier understanding, it can be considered as consisting of two independent parts: the extraction of the subtrees, and the root edge detection.

The first part, the extraction of the tree-like structures, minimizes the total weights of all extracted subtrees. The total weight of a single subtree is the sum of all of its edge pair weights plus the root penalizing value σ (see Equation (3.11)). Every subtree is created by concatenating paths until no other path, that would reduce the total weight of the subtree, can be found. It has to be noted that a path will only be appended to this specific subtree, if it is not better - based on the edge pair weights - to append it to any other subtree.

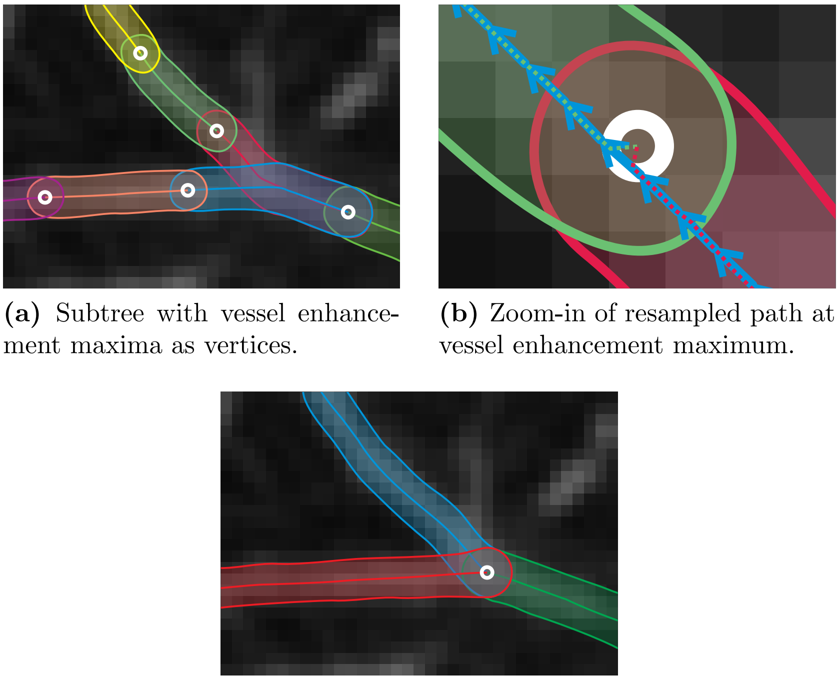
The second part is the root edge detection. Generally, the root of a tree is the node without any predecessors. Therefore, without a previously defined root node, every leaf of the subtree could be seen as its root. If we take the fully expanded subtree as fixed, the aim is now to find the root node that minimizes the weight of the tree. A combination of $w_{ijk}^{direction}$ and w_{ijk}^{radius} leads to the final orientation and root node of the subtree. The outgoing angle of vessels for traversing a bifurcation in the correct direction is usually lower than for traversing it in the wrong direction. This leads to a lower $w_{ijk}^{direction}$ for more feasible vessel paths. Furthermore, the radius of pulmonary vessels usually shrinks from proximal to distal regions of the lungs, which is preferred by w_{ijk}^{radius}. Therefore, the orientation of the subtree that results in the minimal weight creates the most feasible vessel paths.

3.4.4 Branching Point Extraction

In the previous step of the algorithm, anatomically meaningful subtrees have been extracted. Although the extracted subtrees are indeed tree-like structures, their branching points do not coincide with those of the actual vessel trees, but with vertices of the local maxima graph G (see Section 3.3.2). Moreover, the generated vessel paths may be irregular at their end-points. For further processing, the branching points of the actual vessel trees need to be calculated and the vessel paths need to be resampled for each extracted subtree, which is shown in Figure 3.11.

The actual branching points are generated by analyzing the tree structures of the subtrees. If a node of the tree structure has just one child, there cannot be a branching point on its path. If a node has more than one child, there must be a branching point of the actual vessel tree on the paths of the children. At the beginning, the children paths typically run along on approximately the same coordinates, while at some point they start to diverge. This is the sought after branching point of the actual vessel tree. An example of extracting a real branching point is visualized in Figure 3.11c.

The vessel segment path, i.e., the vessel path between consecutive start-, branching- and end-points of the subtree, is calculated by concatenating the paths in between them. Although the points of a vessel path between two maxima are regularly spaced, as explained in Section 3.3.3, their concatenations do not have to be. As a result of the gradient descent method, the distance between the next-to-last and the last point of a path does not need to be equal to the distance between other adjacent point pairs in the path.



(a) Subtree with vessel enhancement maxima as vertices.

(b) Zoom-in of resampled path at vessel enhancement maximum.

(c) Resampled subtree with real branching points as vertices.

Figure 3.11: The branching point extraction of a subtree (a) with the subsequent linear resampling (b) generates the actual subtree (c). For better visualization, the images show 3D (x, y, r) instead of 4D (x, y, z, r) paths.

Furthermore, the start- and end-point of a path are restricted to integer coordinates, in contrast to all the other points. These two characteristics can lead to outliers and spikes throughout concatenated paths that do not influence the segmentation itself, but the analysis of the paths (see Chapter 5). Therefore, the paths between two branching points are linearly resampled in a fixed distance to reduce these effects, as depicted in Figure 3.11b.

This leads to the final vessel subtrees s_i, with branching points of the actual vessel and paths of vessel segments with subvoxel accuracy, which is visualized in Figure 3.9.

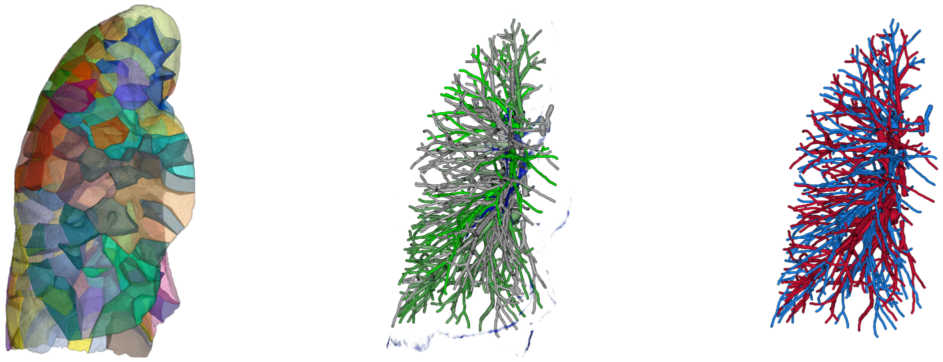
3.5 Artery/Vein Labeling

After extracting the subtrees, as described in the previous section, each of them has to be labeled as either artery or vein. The labeling problem is modeled with a second integer program, whose objective function is explained in Section 3.5.1. This integer program makes use of two anatomical properties of vessels inside the lung.

With the first property we utilize that arteries and veins are roughly uniformly distributed in the lung, i.e., there are no large regions, where just arteries and no veins, or conversely, are present. This property holds for all patients, irrespective of a possible lung disease, as blood that flows from the heart into the pulmonary arteries has to flow out of the veins back to the heart again. We calculate the measure of uniformity with the help of a Generalized Voronoi Diagram (GVD) [1]. The *GVD* determines the nearest subtree for every voxel inside the lung segmentation. By maximizing the number of voxels on the contact surface between regions labeled as arteries and regions labeled as veins, a uniform distribution of those regions is encouraged. A more detailed explanation of how we calculate this measure is given in Section 3.5.2.

The second anatomical property used here is that arteries run in parallel and in close proximity to bronchi, as previously proposed by Bülow et al. [6]. They define a measure that assigns to every vessel segment a certainty of being an artery by searching for bronchus points in close proximity to vessels. In contrast to them, we do not search for bronchus points in the input *CT* image, but create a bronchus enhancement image for this purpose. A more detailed description of this so-called ”arterialness” measure is given in Section 3.5.3.

Intermediate images for both anatomical measures, as well as an A/V segmentation calculated with the A/V labeling, are shown in Figure 3.12.



(a) The GVD. **(b)** Arterialness measure. **(c)** A/V labeling result.

Figure 3.12: The steps of the A/V labeling of subtrees. The objective function uses two weights that are based on a *GVD* shown in (a) that labels each voxel depending on its nearest subtree, and on an arterialness measure visualized in (b) that gives higher responses (green) for subtrees closer the bronchi (blue). The final result of the A/V labeling is shown in (c), where arteries are colored in blue and veins in red.

3.5.1 Objective Function

The quadratic objective function used for A/V labeling incorporates two binary variables for every previously extracted subtree s_i. The first variable, a_i, is set to 1, if and only if the subtree s_i is an artery, whereas the second variable, v_i, is set to 1, if and only if the subtree s_i is a vein. The objective function looks as follows:

$$\underset{a,v}{\arg\max} \sum_{s_i, s_j \in T} a_i\ v_j\ w_{ij}^{border} + \lambda \sum_{s_i \in T} a_i\ w_i^{artery} \tag{3.20}$$

subject to:

$$a_i + v_i = 1 \qquad \forall i\ \in\ R, \tag{3.21}$$

where $a_i, v_j \in \{0,1\}$, and $\lambda, w_{ij}^{border}, w_i^{artery} \in \mathbb{R}_0^+$.

The first term of Equation (3.20) counts the number of voxels w_{ij}^{border} on the contact surface between artery and vein regions modeled by a *GVD*. If the trees s_i and s_j have different labels ($a_i = 1$ and $v_j = 1$) and those two regions share a common border ($w_{ij}^{border} > 0$), their quantified border is added to the sum. The labeling that maximizes the number of voxels on the contact surface encourages a uniform distribution of arteries and veins throughout the lung. See Section 3.5.2 for a more detailed explanation of w_{ij}^{border}.

The second term of Equation (3.20) sums up the arterialness values w_i^{artery} for every tree s_i labeled as an artery ($a_i = 1$) to incorporate a distinction between arteries and veins. Every arterialness value of vessels labeled as arteries increases the sum, whereas arterialness values of vessels labeled as veins do not change the sum. Therefore, the labeling that maximizes the arterialness sum of arteries encourages vessels with high arterialness values to be labeled as arteries. The arterialness measure is explained in detail in Section 3.5.3.

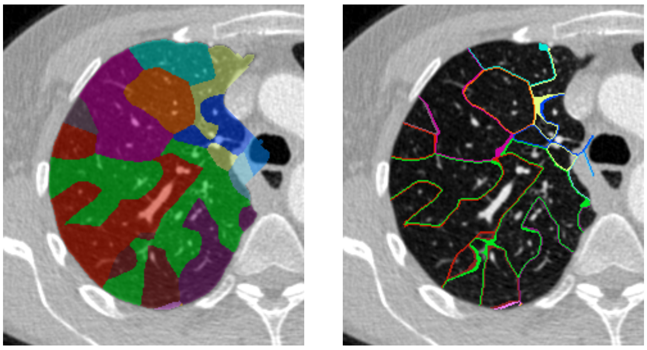
The constraint (3.21) of the objective function ensures that both labels are not active at the same time for the same tree s_i.

A critical part of getting good results with Equation (3.20) is normalization of the values. As w_{ij}^{border} and w_i^{artery} are scaled differently, they have to be normalized with their maximum value so that they range between 0 and 1. As a result, both sums have a similar range of values. The sums can further be weighted with a factor λ that allows shifting their influence in the total sum.

3.5.2 Generalized Voronoi Diagram of Subtrees

To enforce a uniform distribution of arteries and veins throughout the lung, we use a *GVD*. The *GVD* partitions a volume into regions, based on the distance to their nearest seed points. Every voxel in the same connected region has the same nearest seed points [1]. In our case, every extracted subtree acts as a seed, leading to a *GVD* where every voxel inside the lung segmentation is set to the label of its nearest subtree.

We create the *GVD* by calculating a distance transformation for every subtree, where the region inside the subtree has the distance 0. By combining all distance transformations and restricting them to the lung segmentation, the nearest structure for all voxels inside the lung can be calculated. The resulting image consists of multiple connected regions that are set to the label of their nearest subtree. The *GVD* of the subtree extraction output of Figure 3.9 is shown in Figure 3.12a. A slice of the *GVD* as well as the borders of its regions are shown in Figure 3.13.



(a) Slice of a *GVD*. **(b)** Slice of *GVD* borders.

Figure 3.13: A single slice of a *GVD* is shown in (a), the borders of its regions are shown in (b).

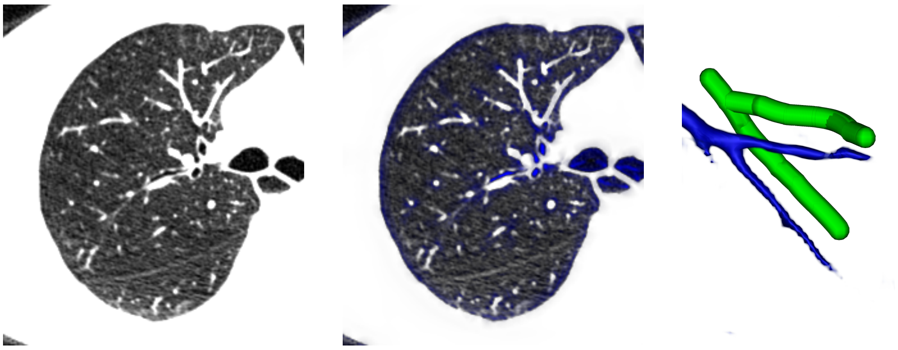
The binary labeling that maximizes the number of voxels on the contact surfaces of different regions, i.e., their common border (see Figure 3.13b), encourages a uniform distribution of both labels throughout the lungs. The number of voxels w_{ij}^{border} on the contact surface of the two Voronoi regions i and j is calculated by counting all voxels inside region i that have a voxel of region j as a direct neighbor and conversely.

