FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Automated Processing and Visualization of Vessel Trees

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Computergraphik/Digitale Bildverarbeitung

eingereicht von

Gabriel Mistelbauer
Matrikelnummer 0326641

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer: Univ. Doz. Dipl. Ing. Dr. Miloš Šrámek
Mitwirkung: Dipl. Ing. Andrej Varchola

Wien, 25.07.2010

(Unterschrift Verfasser) (Unterschrift Betreuer)

Abstract

Automated processing and visualization of vascular structures is a common task in medical imaging. *Maximum Intensity Projection (MIP)* and *Curved Planar Reformation (CPR)* are well established and robust methods for clinical use. In case of calcified vessel walls, occlusion prevents exploring the inside of the vessels when using *MIP*. *CPR* allows to cut a single vessel along its centerline and to visualize the lumen. Extending the idea of *CPR*, a novel automatic method for vessel visualization is proposed. It works with multiple vessel centerlines that do not necessarily need to be connected into a tree structure. Arbitrarily complex vascular structures are rendered in the volume as point sets and optionally, occlusion halos are created around them to enhance depth perception. Vessel centerlines are automatically extracted from a volumetric data-set after performing feature extraction in a scale-space. The user is provided with the ability to control the final image and he or she can visually select the desired centerlines with visual queries by stroking with the mouse. Furthermore, a combination with the recent *Maximum Intensity Difference Accumulation (MIDA)* visualization technique is presented, which has the advantages of *Direct Volume Rendering (DVR)* such as occlusion and depth cues, but does not require an explicit transfer function specification. It is demonstrated how the proposed technique can be applied to large data-sets, particularly to data featuring peripheral arterial occlusive diseases or in order to detect possible embolisms as presented on a pulmonary data-set.

Kurzfassung

Die automatisierte Verarbeitung und Visualisierung von vaskulären Strukturen ist ein gebräuchlicher Ablauf im Bereich der medizinischen Bildverarbeitung. *Maximum Intensity Projection (MIP)* und *Curved Planar Reformation (CPR)* sind gut etablierte und robuste Methoden für den klinischen Einsatz. Im Falle verkalkter Gefäßwände verhindern visuelle Verdeckungen das Erkunden des Inneren der Adern, bei der Anwendung von *MIP*. *CPR* hingegen erlaubt einen Schnitt entlang der Mittelachse und somit die Darstellung des Querschnitts. Als Erweiterung der Idee von *CPR* wird eine neuartige automatische Methode zur Visualisierung von Blutgefäßen vorgestellt. Diese funktioniert auch mit mehreren Mittelachsen von Adern, welche nicht notwendigerweise verbunden sein müssen, oder gar eine Baumstruktur vorweisen. Beliebig komplexe vaskuläre Strukturen werden im Volumen als Punktemengen dargestellt und weiters kann man optional, um die Tiefenwahrnehmung zu erhöhen, Okklusion-Halos hinzufügen. Nach der Durchführung einer Merkmalsextraktion im Scale-Raum, werden die Mittelachsen der Adern automatisch aus dem Datenvolumen extrahiert. Dem Benutzer wird die Möglichkeit gegeben das finale Bild zu justieren und die gewünschten Adern, mit Hilfe von visuellen Abfragen anhand der Mausbewegung, zu selektieren. Darüber hinaus wird eine Kombination mit der kürzlich veröffentlichten Visualisierungstechnik namens *Maximum Intensity Difference Accumulation (MIDA)* vorgestellt. Diese Technik hat die Vorteile des *Direkten Volumen Renderings (DVR)*, nämlich Verdeckungs- und Tiefeninformationen, allerdings benötigt sie keine explizite Angabe einer Transferfunktion. In der vorliegenden Arbeit wird die Anwendung der erwähnten Technik auf große Datensätze gezeigt, welche sowohl für die Feststellung von Embolien, speziell Lungenembolien, als auch für die Diagnose von peripheren Verschlusskrankheiten geeignet ist.

Acknowledgement

First I would like to thank my supervisor Miloš Šrámek for his extensive support and advises during the development of the thesis. He opened an interesting topic to me and guided me through the thesis with great effort. Moreover, I have learnt a lot from his great experience and technical knowledge. Furthermore, I would like to thank Andrej Varchola for his effort in order to deploy the thesis and for his technical discussions, since they always have been fruitful.

Secondly, I would like to thank Rüdiger Schernthaner for providing several CT data-sets and his corporation and interest in using recent programs in clinical environments. His openness to incorporate new technologies into the medical daily routine is a great pleasure. Additionally, thanks goes to Jianming Liang, from the Department of Biomedical Informatics at Arizona State University, for the availability of the pulmonary data-set, which is extensively used in this thesis.

Furthermore, I would like to thank Sabine Crapouse-Wünsch for spending plenty of time and effort to read and comment this thesis concerning the language grammar.

Special thanks goes to my girlfriend Sandra for supporting and motivating me during the development of the thesis and, furthermore, made it possible for me to focus on the thesis. Moreover, I would like to thank my brother Florian who always supported me with his technical knowledge during the study.

Finally, I would like to thank my parents for financing as well as supporting me during my study at the university.

Contents

List of Figures

List of Tables

List of Abbreviations

AVT	AngioVis Toolbox
BPB	Back Propagation Buffer
BPQ	Boundary Pixel Queue
CP	Control Point
CPL	Candidate Point List
CPR	Curved Planar Reformation
CR	Centerline Reformation
CT	Computed Tomography
CTA	Computed Tomography Angiography
CUDA	Compute Unified Device Architecture
DVR	Direct Volume Rendering
DOHT	Distance-Ordered Homotopic Thinning
DSA	Digital Subtraction Angiography
DTI	Diffusion Tensor Imaging
F3D	File 3D
fMRI	functional MRI
FPB	Front Propagation Buffer
GL	OpenGL
GPPL	Global Propagated Point List
GPU	Graphics Processing Unit
GSL	GNU Scientific Library
HU	Hounsfield Units
HT	Hysteresis Threshold
IBS	Initial Boundary Set
ITK	Insight Toolkit
LPL	Local Propagation List
MIDA	Maximum Intensity Difference Accumulation
MIP	Maximum Intensity Projection
MRI	Magnetic Resonance Imaging
MRA	Magnetic Resonance Angiography
mpCPR	Multipath Curved Planar Reformation
PE	Pulmonary Embolism
PET	Positron Emission Tomography
PI	Pixel Index

RF	Radio Frequency
SP	Sampling Position
SPECT	Single-Photon Emission Computed Tomography
US	Ultrasound

Preface

Visualization of any type of information has become very popular in various domains. One of these domains is the application of visualization in medicine. Since recent computers are very powerful and scale well according to computation power, many different visualization techniques in the medical domain can be realized nowadays, even in real-time, which gives a medical doctor the great possibility to explore regions of interest on the fly.

In this thesis the focus is on vessel visualization with the aim to detect calcifications or any other