It has to be noted that just by maximizing the contact surfaces, it is not clear which one of the binary labels corresponds to arteries and which one corresponds to veins. Additional values have to be incorporated to decide which label belongs to which vessel type. This part will be presented in the next section.

3.5.3 Arterialness Measure of Vessel Segments

We propose a measure that gives a high value for arteries, the so-called ”arterialness”, that is based on the measure described by Bülow et al. [6]. It uses the anatomical property that arteries, in contrast to veins, are usually in close proximity and run in parallel to bronchi. As automatic airway segmentations are usually not very deep and therefore of limited use in the range of interest, Bülow et al. propose to analyze planes perpendicular to vessels and search for probable bronchus points inside them. The steps needed for calculating the arterialness measure are shown in Figure 3.14.

For our arterialness measure, we require a bronchus enhanced image (see Figure 3.14b) and the estimated bronchus directions. These are calculated with the same filter that was used for vessel enhancement (see Section 3.3.1), but configured to enhance dark-on-bright instead of bright-on-dark structures. In contrast to the previous vessel enhancement step, we additionally extract the estimated bronchus direction. The input *CT* images are beforehand clamped to have a maximal value of -400 Hounsfield Unit (HU) to further



(a) CT slice clamped below -400 HU.

(b) Bronchus enhancement slice.

(c) Close artery and bronchus.

Figure 3.14: The steps of calculating the arterialness measure. A slice of the preprocessed input image, which clamps the *CT* image below -400 HU and makes the airway more prominent, is shown in (a). The result of the subsequent bronchus enhancement for this slice is shown in (b), where the bronchus response is overlayed in blue. A 3D rendering of two artery segments (green) and a nearby bronchus (blue) is shown in (c).

enhance the bronchus regions and suppress responses outside the *HU* range of interest (see Figure 3.14a) .

The arterialness measure is calculated independently for each vessel segment, i.e., the part of the vessel between branching- or end-points. At each voxel along a segment, we search for similarly oriented bronchial structures giving high tubularity response in a plane orthogonal to the vessel direction (see Figure 3.14c). Every point inside this plane that has a value below a certain threshold in the input *CT* image, a high value in the bronchus enhanced image, as well as a similar bronchus direction as the vessel direction, is added to a list of possible bronchus points. The bronchus near a vessel segment is a roughly straight line. To connect the extracted probable bronchus points of a whole vessel segment, a line l is fitted to all of them. The line fitting algorithm needs to be robust to outliers, as some non-bronchus points may be in the list of probable bronchus points as well, i.e., points adjacent to a vessel, which usually get high tubularity values and have directions similar to the vessel. Therefore, the applied line fitting algorithm uses a combination of Random Sample Consensus (RANSAC) and iteratively reweighted least squares to be robust against outliers. In case too few bronchus points are found in close proximity to l, which implies that the fitted line does not represent a bronchus branch, we set the arterialness measure to 0. Otherwise, the final arterialness value is the mean of the dot products of the vessel direction and the direction of the fitted line divided by their point distance. It has higher values for arteries running closer and in parallel to bronchi, while veins, typically more distant and deviating stronger from the bronchus direction, will receive lower values. The arterialness value of a vessel path p is calculated with the following formula:

$$arterialness(p) = \sum_{i=1}^{n-1} \frac{c(\mathbf{u}_i, \mathbf{v})}{d(\mathbf{p}_i, l)}, \tag{3.22}$$

with $c(\mathbf{u}_i, \mathbf{v}) = \max\left(\mathbf{u}_i \cdot \mathbf{v}, (-\mathbf{u}_i) \cdot \mathbf{v}\right)$ being the undirected coorientation of the vectors $\mathbf{u}_i$ and $\mathbf{v}$, and $d(\mathbf{p}_i, l)$ being the minimal Euclidean distance between path point $\mathbf{p}_i$ and line l. The series $\mathbf{u}_i$ is composed of $n-1$ normalized direction vectors between the n points of path p, whereas $\mathbf{v}$ is the normalized direction vector of line l.

The arterialness weight w_i^{artery} of a tree s_i is the sum of all arterialness values of its vessel segments.

3.6 Finalization of Artery/Vein Separation

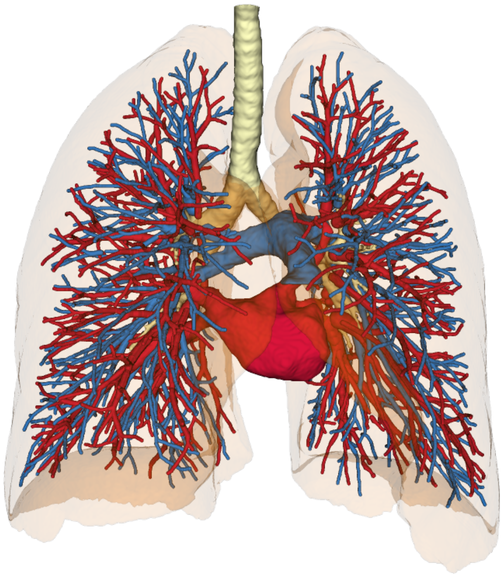


Figure 3.15: Rendering of the final A/V paths inside the lung segmentation with additional segmentations of airways, pulmonary artery and left atrium. Arteries are colored blue, whereas veins are colored red.

As the previous steps of the algorithm have been calculated for both sides of the lung independently, the resulting lists of subtrees for both lungs need to be merged. This leads to a list of arterial and a list of venous subtrees that consist of 4D vessel paths. These 4D paths can be rendered or used for further analysis, which will be explained in the next chapters.

Figure 3.15 shows an example rendering of the extracted arterial and venous 4D vessel paths inside the lung segmentation with additional segmentations of airways, pulmonary artery and left atrium.

4

Segmentation Evaluation

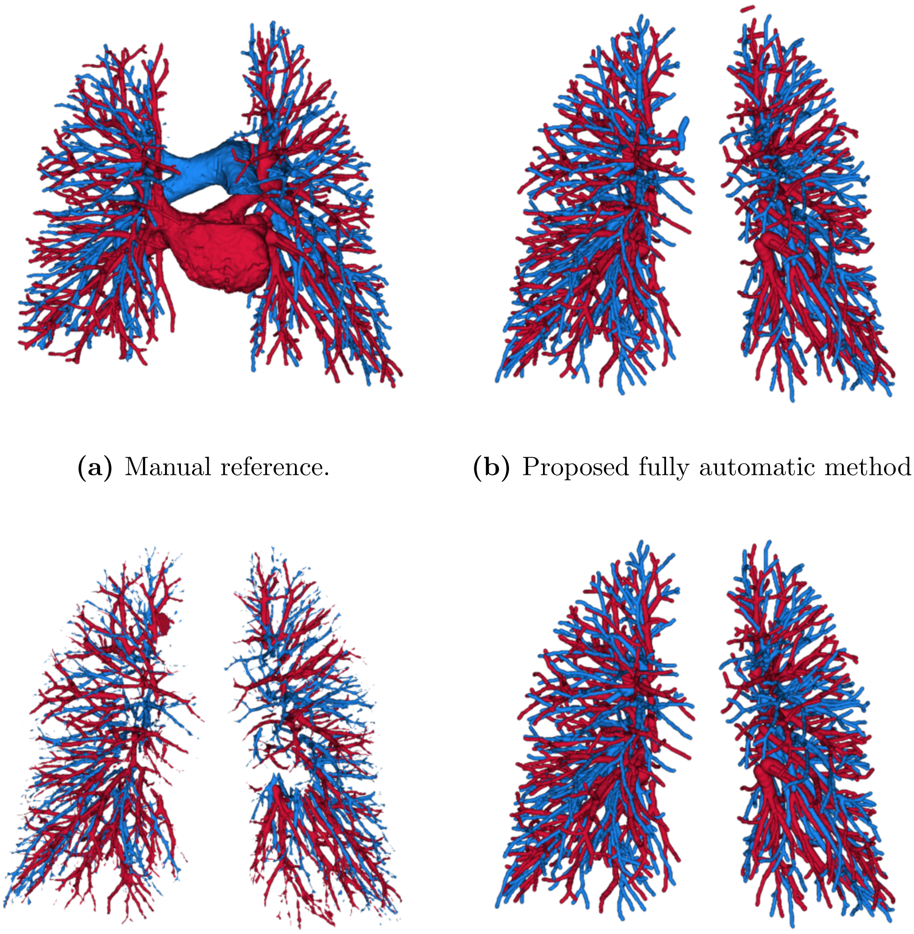
Contents

Validation of the algorithm was performed on a dataset consisting of 10 thoracic contrast-enhanced Dual-Energy Computed Tomography (DECT) images (6 female/4 male) from the LBI-LVR-Clinical study dataset. The Computed Tomography (CT) scans were either acquired with a Siemens Somatom Definition Flash (D30f reconstruction kernel) or a Siemens Somatom Force (Qr40d reconstruction kernel) CT scanner. The mean size of the *CT* volumes was 512 × 512 × 463 pixels with an almost isotropic voxel size of approximately 0.6 mm per dimension. For these 10 volumes, a manual reference segmentation of arteries and veins was created. The proposed algorithm as well as a reimplementation of a current state-of-the-art method of Park et al. [34] were compared to this manual reference segmentation. Additionally, the proposed algorithm was compared to a user-defined labeling of the subtrees, where all extracted subtrees were labeled manually as arteries, veins or non-vessels.

A description of the methods and their implementation are given in Section 4.1, whereas the segmentation results are shown in Section 4.2 and discussed in Section 4.3.

4.1 Implementation Details

For each dataset, A/V segmentations were created with four different A/V separation methods. The manual reference segmentation was used as a ground-truth and is described in Section 4.1.1. The proposed algorithm and details of the parameters used for obtaining the results are shown in Section 4.1.2. The reimplementation of the method of Park et al.



(a) Manual reference.

(b) Proposed fully automatic method.

(c) Method of Park et al.

(d) User-defined labeling.

Figure 4.1: Separated vessel tree renderings of patient #3 for all four segmentation methods. The images show the manual reference (a), the fully automatic segmentation (b), the result of the reimplementation of Park et al. (c) and the user-defined labeling (d). Arteries are colored in blue, veins are colored in red.

[34] is described in Section 4.1.3, the user-defined labeling in Section 4.1.4. Examples of all segmentations for the same dataset are shown in Figure 4.1.

The development and testing platform consisted of an Intel Core i5-4670 @ 3.40 GHz with 16 GB RAM running Windows 7. Development was done in C++ using ITK[1], with minor parts written in CUDA[2]. Visualizations were created with VTK[3] and ITK-SNAP[4].

[1] ITK version 4.3.2 from `http://www.itk.org`

[2] CUDA toolkit version 6.5 from `https://developer.nvidia.com/cuda-toolkit`

[3] VTK version 5.10.1 from `http://www.vtk.org`

[4] ITK-SNAP version 3.2.0 from `http://www.itksnap.org`

The time needed to generate the segmentations on the testing PC is shown in Section 4.1.5.

4.1.1 Manual Reference

The manual reference segmentation was created using ITK-SNAP. It includes the main pulmonary artery and the left atrium and covers pulmonary vessels with a diameter down to about 2 mm. To reduce image noise and simplify segmentation, the input volume was denoised with the edge-preserving total variation denoising algorithm from Rudin, Osher, and Fatemi [38]. Most of the voxels were marked with ITK-SNAP's adaptive brush in the denoised image. This adaptive brush creates segmentations that stop at the largest gradient, and therefore cover whole vessels from their centerline to their borders.

First, the pulmonary artery and the left atrium were segmented. The vessels inside the lungs were segmented by labeling larger vessels starting from the pulmonary artery or the left atrium and letting these vessel trees grow towards finer vessels. Especially in the proximal regions of the lung, borders of vessels are often hardly visible. Therefore, already set labels sometimes had to be adapted throughout the process of creating the segmentation.

4.1.2 Proposed Fully Automatic Method

The fully automatic segmentation was generated with the method described in Chapter 3. Most of the implementation was done in C++ using ITK, with the distance transformation used for generating the Generalized Voronoi Diagram (GVD) being the only exception that was implemented in CUDA. The integer programs were solved using the general purpose Gurobi Optimizer[5]. This section lists the parameter values used to obtain the results. Most of the parameters were determined empirically, while the parameters of the objective function of the A/V labeling integer program were obtained by grid search, which is an exhaustive search for best parameters from a predefined subset.

The implementation of the Optimally-Oriented Flux (OOF) vessel enhancement filter was taken from [3]. Its radius range was set from 0.5 mm to 5.0 mm in 0.5 mm intervals. The σ of the Gaussian function $G_\sigma(\cdot)$ used for smoothing the input of the vessel enhancement was set to 0.5 mm.

For generating the local maxima graph G, the minimal tubularity value of points was set to 200. This led to a relatively low number of extracted points, but was necessary for runtime reasons. The minimal distance between local maxima (d_{maxima}) was set to 3 mm, the maximal distance of connected neighbors ($d_{neighbor}$) was set to 15 mm.

The implementation of the 4D path extraction was taken from [3], with the step size of the gradient descent algorithm set to 0.3 mm. For removing overlapping paths, a path was considered completely inside another one, if all of its points were at most 1 mm distant from the other one.

[5] Gurobi Optimizer version 6.0 with academic license from `http://www.gurobi.com/`

The subtree extraction integer program was solved with the Gurobi Optimizer. The edge-pair weights w_{ijk} had the following weighting factors $\alpha = 1$, $\beta = 0.5$, $\gamma = 1$, and $\delta = -0.2$, which provided a good result and generated reasonable subtrees. The root penalizing factor σ was set 0.2.

As the finding of the optimal solution of the objective function took between 5 and 50 hours, some interventions were used to reduce runtime. By analyzing the output of the optimizer, we observed that the optimal solution had been found within one hour in all datasets. The optimizer needed the additional runtime to confirm that this solution was the optimal one (see Section 3.1.2). Furthermore, we observed that the optimizer found improved intermediate results every one to two minutes until reaching the optimal one. Therefore, we took the current best intermediate result of the optimizer, if its last improvement had been more than ten minutes ago. This way, the runtime reduced drastically, with the downside that the results were not guaranteed anymore to be optimal.

The subsequent branching point extraction and path resampling used the following parameters: A branching point was detected when two concurrent paths were deviating by more than 0.1 mm. The resulting vessel segments were linearly resampled with a distance of 0.5 mm.

The subtree labeling integer program was solved with the Gurobi Optimizer as well. The parameters for the objective function were obtained by grid search using the manual segmentations as reference, as these parameters are very hard to estimate and slight changes result in large differences of the final A/V segmentation. The border weights w_{ij}^{border} were calculated straightforward without the need of any parameters, as opposed to the artery weights w_i^{artery}. Arterialness values were calculated for vessel segments with a mean radius between 1.5 mm and 4.5 mm. The bronchus-enhanced images were calculated with the *OOF* and the parameters listed before, but configured for enhancing dark-on-bright instead of bright-on-dark structures. A point near the vessel segment was considered a bronchus candidate, if its attenuation was less than -700 HU, its bronchus-enhanced value was greater than 100, and the dot product of its estimated bronchus direction and the vessel direction was greater than 0.7. Afterwards, the line through all bronchus candidates was calculated with a line fitting algorithm taken from OpenCV[6] using the Huber distance. This line was then used for calculating the final arterialness values. Additionally, if less than 70% of all bronchus candidates were closer than 2 mm to the fitted line, the arterialness value was set to 0. The weighting factor λ of the objective function was set to 6.

The final voxel-based A/V segmentation was generated by setting all voxels inside the spheres traversing the 4D arterial and venous vessel paths accordingly.

[6] OpenCV version 3.0 beta from `http://www.opencv.org/`

4.1.3 Reimplementation of the Method of Park et al.

As the method of Park et al. [34] needs a full vessel segmentation as input, a simple vessel segmentation had to be performed beforehand. The vessel segmentation was obtained by binarizing the same vessel-enhanced image that was used for the proposed method, with a threshold of 150. Afterwards, the shortest path trees from the central point of the input were calculated for both lungs separately. As Park et al. proposed, each of those trees were cut manually to remove the critical end erroneous central regions of the lungs. The regions for the manual cuts were created using a segmentation in ITK-SNAP, where the border of the segmentation served as the cut in the tree.

In order to automatically group vessel subtrees that belong together, Park et al. proposed to generate cylinders at the root of each subtree and to calculate their mutual overlap. Vessel trees, whose cylinders are overlapping by a certain value, are merged into the same group until no more merging is possible. The remaining vessel groups have to be labeled manually as either arteries or veins.

We decided to label each subtree manually without trying to merge them automatically. This way, we needed more user interaction time, but removed possible errors of the automatic merge. The decision of how to label a subtree was made by looking at its connection to the heart using ITK-SNAP. Subtrees that are composed of vessels of both types or other structures, e.g., because of wrong bifurcations, were set to its predominant vessel type.

4.1.4 User-defined Labeling

This segmentation was created in order to evaluate the automatic A/V labeling part of the proposed algorithm, as well as to generate quantitative readouts for medical application (see Chapter 5) that are not distorted by segmentation errors. This segmentation was created by intervening in the proposed algorithm after the subtree extraction step of Section 3.4 and performing the A/V labeling manually, instead of using the automatic labeling of Section 3.5.

Every subtree was labeled as either an artery, a vein or a non-vessel. The manual labeling was performed similar to the reimplementation of Park et al., as described in the previous subsection, but with a small change in the exclusion of merged vessels. Subtrees that are composed of mostly one vessel type and just little parts of the other vessel type or other structures, were set to its predominant vessel type, like before. However, subtrees, where arteries, veins or other structures are merged extensively were labeled as non-vessels. The reason for this change is, that we created the user-defined labeling for the calculation of quantitative readouts and to measure changes in arteries and veins. Allowing extensively merged vessels could bias the readouts for the different vessel types.

4.1.5 Calculation Times

The times for creating the segmentations for the different methods are displayed in Table 4.1. The total time needed is divided into automatic time, i.e., the calculation time of the computer, and interactive time, i.e., the additional time the user spent for obtaining the results.

	Manual reference	Proposed fully automatic	Reimplementation of Park et al.	User-defined labeling
Automatic [h]	-	5	0.5	4.5
Interactive [h]	5	-	2	2
Total [h]	5	5	2.5	6.5

Table 4.1: Approximated average time needed for an A/V segmentation for each method.

4.2 Segmentation Results

In this section the three methods described in the previous section, i.e., the fully automatic, the reimplementation of [34], and the user-defined labeling, are compared to the manual reference segmentation (see Section 4.2.1). Additionally, the proposed automatic segmentation is compared to the segmentation of the user-defined labeling in order to quantify misclassified and non-vessel regions (see Section 4.2.2).

4.2.1 Comparison with Manual Reference Segmentation

To evaluate the different methods, the agreement of the A/V segmentations and the manual reference segmentation were calculated. The calculated segmentations may contain vessels which were correctly segmented and labeled, but are not inside the manual reference, and conversely. Therefore, the comparison with the manual reference is restricted to voxels which are present in both the manual and the calculated segmentation, i.e., intersecting voxels of both segmentations.

We calculated the agreement, i.e., the percentages of identically labeled voxels per total number of intersecting voxels, with the following formula inspired by Dice [8]:

$$\text{agreement} = \frac{|A_{ref} \cap A_{test}| + |V_{ref} \cap V_{test}|}{|(A_{ref} \cup V_{ref}) \cap (A_{test} \cup V_{test})|}, \tag{4.1}$$

where A_{ref} and V_{ref} are the sets of arteries and veins of the manual reference segmentation, respectively, and A_{test} and V_{test} are the ones of the segmentation to test.

The calculated agreement values for each dataset are visualized in Figure 4.2. The mean agreement value for our proposed fully automatic segmentation is 94.1% with a range of 85% to 98.7%. The mean agreement for the method of Park et al. is 91.9% (88%

– 95.3%), whereas it is 98.8% (97.6% – 99.8%) for the user-defined labeling. All fully automatic A/V segmentations and their overlap with the manual reference segmentation are visualized in Figure 4.3 and 4.4.

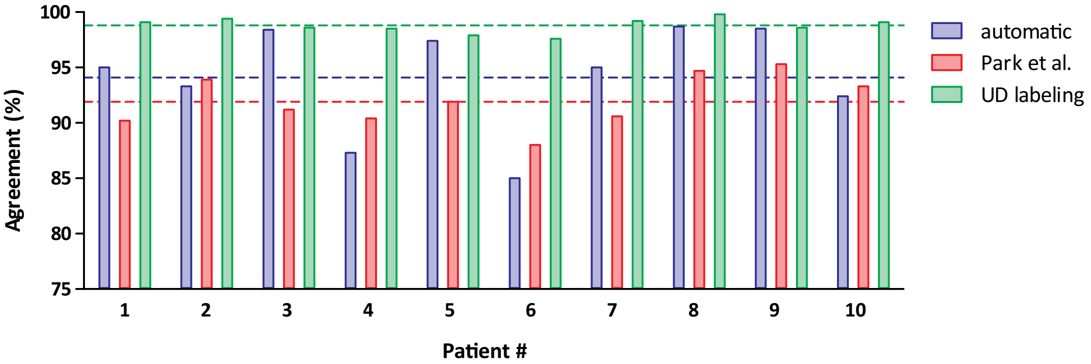


Figure 4.2: Identically labeled voxels per total number of intersecting voxels for each dataset. The results of the proposed method are visualized in blue, the results of the method from Park et al. are visualized in red, and the results of the user-defined labeling are visualized in green. The means are visualized as dashed lines. Values are given in % and are displayed between 75% and 100%.

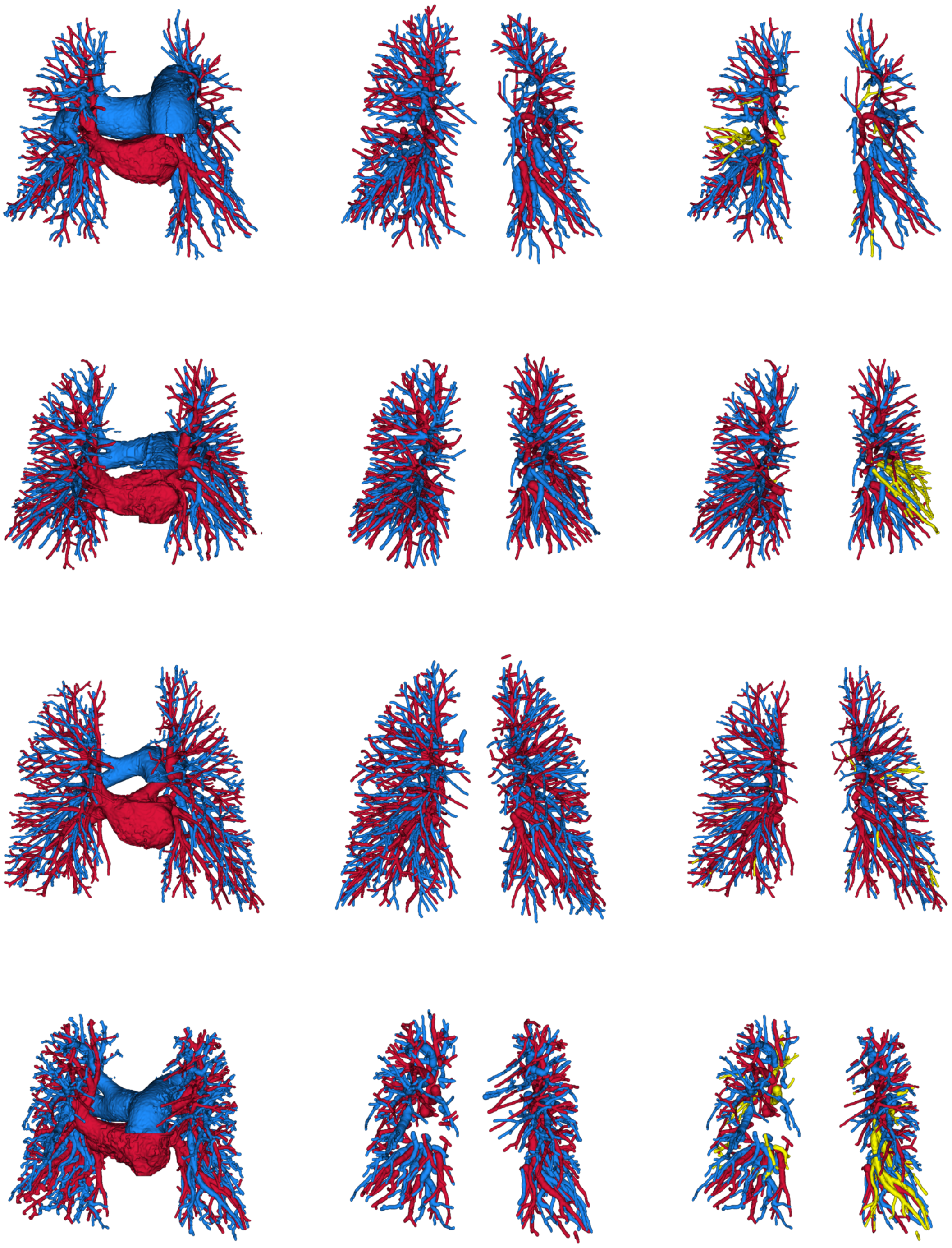


Figure 4.3: Separated vessel tree renderings for the first four patients. The first column shows the manual reference segmentation, the second column shows the result of the proposed fully automatic method, and the last column shows the overlap of automatic and reference segmentations, where disagreeing voxels are colored yellow, arteries are colored blue, and veins are colored red.

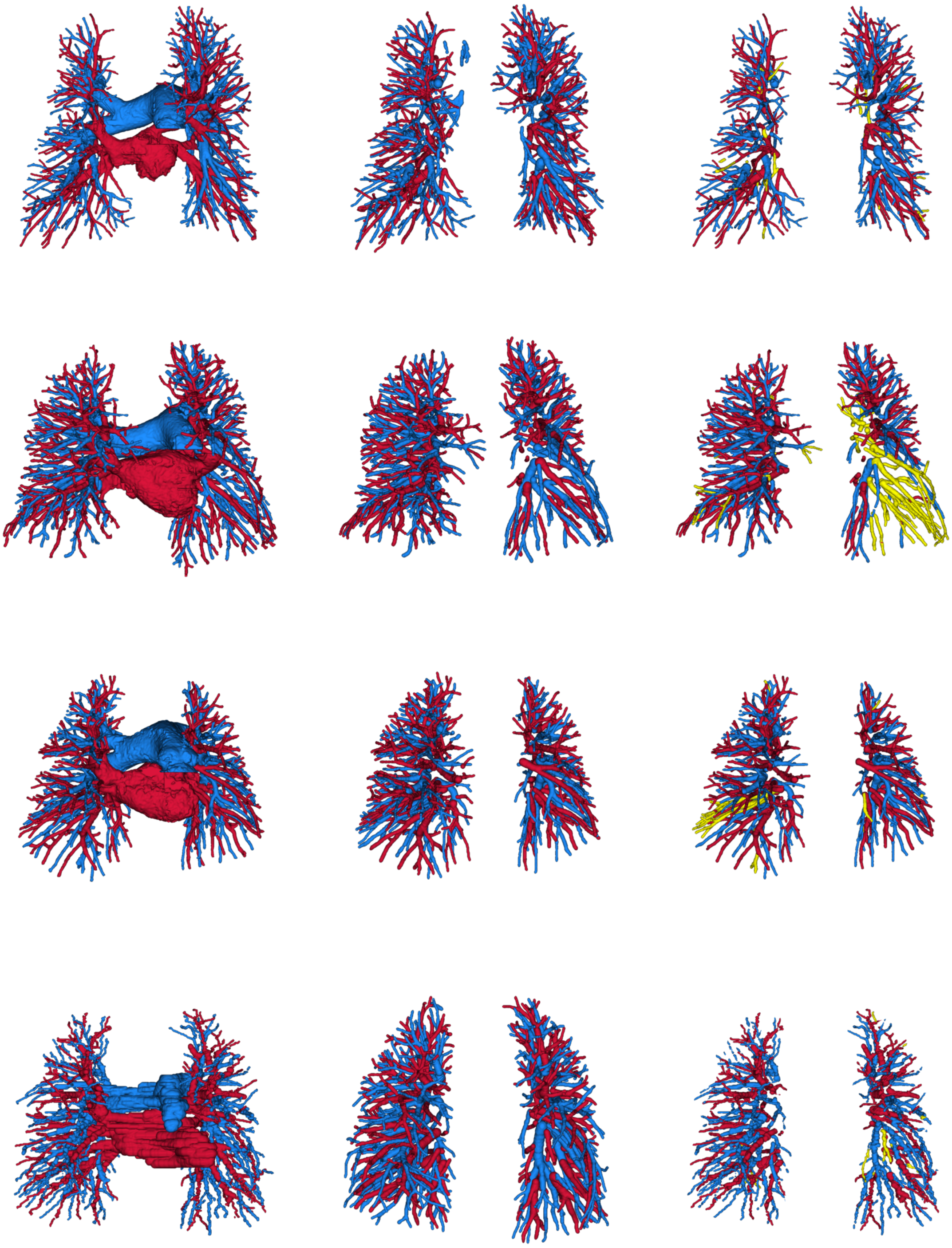


Figure 4.4: Separated vessel tree renderings for the second four patients. The visualization is the same as in Figure 4.3.

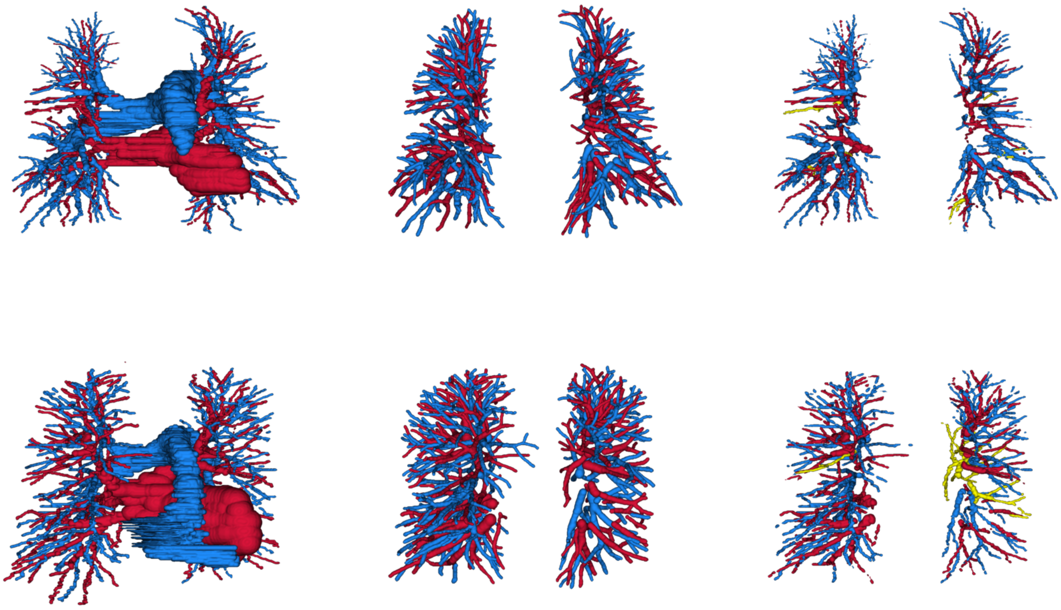
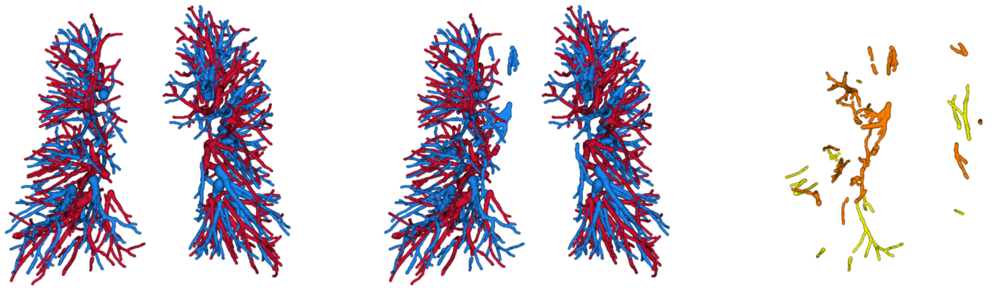


Figure 4.5: Separated vessel tree renderings for the last two patients. The visualization is the same as in Figure 4.3.

4.2.2 Comparison with User-defined Labeling

As the user-defined labeling does not produce new voxels, it is a subset of the fully automatic segmentation. Therefore, we were able to quantify the number of structures misclassified as being vessels, i.e., the voxels that are set in the automatic segmentation, but not in the user-defined labeling. Additionally, we were able to quantify the number of misclassified vessels, i.e., the switched arteries and veins by comparing the labels of the vessel voxels. An example of the mislabeled and non-vessel voxels for one patient is shown in Figure 4.6.



(a) User-defined labeling. **(b)** Fully automatic method. **(c)** Disagreement.

Figure 4.6: Disagreement between user-defined labeling (a) and fully automatic method (b) for patient # 5. The last image (c) shows the mislabeled vessels in yellow and the non-vessel structures in orange.

The formulas used for calculating the mislabeled and non-vessel values are as follows:

$$\text{mislabeled} = \frac{|A_{ud} \cap V_{auto}| + |A_{auto} \cap V_{ud}|}{|A_{auto} \cup V_{auto}|}, \tag{4.2}$$

$$\text{non-vessel} = \frac{|(A_{auto} \cup V_{auto}) \setminus (A_{ud} \cup V_{ud})|}{|A_{auto} \cup V_{auto}|}, \tag{4.3}$$

where A_{ud} and V_{ud} are the sets of arteries and veins of the user-defined segmentation, respectively, and A_{auto} and V_{auto} are the ones of the fully automatic segmentation.

The calculated mislabeled and non-vessel percentages of the 10 datasets are visualized in Figure 4.7. The mean value of mislabeled vessels is 4.9%, ranging from 0.6% to 11.6%, whereas the mean value of non-vessel structures is 1.7%, ranging from 0.0% to 5.1%.

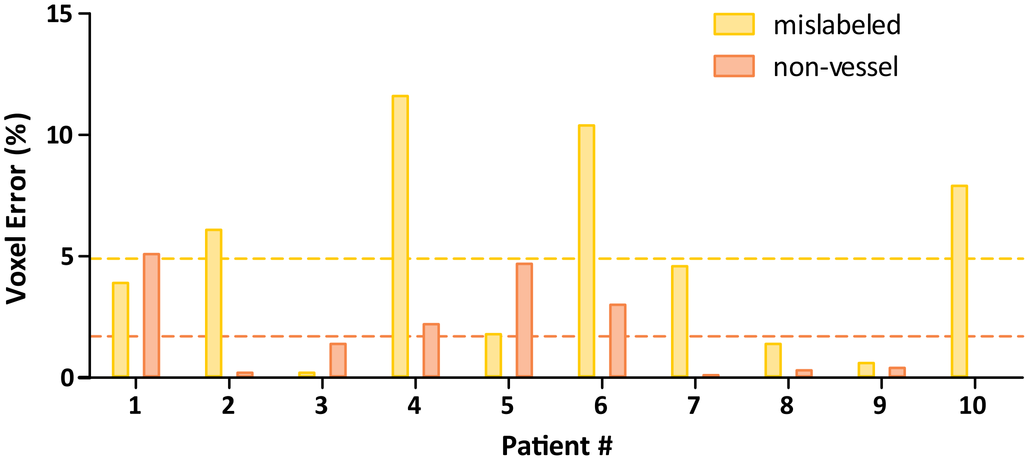


Figure 4.7: Mislabeled vessels and non-vessel structures per total number of voxels for each dataset. The mislabeled percentage is colored yellow and the non-vessel percentage is colored orange. The mean of both values is shown as a dashed line. Values are given in % and are displayed between 0% and 15%.

4.3 Discussion

We showed that our proposed fully automatic algorithm achieves a mean voxel-based agreement with a manual reference segmentation of 94.1% on our dataset of 10 patients, outperforming a reimplementation of a recent state-of-the-art interactive algorithm [34]. Although manual postprocessing further increased the mean agreement to 98.8%, we see the main benefit of our proposed algorithm in not needing user interaction at all, as compared to other A/V separation methods (see Chapter 2). Neither manual preprocessing, like creating training samples for machine learning methods [20], nor interaction during runtime, like defining seed points [40], nor manual postprocessing, like labeling of subtrees [34], is necessary. Additionally, the algorithm does not need to know the roots of the pulmonary arterial and venous vessel trees in the heart. This could be used for A/V separation in non-contrast-enhanced *CT* images, where the pulmonary vessels are hardly distinguishable from their surrounding tissue in the heart. Furthermore, the four-dimensional path representation allows straightforward vessel segmentation and structural analysis which could aid in diagnosing vascular diseases.

The main cause of errors in the agreement with the manual reference is the A/V labeling part of the algorithm. One reason for this could be the unstable arterialness measure. Sometimes veins get high arterialness values, which happens when they are close and in parallel to bronchi or simply because of noise. A single vein with higher arterialness measure than the surrounding arteries combined can lead to a large region of misclassified subtrees, as can be observed in Figure 4.3, 4.4 and 4.5. The reason for this is that we try to maximize the uniform distribution of arteries and veins. By labeling this vein mistakenly as an artery, this error propagates in a large region until a real artery with a high arterialness measure is found. Furthermore, the arterialness measure can only be calculated for relatively large vessels, as bronchi are not visible in *CT* images in more distal parts of the lungs. Unfortunately, in more proximal regions, arteries, veins and bronchi are often close to each other, which could further distort the arterialness measure. We suggest that using a deep-reaching airway segmentation instead of searching for probable bronchus points could improve the arterialness calculation.

During setting up the algorithm, we observed that the measure of uniformity of arteries and veins throughout the lungs alone often delivers better separation results than its combination with the arterialness. The main problem with this approach is that this results frequently in a complete switch of labels in a large region. By analyzing these results further, we observed that these complete switches occur most of the time in a whole lung lobe. Therefore, we assume that our measure of uniform distribution of arteries and veins does not hold throughout the whole lung, but just throughout individual lung lobes, which would also be in better concordance with the anatomy.

As a consequence, our algorithm could benefit from a proper lung lobe segmentation in order to process each lobe independently, instead of each side of the lung, which is the case in the current state of the algorithm. Furthermore, by splitting up the computation into lung lobes, the runtime could greatly decrease, as it would reduce the input complexity for the individual steps of the algorithm.

On the other hand, some parts of the proposed algorithm could be used for obtaining a lung lobe segmentation. As vessels do not cross the borders of a lung lobe, the branches of the extracted subtrees stop just before lung lobe borders. This leads to a *GVD*, where some region borders coincide with the lung lobe borders. Another integer program could be used for calculating a labeling for those regions by enforcing that the contact surface between regions with different labels is almost plane-like, which is the case for lung lobes.

As the A/V labeling part of the algorithm is unable to recover from errors of the subtree extraction part, combining their integer programs into one could improve the results. Robben et al. [37] showed that such simultaneous segmentation and labeling is possible with just one objective function.

Although most of the errors are caused by the A/V labeling part, there are still some possibilities for improvement in the subtree extraction part. For runtime reasons, we set the threshold for extracting maxima points inside the vessel-enhanced image quite high, which leads to a relatively low number of extracted vessels. By reducing this threshold, the number of extracted vessels would increase, but the total running time would increase as well. Although we would not see a great benefit in the overlap values, as the manual segmentation is not as detailed as the automatic segmentation even with the current threshold, the value of the quantitative readouts (see Chapter 5) could improve. It is assumed that the small vessels contain substantial information for various disease like Pulmonary Hypertension (PH), which we do not cover in the current state of the algorithm.

Furthermore, non-vessel structures like airway walls or other tissue give high response in the vessel-enhanced images. Therefore, these structures may be mistakenly merged with vessel subtrees. There are also some non-vessels structures detected at the outer border of the lung segmentation, as the vessel enhancement measure gives high values for these borders as well (see Section 4.2.2). By reducing their value in the vessel-enhanced images, e.g., by dilating the bronchi or by eroding the lung segmentation, the erroneous extraction of these structures could be reduced.

A drawback of the algorithm is the large number of parameters (see Section 4.1.2). Nevertheless, most parameters were easy to obtain and were set to their current values for runtime reasons. The only parameters that were hard to estimate were the ones used by the A/V subtree labeling. We performed a grid search, using the manual segmentations as a reference in order to obtain the best parameters, and observed that the most sensitive ones were used for calculating the arterialness measure. We did not identify a significant structure for them, because slight changes resulted in large differences in the final A/V separation. As the parameters were optimized for the 10 available manual segmentations, they might not perform as well on other *CT* images.

Finally, although the proposed algorithm achieves very good results, the performance of results and runtime could further be improved in order to use it in clinical practice. A graphical user interface for manual corrections would also increase clinical applicability.

5 Quantitative Readouts and Clinical Application

Contents

To address the clinical application of A/V separation, various quantitative readouts that describe the structure of the arterial and venous trees were calculated with the results of the user-defined labeling (see Section 4.1.4). These results were correlated with clinical data of the patients in order to distinguish between patients with and without Pulmonary Hypertension (PH). The readouts are described in Section 5.1, whereas the results for the 10 datasets are shown in Section 5.2. Finally, results are discussed in Section 5.3.

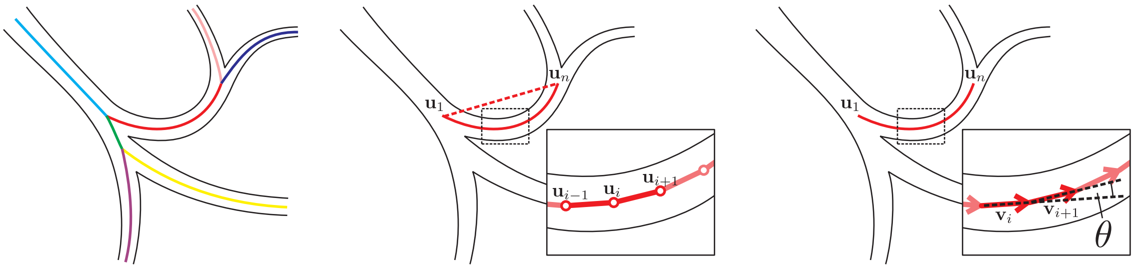
5.1 Definition of Readouts

The extracted arterial and venous subtrees allow calculation of various quantitative readouts ranging from straightforward vessel segment counting to more sophisticated tortuosity measures.

We calculated the length, mean radius, and tortuosity measures *DM* and *SOAM* for each vessel segment (see Figure 5.1a). Both assign higher values to bent vessels than to straight ones.

5.1.1 Distance Metric

The *DM* is the first value of tortuosity that we used as a readout after Bullitt et al. [5]. It is calculated by dividing the length of a vessel segment by its end-point distance, as



(a) Colored segments.

(b) Zoom-in of *DM*.

(c) Zoom-in of *SOAM*.

Figure 5.1: Quantitative readouts are calculated with vessel segments as depicted in (a), where each vessel segment is colored differently. The *DM* calculation for a vessel segment is visualized in (b), the *SOAM* calculation in (c).

illustrated in Figure 5.1b, and is defined as

$$DM(p) = \frac{\sum_{i=1}^{n-1} |\mathbf{u}_i - \mathbf{u}_{i+1}|}{|\mathbf{u}_1 - \mathbf{u}_n|}, \tag{5.1}$$

where p is the path of a vessel segment that is composed of a series of n points $\mathbf{u}_1$ to $\mathbf{u}_n$.

Totally straight vessels will give a *DM* value of exactly 1, as their end-point distance is the same as their length, whereas more bent vessels have higher *DM* values. A disadvantage of the *DM* is that it cannot distinguish between different kinds of bending, e.g., corkscrew-, S-, and C-shapes.

5.1.2 Sum of Angles Metric

The second metric for quantifying the tortuosity was the *SOAM*. It has higher values for corkscrew- or S-shaped vessels than for C-shaped vessels. Our *SOAM* calculation is similar to the one in Bullitt et al. [5], but without using the torsional angle. The *SOAM* is calculated by summing up the angles between subsequent directional vectors of a vessel segment, as illustrated in Figure 5.1b. It is defined as

$$SOAM(p) = \frac{\sum_{i=1}^{n-2} \cos^{-1}(\mathbf{v}_i \cdot \mathbf{v}_{i+1})}{\sum_{i=1}^{n-1} |\mathbf{u}_i - \mathbf{u}_{i+1}|}, \tag{5.2}$$

with

$$\mathbf{v}_i = \frac{\mathbf{u}_{i+1} - \mathbf{u}_i}{|\mathbf{u}_{i+1} - \mathbf{u}_i|}, \tag{5.3}$$

where p is the path of a vessel segment that is composed of a series of n points $\mathbf{u}_1$ to $\mathbf{u}_n$ with $n-1$ normalized direction vectors $\mathbf{v}_1$ to $\mathbf{v}_{n-1}$.

The *SOAM* returns the mean in-plane angle θ along the direction vectors of subsequent points in the vessel path. Totally straight vessels have a *SOAM* value of exactly 0, as every direction vector of the path is the same, whereas more bent vessels have higher *SOAM* values.

We used only the in-plane angle and not the torsional angle, as compared to Bullitt et al. [5], because we observed high variations of the torsional angle in almost straight paths, which leads to high *SOAM* values for them. Therefore, our *SOAM* is not increased in vessels with corkscrew-shape as compared to vessels with S-shape, but it is still increased in S-shaped vessels as compared to C-shaped ones.

5.2 Clinical Data Analysis

As we wanted to show the potential of A/V separation in a clinical setting, we needed the best segmentations available. Therefore, we used the results of the user-defined labeling, as it contains the least errors (see Section 4.2.1), which could distort the readouts. The readouts for vessel segments and the individual patients are shown in Section 5.2.1 with their correlation to *PH* in Section 5.2.2.

5.2.1 Segment-Based Readouts

Various segment-based readouts were calculated in order to describe the arterial and venous trees. We calculated the length, mean radius, *DM* and *SOAM* for each vessel segment and evaluated these values separately for all artery and vein segments as well as their combination. Differences between groups were tested with the Mann–Whitney test, where $p \geq 0.05$ was considered significant.

The mean total number of extracted vessel segments per arterial and venous tree is 609.3, ranging from 460 to 905 and 570.4, ranging from 416 to 968, respectively, which is visualized in Figure 5.2.

The mean length of vessel segments is 13.73 mm, with a range of 13.07 – 14.57, whereas the mean length for arterial segments is 13.79 mm (13.05 – 14.34) and for venous segments 13.77 mm (12.36 – 16.32), respectively. We observed significantly longer arteries for four patients, significantly longer veins for three patients, while there was no difference in length for arteries and veins for the remaining three patients. Figure 5.3 shows box plots of the length of arteries and veins separated as well as of all vessel segments combined for the individual patients.

The mean radius for arteries is 1.490 mm (1.244 – 1.754) and for veins 1.512 mm (1.363 – 1.687). For all segments combined it is 1.502 mm (1.325 – 1.683). We found no difference in radius for six patients, significantly increased radius of the veins for three patients, whereas there was one patient with significantly increased radius of the arteries (see Figure 5.4).

The first tortuosity readout is the *DM*. The mean *DM* of arteries is 1.027 (1.020 – 1.048), whereas for veins it is 1.023 (1.019 – 1.029). If they are combined, it is 1.025

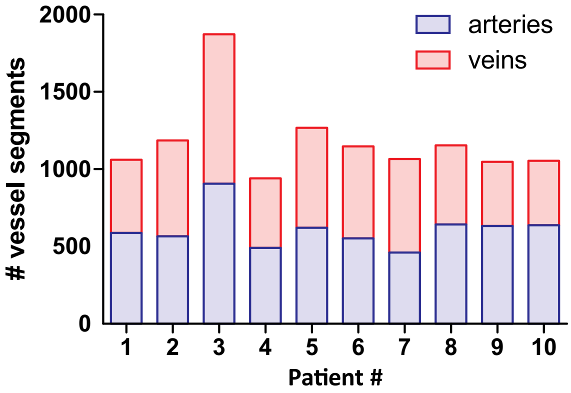
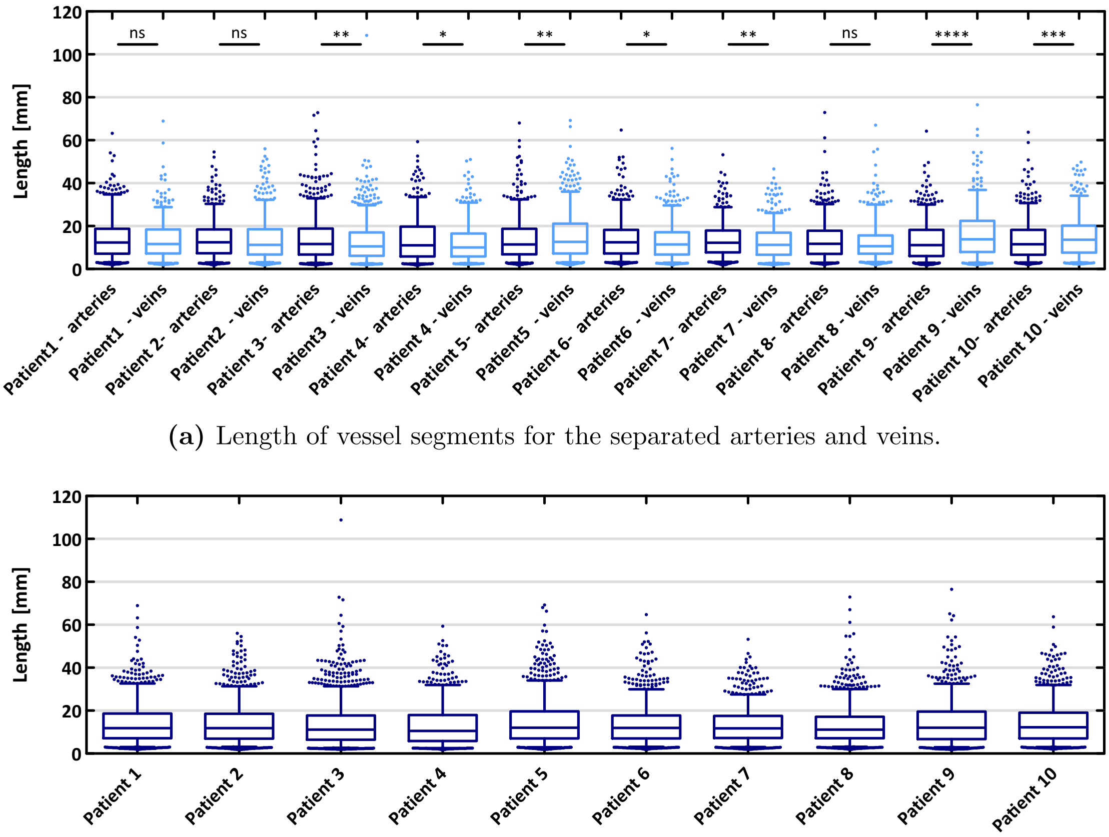


Figure 5.2: The total number of vessel segments generated for each patient. The number of arteries is visualized in blue, while the number of veins is visualized in red.

(1.021 – 1.037). We observed a significantly increase in the *DM* in arteries for four patients, whereas all the others have no significant differences (see Figure 5.5).

For *SOAM*, the second tortuosity readout, we found a mean value of 0.178 (0.171 – 0.193) for arteries, 0.171 (0.154 – 0.181) for veins and 0.175 (0.164 – 0.187) for their combination. For five patients we observed a significant increase in the *SOAM* of arteries, whereas there was no significant difference for the others (see Figure 5.6).

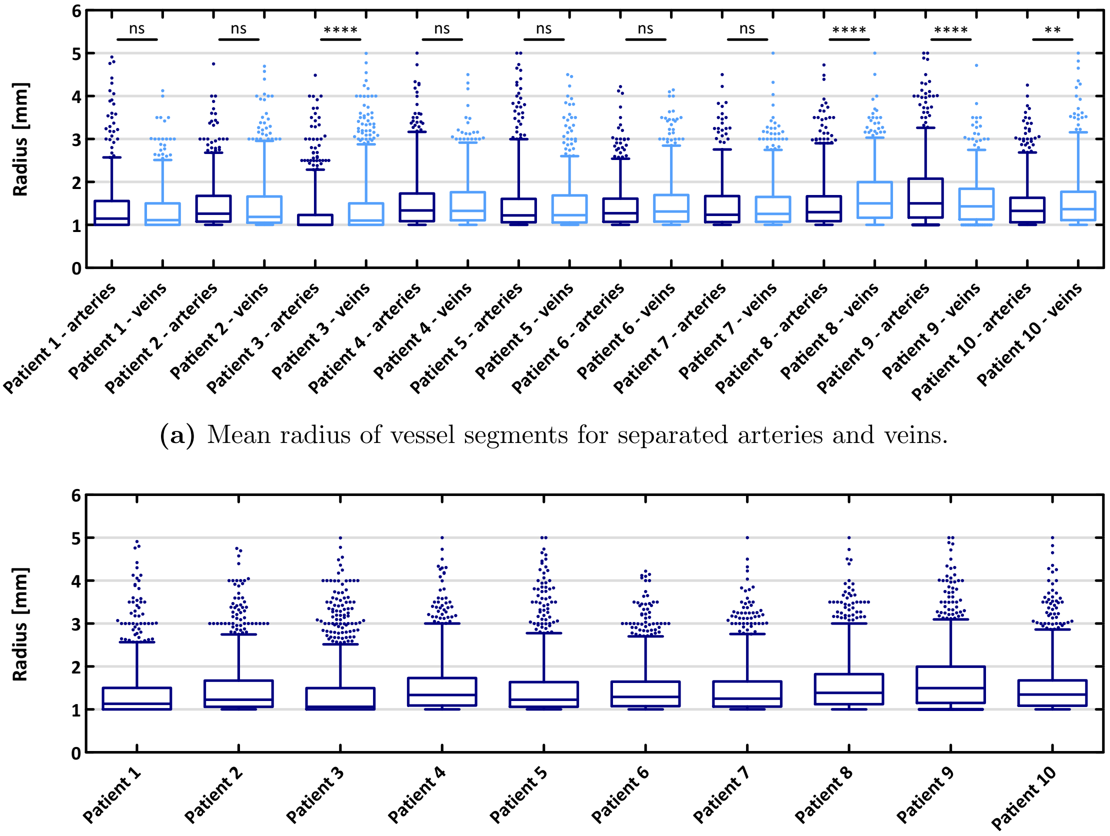
All individual mean values for the individual patients are listed in Table 5.1 and 5.2.



(a) Length of vessel segments for the separated arteries and veins.

(b) Length of vessel segments for combined arteries and veins.

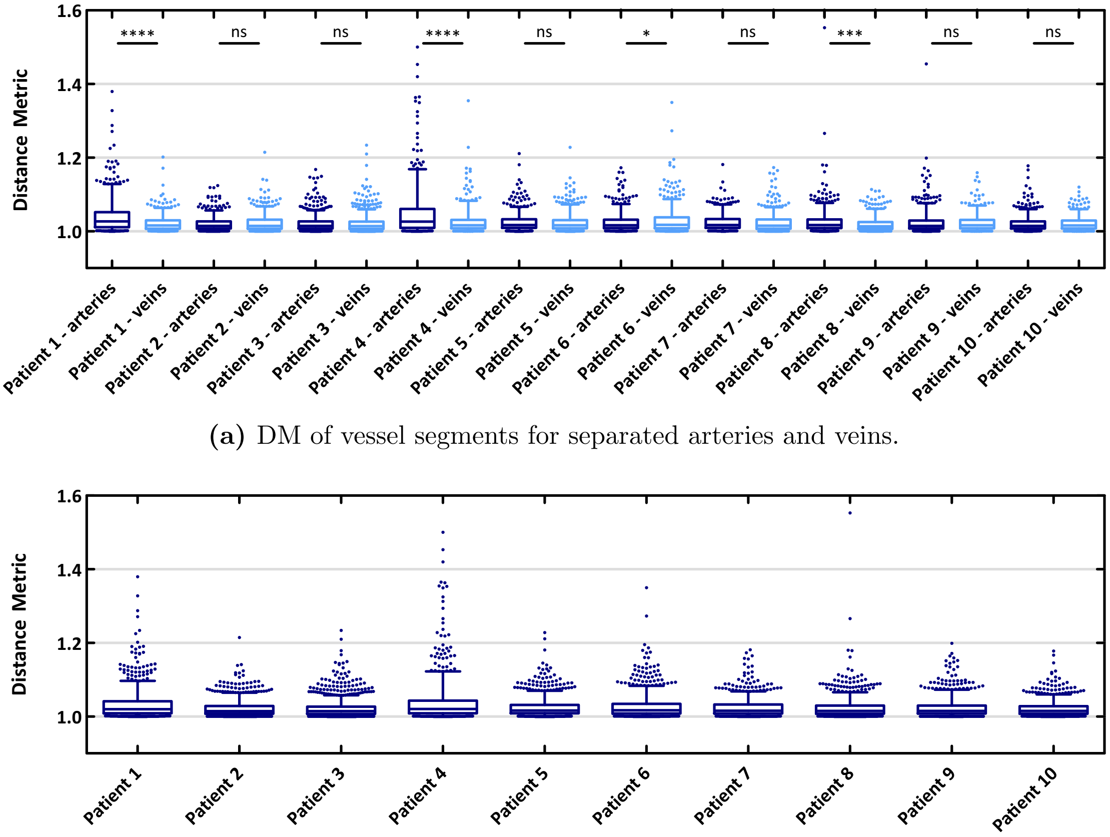
Figure 5.3: Length of all vessel segments for arteries and veins separated (a) and arteries and veins combined (b). The values are visualized as box plots, where the boxes range from the 25th to the 75th percentile, and the whiskers from the 10th to the 90th percentile. Outliers are visualized as dots. Significant differences are tested with the Mann–Whitney test, $^{*}p \leq 0.05$, $^{**}p \leq 0.01$, $^{***}p \leq 0.001$, $^{****}p \leq 0.0001$.



(a) Mean radius of vessel segments for separated arteries and veins.

(b) Mean radius of vessel segments for the combined arteries and veins.

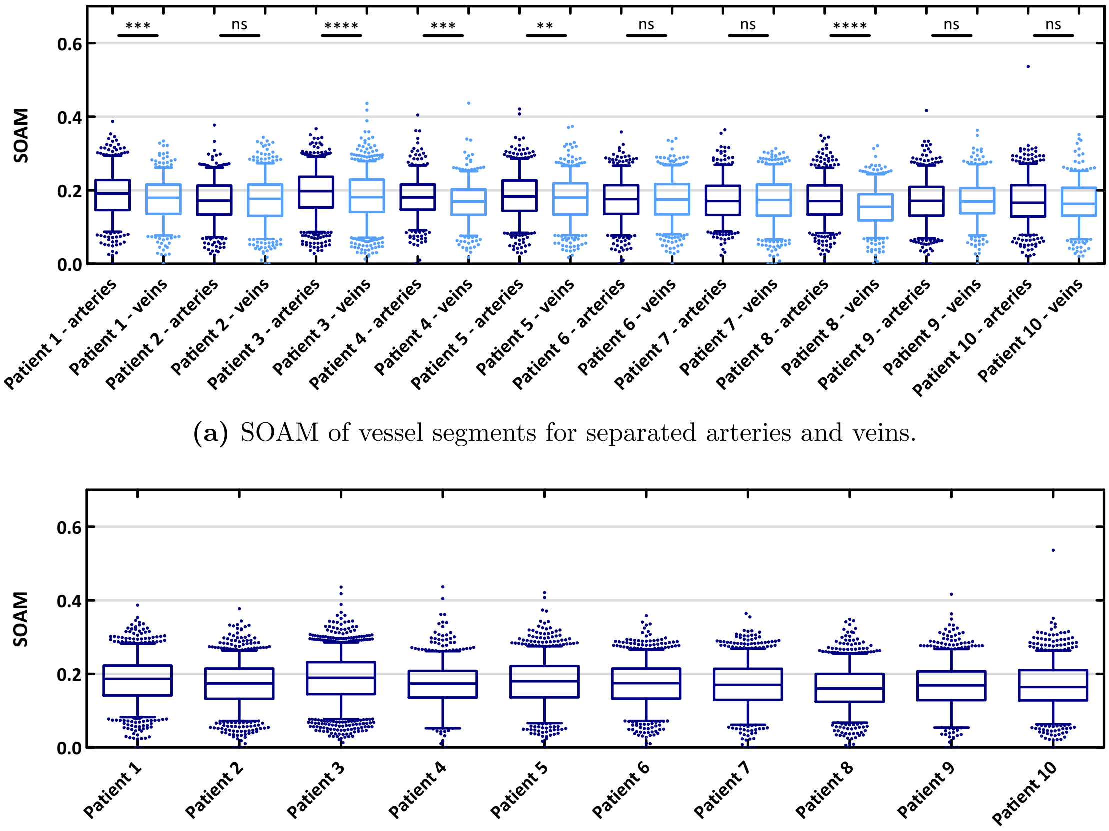
Figure 5.4: Mean radius of all vessel segments for arteries and veins separated (a) and arteries and veins combined (b). The box plots are visualized in the same way as in Figure 5.3.



(a) DM of vessel segments for separated arteries and veins.

(b) DM of vessel segments for combined arteries and veins.

Figure 5.5: *DM* of all vessel segments for arteries and veins separated (a) and arteries and veins combined (b). The box plots are visualized in the same way as in Figure 5.3.



(a) SOAM of vessel segments for separated arteries and veins.

(b) SOAM of vessel segments for combined arteries and veins.

Figure 5.6: *SOAM* of all vessel segments for arteries and veins separated (a) and arteries and veins combined (b). The box plots are visualized in the same way as in Figure 5.3.

	artery segments					vein segments				
ID	#	length	radius	DM	SOAM	#	length	radius	DM	SOAM
1	587	14.34	1.415	1.041	0.189	474	13.63	1.363	1.022	0.176
2	566	13.95	1.497	1.020	0.171	619	13.63	1.482	1.022	0.174
3	905	14.08	1.244	1.021	0.193	968	12.75	1.401	1.020	0.181
4	490	13.72	1.566	1.048	0.182	451	12.36	1.538	1.026	0.168
5	621	13.81	1.484	1.024	0.185	647	15.30	1.479	1.024	0.176
6	553	14.16	1.457	1.024	0.174	594	13.12	1.519	1.029	0.175
7	460	13.94	1.485	1.024	0.172	606	12.47	1.478	1.023	0.172
8	642	13.53	1.523	1.025	0.172	512	12.81	1.687	1.019	0.154
9	632	13.05	1.754	1.023	0.171	416	16.32	1.575	1.023	0.170
10	637	13.36	1.479	1.021	0.171	417	15.35	1.597	1.022	0.166

Table 5.1: Readouts for all datasets grouped by arteries and veins. The values show the number of vessel segments and the mean value for their length, radius, *DM* and *SOAM*, respectively.

	combined segments				
ID	#	length	radius	DM	SOAM
1	1061	14.02	1.392	1.032	0.183
2	1185	13.78	1.489	1.021	0.173
3	1873	13.40	1.325	1.021	0.187
4	941	13.07	1.552	1.037	0.176
5	1268	14.57	1.482	1.024	0.180
6	1147	13.62	1.489	1.027	0.174
7	1066	13.11	1.481	1.023	0.172
8	1154	13.21	1.596	1.023	0.164
9	1048	14.35	1.683	1.023	0.171
10	1054	14.15	1.526	1.021	0.169

Table 5.2: Readouts for all datasets for combined arteries and veins. The values show the number of vessel segments and the mean value for their length, radius, *DM*, and *SOAM*, respectively.

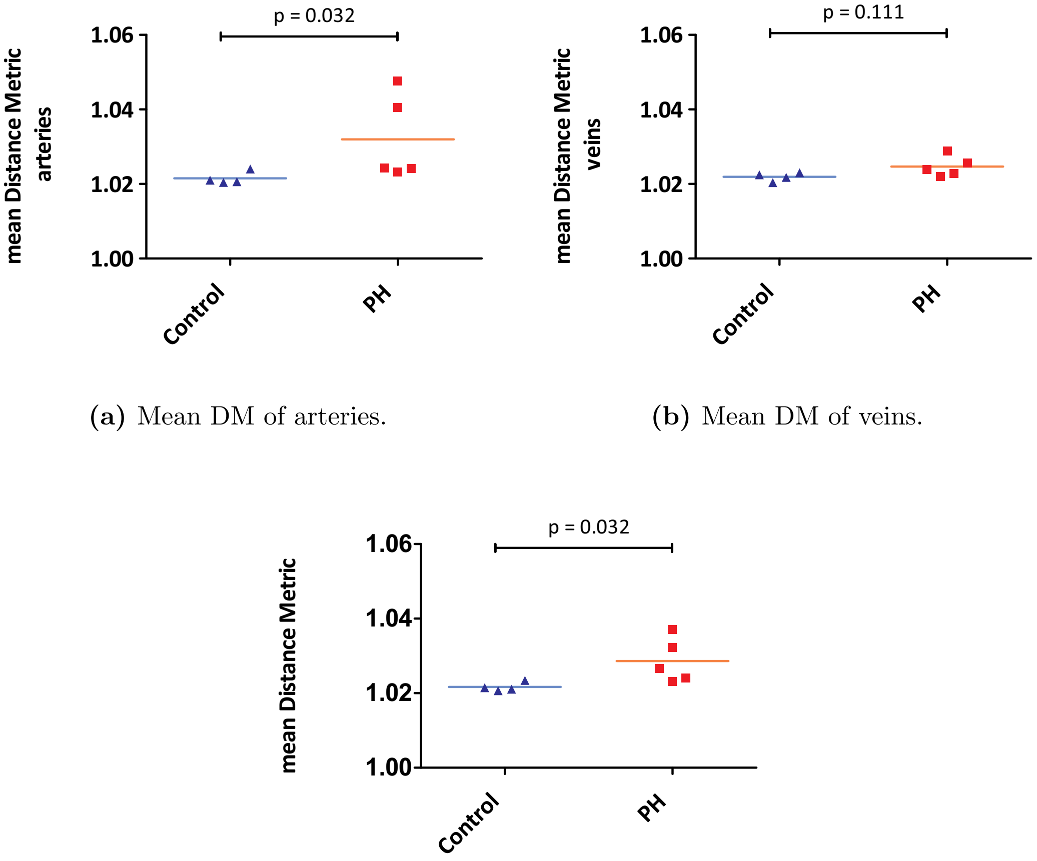
5.2.2 Correlation with Pulmonary Hypertension

In order to analyze the power of these metrics to differentiate between patients with and without *PH* we performed statistical analysis with the tortuosity readouts *DM* and *SOAM*. Nine out of the ten patients were used for this analysis, as medical data of patient #8 was not available. The patients were grouped by their Mean Pulmonary Artery Pressure (mPAP) below or above 25 mmHg, which is the defined threshold for *PH*. In this pilot group of nine patients there were five patients with and four patients without *PH*. The patient characteristics are shown in Table 5.3. Right-Heart Catheterization (RHC) was used for measuring *mPAP*. The median delay of the Computed Tomography (CT) examination to the *RHC* was 1 day with a range of 1 to 5 days.

Patient #	Age at CT examination [years]	Diagnosis	mPAP [mmHg]
1	70	IPAH	47
2	54	Scleroderma (no PH)	15
3	55	Scleroderma (no PH)	16
4	73	CTEPH	59
5	65	CTEPH	59
6	67	PAH	39
7	68	Scleroderma (no PH)	15
8	-	-	-
9	64	CTEPH	54
10	44	not known (no PH)	16

Table 5.3: Patient characteristics of ten patients of the pilot study. *PH* is diagnosed in patients with a *mPAP* $\geq$ 25 mmhg. For patient #8 medical data was not available. IPAH = Idiopathic Pulmonary Arterial Hypertension, CTEPH = Chronic Thromboembolic Pulmonary Hypertension, PAH = Pulmonary Arterial Hypertension.

Differences between groups were tested with the Mann–Whitney test, where $p \geq 0.05$ was considered significant. The mean *DM* of arteries of patients with *PH* was significantly higher than in controls ($p = 0.032$), see Figure 5.7a. The difference in the mean *DM* of veins was not significant ($p = 0.111$), see Figure 5.7b, proving that the rise in the mean *DM* of all the vessels in Figure 5.7c is driven by the bent arteries. In the case of the second tortuosity readout, *SOAM*, we did not observe differences between the two patient groups (see Figure 5.8).

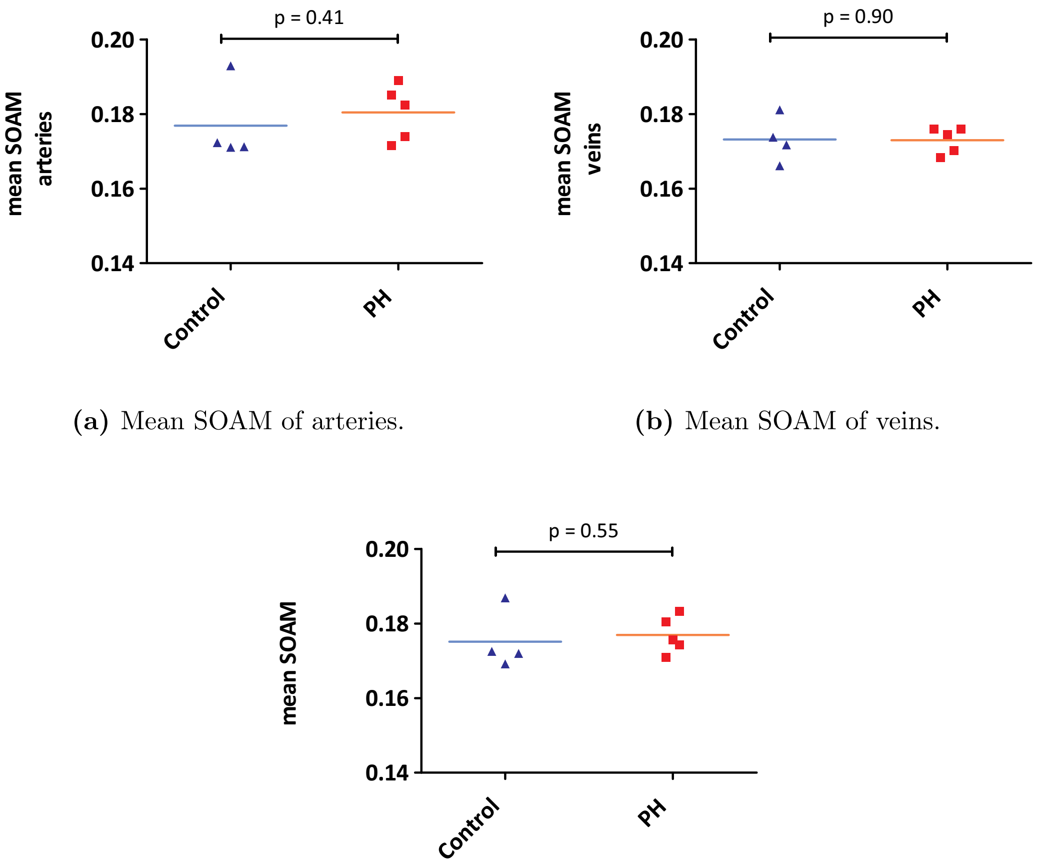


(a) Mean DM of arteries.

(b) Mean DM of veins.

(c) Mean DM of all vessels.

Figure 5.7: Correlation of the *DM* used to distinguish between patients with and without *PH*. Differences between groups were tested with the Mann–Whitney test.



(a) Mean SOAM of arteries.

(b) Mean SOAM of veins.

(c) Mean SOAM of all vessels.

Figure 5.8: Correlation of the *SOAM* used to distinguish between patients with and without *PH*. Differences between groups were tested with the Mann–Whitney test.

5.3 Discussion

We presented various quantitative readouts of the arterial and venous trees as well as their combination and tested, if they are able to distinguish between patients with and without *PH*. By calculating the tortuosity measure *DM* with our proposed method, we confirmed the previous results from Helmberger et al. [15], who showed that the *DM* is higher for patients with *PH*. By further evaluating the *DM* of arteries and veins independently, we showed that the bending of the vessels is driven rather by arteries than by veins (see Figure 5.7). We suppose that this increased bending comes from the higher pressure in arteries as compared to veins of patients with *PH*.

All other readouts did not correlate with the existence of *PH*. The mean length as well as the mean radius did not change for patients with and without *PH*, although individual changes between arteries and veins were observed (see Figure 5.3 and 5.4). Additionally, even though the *SOAM* is a more sophisticated tortuosity measure, we did not see any significant changes for patients with and without *PH*. This was caused by the very similar absolute values of the *SOAM* for arteries and veins. Nevertheless, *SOAM* values for all patients individually were either unchanged or increased in arteries as compared to veins, but never decreased. This further suggests that arteries are overall more bent than veins.

A limitation of the results is the low number of patients. As this is a pilot study, additional analyses have to be performed in future studies in order to obtain more generalizable results. Additionally, the vessel segmentation could further be refined in order to improve the accuracy of the readouts. As we did not calculate complete vessel trees, but multiple distinct subtrees, not all bifurcations could be detected. This led to some very long vessel segments, as can be observed in Figure 5.3. Due to the segment-based readout calculation, missing bifurcations could distort the results. Furthermore, generation-based readouts of the vessel trees, that may have increased value as compared to segment-based ones, could not be performed without complete trees. Therefore, we propose that a post-processing of the segmentation results by reconnecting the distinct subtrees could improve the diagnostic value. Another limitation is the low number of extracted vessel segments (see Figure 5.2). As the small vessels are also considered to contain a significant amount of information, including them into the readout calculation could further increase the value.

Moreover, the readouts were generated with the user-defined labeling instead of the automatic one. We used the user-defined labeling in order to get undistorted readouts for all patients that could influence artery- and vein-based results, because the automatic segmentation contained mislabeled and non-vessel structures (see Section 4.2.1). As there was significant interaction time needed to generate the user-defined labeling (see Table 4.1), it is not suitable in clinical practice. Additional comparison of the readouts of the user-defined labeling with the readouts of the fully automatic algorithm could show their connection with the segmentation accuracy.

6 Summary and Conclusion

We presented a fully automatic algorithm to separate arteries and veins in thoracic Computed Tomography (CT) images. Evaluation of the algorithm with 10 manually labeled reference segmentations showed an agreement of 94.1% (ranging from 85% to 98.7%), outperforming a recent state-of-the-art interactive algorithm [34].

After segmentation of the left and right side of the lung, we calculated the artery-vein (A/V) separation for each side independently. We used a four-dimensional (4D) vessel enhancement filter to calculate the certainty that a vessel with a certain radius (r) centers a specific position (x, y, z). Afterwards, we created a graph of regularly spaced local maxima of this vessel enhanced image and calculated 4D paths between neighboring maxima by minimizing a geodesic distance based on the vessel enhanced image. As this resulting 4D path graph contained spurious as well as the sought after vessel paths, the next step was to extract multiple distinct subtrees from it. An integer program calculated these subtrees by minimizing a constraint objective function with binary variables, that is weighted with the geometric relationship of adjoining path pairs. The A/V labeling step of the algorithm used two anatomical properties of arteries and veins in the lungs. The first property was that arteries and veins are uniformly distributed throughout the lungs, whereas the second property used the fact that arteries are in proximity and run in parallel to bronchi. A second integer program calculated the A/V labeling of the previously extracted subtrees by finding a configuration of binary variables that maximized weights based on these two properties. The final voxel-based A/V segmentation was generated by setting all voxels inside the spheres traversing the 4D arterial and venous vessel paths accordingly. These fully automatic segmentations achieved a mean agreement of 94.1% with 10 manually labeled reference segmentations, outperforming a reimplementation of a recent state-of-the-art interactive algorithm [34] with a mean agreement of 91.9%. Additionally, by intervening in the subtree A/V labeling step in the same way as for the interactive algorithm, we were able to achieve a mean agreement of 98.8%, suggesting excellent separation of the vessel subtrees.

As the extracted 4D vessel paths represent a vessel segmentation and its centerline at the same time, subsequent calculation of quantitative readouts could be performed. We showed that by quantifying the tortuosity of vessel segments with the Distance Metric (DM), we are able to distinguish between patients with and without Pulmonary Hypertension (PH). Vessels of patients with *PH* had a higher *DM* as compared to patients without *PH*, and by further differentiating the *DM* of arteries and veins, we observed that the bending in the arteries is the driving force of this change.

Additional improvement in the A/V separation algorithm could be achieved by employing a lung lobe segmentation in order to calculate arterial and venous trees for each lobe independently. This could reduce errors caused by the A/V labeling part of the algorithm while simultaneously decreasing runtime. Furthermore, a post-processing to merge the distinct subtrees to a single arterial and a single venous tree could improve the final segmentation as well as the quantitative readouts. In conclusion, our proposed method provides an opportunity to become an integral part of computer aided diagnosis of lung diseases.

A

List of Acronyms

CAD	Computer Aided Diagnosis
CT	Computed Tomography
DECT	Dual-Energy Computed Tomography
DM	Distance Metric
GVD	Generalized Voronoi Diagram
HU	Hounsfield Unit
mPAP	Mean Pulmonary Artery Pressure
MRI	Magnetic Resonance Imaging
OOF	Optimally-Oriented Flux
PH	Pulmonary Hypertension
RANSAC	Random Sample Consensus
RHC	Right-Heart Catheterization
SOAM	Sum of Angles Metric

Bibliography

[1] Aurenhammer, F. (1991). Voronoi Diagrams – A Survey of a Fundamental Data Structure. *ACM Computing Surveys*, 23(3):345–405. 39, 40

[2] Bauer, C. (2010). *Segmentation of 3D Tubular Tree Structures in Medical Images*. Phd thesis, Graz University of Technology. 8

[3] Benmansour, F., Türetken, E., and Fua, P. (2013). Tubular Geodesics using Oriented Flux : An ITK Implementation. *The Insight Journal*. 6, 27, 29, 31, 47

[4] Benza, R. L., Miller, D. P., Barst, R. J., Badesch, D. B., Frost, A. E., and McGoon, M. D. (2012). An evaluation of long-term survival from time of diagnosis in pulmonary arterial hypertension from the reveal registry. *Chest*, 142(2):448–456. 2

[5] Bullitt, E., Gerig, G., Pizer, S. M., Lin, W., and Aylward, S. R. (2003). Measuring Tortuosity of the Intracerebral Vasculature from MRA Images. *IEEE Transactions on Medical Imaging*, 22(9):1163–1171. 6, 59, 60, 61

[6] Bülow, T., Wiemker, R., Blaffert, T., Lorenz, C., and Renisch, S. (2005). Automatic extraction of the pulmonary artery tree from multi-slice CT data. In *Proceedings of SPIE*, volume 5746, pages 730–740. 6, 17, 18, 19, 20, 39, 41

[7] Cañero, C. and Radeva, P. (2003). Vesselness enhancement diffusion. *Pattern Recognition Letters*, 24:3141–3151. 3, 7, 8

[8] Dice, L. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26:297–302. 50

[9] Dijkstra, E. W. (1959). A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1(1):269–271. 15

[10] Estépar, R. S. J., Kinney, G. L., Black-Shinn, J. L., Bowler, R. P., Kindlmann, G. L., Ross, J. C., Kikinis, R., Han, M. K., Come, C. E., Diaz, A. A., Cho, M. H., Hersh, C. P., Schroeder, J. D., Reilly, J. J., Lynch, D. A., Crapo, J. D., Wells, J. M., Dransfield, M. T., Hokanson, J. E., and Washko, G. R. (2013). Computed tomographic measures of pulmonary vascular morphology in smokers and their clinical implications. *American Journal of Respiratory and Critical Care Medicine*, 188:231–239. 3

[11] Foley, W. D., Mallisee, T. A., Hohenwalter, M. D., Wilson, C. R., Quiroz, F. A., and Taylor, A. J. (2000). Multiphase hepatic CT with a multirow detector CT scanner. *American Journal of Roentgenology*, 175(3):679–685. 9, 10

[12] Francis, I. R., Cohan, R. H., McNulty, N. J., Platt, J. F., Korobkin, M., Gebremariam, A., and Ragupathi, K. (2003). Multidetector CT of the liver and hepatic neoplasms: Effect of multiphasic imaging on tumor conspicuity and vascular enhancement. *American Journal of Roentgenology*, 180(5):1217–1224. 9

[13] Frangi, A. F., Niessen, W. J., Vincken, K. L., and Viergever, M. A. (1998). Multiscale vessel enhancement filtering. *Medical Image Computing and Computer-Assisted Invervention - MICCAI'98. Lecture Notes in Computer Science, vol 1496*, 1496:130–137. 3, 7, 9

[14] Gao, Z., Grout, R. W., Holtze, C., Hoffman, E. A., and Saha, P. K. (2012). A new paradigm of interactive artery/vein separation in noncontrast pulmonary CT imaging using multiscale topomorphologic opening. *IEEE Transactions on Biomedical Engineering*, 59(11):3016–3027. 14

[15] Helmberger, M., Pienn, M., Urschler, M., Kullnig, P., Stollberger, R., Kovacs, G., Olschewski, A., Olschewski, H., and Bálint, Z. (2014). Quantification of tortuosity and fractal dimension of the lung vessels in pulmonary hypertension patients. *PLoS ONE*, 9(1):e87515. 3, 6, 8, 21, 25, 71

[16] Hoeper, M. M., Bogaard, H. J., Condliffe, R., Frantz, R., Khanna, D., Kurzyna, M., Langleben, D., Manes, A., Satoh, T., Torres, F., Wilkins, M. R., and Badesch, D. B. (2013). Definitions and diagnosis of pulmonary hypertension. *Journal of the American College of Cardiology*, 62(25 SUPPL.):D42–D50. 3

[17] Hu, Q., Abràmoff, M. D., and Garvin, M. K. (2013). Automated separation of binary overlapping trees in low-contrast color retinal images. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8150 LNCS:436–443. 9

[18] Joshi, V. S., Reinhardt, J. M., Garvin, M. K., and Abràmoff, M. D. (2014). Automated method for identification and artery-venous classification of vessel trees in retinal vessel networks. *PLoS ONE*, 9(2):1–12. 9

[19] Kirbas, C. and Quek, F. (2004). A review of vessel extraction techniques and algorithms. *Computing Surveys*, 36:81–121. 3, 7, 8

[20] Kitamura, Y., Li, Y., Ito, W., and Ishikawa, H. (2013). Adaptive higher-order submodular potentials for pulmonary artery-vein segmentation. In *Fifth International Workshop on Pulmonary Image Analysis*, pages 53–61. 4, 16, 17, 20, 57

[21] Lau, Q. P., Lee, M. L., Hsu, W., and Wong, T. Y. (2013). Simultaneously identifying all true vessels from segmented retinal images. *IEEE Transactions on Biomedical Engineering*, 60(7):1851–1858. 9, 10

[22] Law, M. W. K. and Chung, A. C. S. (2008). Three dimensional curvilinear structure detection using optimally oriented flux. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5305 LNCS, pages 368–382. 3, 6, 7, 27

[23] Lesage, D., Angelini, E. D., Bloch, I., and Funka-Lea, G. (2009). A review of 3D vessel lumen segmentation techniques: Models, features and extraction schemes. *Medical Image Analysis*, 13(6):819–845. 3, 7, 8

[24] Li, H. and Yezzi, A. (2007). Vessels as 4-D curves: Global minimal 4-D paths to extract 3-D tubular surfaces and centerlines. *IEEE Transactions on Medical Imaging*, 26:1213–1223. 8, 9, 26

[25] Linguraru, M. G., Pura, J. a., Van Uitert, R. L., Mukherjee, N., Summers, R. M., Minniti, C., Gladwin, M. T., Kato, G., Machado, R. F., and Wood, B. J. (2010). Segmentation and quantification of pulmonary artery for noninvasive CT assessment of sickle cell secondary pulmonary hypertension. *Medical Physics*, 37(April):1522–1532. 2

[26] Manniesing, R., Viergever, M. A., and Niessen, W. J. (2006). Vessel enhancing diffusion. A scale space representation of vessel structures. *Medical Image Analysis*, 10:815–825. 3, 7, 8

[27] Masutani, Y., MacMahon, H., and Doi, K. (2002). Computerized detection of pulmonary embolism in spiral CT angiography based on volumetric image analysis. *IEEE Transactions on Medical Imaging*, 21(12):1517–1523. 2

[28] Matsuoka, S., Washko, G. R., Dransfield, M. T., Yamashiro, T., San Jose Estepar, R., Diaz, A., Silverman, E. K., Patz, S., and Hatabu, H. (2010). Quantitative CT Measurement of Cross-sectional Area of Small Pulmonary Vessel in COPD. Correlations with Emphysema and Airflow Limitation. *Academic Radiology*, 17(1):93–99. 3

[29] Matsuura, Y., Kawata, N., Yanagawa, N., Sugiura, T., Sakurai, Y., Sato, M., Iesato, K., Terada, J., Sakao, S., Tada, Y., Tanabe, N., Suzuki, Y., and Tatsumi, K. (2013). Quantitative assessment of cross-sectional area of small pulmonary vessels in patients with COPD using inspiratory and expiratory MDCT. *European Journal of Radiology*, 82(10):1804–1810. 3

[30] Mekada, Y., Nakamura, S., Ide, I., Murase, H., and Otsuji, H. (2006). Pulmonary artery and vein classification method using spatial arrangement features from X-ray CT image. In *Proc. 7th Asia-pacific Conference on Control and Measurement*, pages 232–235. 19, 20

[31] Murphy, K., van Ginneken, B., Schilham, a. M. R., de Hoop, B. J., Gietema, H. a., and Prokop, M. (2009). A large-scale evaluation of automatic pulmonary nodule detection in chest CT using local image features and k-nearest-neighbour classification. *Medical Image Analysis*, 13(5):757–770. 2

[32] Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66. 25

[33] Park, S., Bajaj, C., and Gladish, G. (2006). Artery-vein separation of human vasculature from 3d thoracic ct angio scans. *CompImage 2006*, pages 23–30. 11, 12

[34] Park, S., Lee, S. M., Kim, N., Seo, J. B., and Shin, H. (2013). Automatic reconstruction of the arterial and venous trees on volumetric chest CT. *Medical Physics*, 40(7):071906. 14, 15, 16, 20, 45, 46, 49, 50, 51, 57, 73

[35] Peacock, A. J., Murphy, N. F., McMurray, J. J. V., Caballero, L., and Stewart, S. (2007). An epidemiological study of pulmonary arterial hypertension. *European Respiratory Journal*, 30(1):104–109. 2

[36] Pock, T., Beichel, R., and Bischof, H. (2005). A novel robust tube detection filter for 3D centerline extraction. In *Image Analysis. 14th Scandinavian Conference, SCIA 2005. Proceedings (Lecture Notes in Computer Science Vol. 3540)*, pages 481–490. 3, 7

[37] Robben, D., Türetken, E., Sunaert, S., Thijs, V., Wilms, G., Fua, P., Maes, F., and Suetens, P. (2014). Simultaneous Segmentation and Anatomical Labeling of the Cerebral Vasculature. *MICCAI*, pages 307–314. 5, 58

[38] Rudin, L. I., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60:259–268. 47

[39] Rudyanto, R. D., Kerkstra, S., van Rikxoort, E. M., Fetita, C., Brillet, P.-Y., Lefevre, C., Xue, W., Zhu, X., Liang, J., Öksüz, I., Ünay, D., Kadipaşaog̃lu, K., Estépar, R. S. J., Ross, J. C., Washko, G. R., Prieto, J.-C., Hoyos, M. H., Orkisz, M., Meine, H., Hüllebrand, M., Stöcker, C., Mir, F. L., Naranjo, V., Villanueva, E., Staring, M., Xiao, C., Stoel, B. C., Fabijanska, A., Smistad, E., Elster, A. C., Lindseth, F., Foruzan, A. H., Kiros, R., Popuri, K., Cobzas, D., Jimenez-Carretero, D., Santos, A., Ledesma-Carbayo, M. J., Helmberger, M., Urschler, M., Pienn, M., Bosboom, D. G., Campo, A., Prokop, M., de Jong, P. A., Ortiz-de Solorzano, C., Muñoz Barrutia, A., and Ginneken, B. V. (2014). Comparing algorithms for automated vessel segmentation in computed tomography scans of the lung: The VESSEL12 study. *Medical Image Analysis*, 18:1217–1232. 3, 7, 8

[40] Saha, P. K., Gao, Z., Alford, S. K., Sonka, M., and Hoffman, E. A. (2010). Topomorphologic separation of fused isointensity objects via multiscale opening: Separating arteries and veins in 3-D pulmonary CT. *IEEE Transactions on Medical Imaging*, 29(3):840–851. 13, 14, 57

[41] Sato, Y., Nakajima, S., Shiraga, N., Atsumi, H., Yoshida, S., Koller, T., Gerig, G., and Kikinis, R. (1998). Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Medical Image Analysis*, 2(2):143–168. 3, 7

[42] Selle, D., Preim, B., Schenk, A., and Peitgen, H. O. (2002). Analysis of vasculature for liver surgical planning. *IEEE Transactions on Medical Imaging*, 21(11):1344–1357. 9, 10

[43] Sethian, J. A. (1999). *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Science*. Cambridge University Press, 2nd edition. 31

[44] Shikata, H., Hoffman, E. A., and Sonka, M. (2004). Automated Segmentation of Pulmonary Vascular Tree from 3D CT images. *Physiology, Function, and Structure from Medical Images*, 5369:107–116. 13

[45] Shikata, H., McLennan, G., Hoffman, E. A., and Sonka, M. (2009). Segmentation of pulmonary vascular trees from thoracic 3D CT images. *International Journal of Biomedical Imaging*, 2009:24:1–24:11. 3, 8

[46] Türetken, E., Benmansour, F., Andres, B., Pfister, H., and Fua, P. (2013). Reconstructing loopy curvilinear structures using integer programming. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1822–1829. 5, 6, 9, 20, 26, 32

[47] Vanderbei, R. J. (2001). *Linear Programming: Foundations and Extensions*. Springer US, 2nd edition. 24, 25

[48] Wala, J., Fotin, S., Lee, J., Jirapatnakul, A., Biancardi, A., and Reeves, A. (2011). Automated segmentation of the pulmonary arteries in low-dose CT by vessel tracking. *CoRR*, abs/1106.5:1–20. 12, 13

[49] Wolsey, L. A. (1998). *Integer Programming*. Wiley, 1st edition. 22, 25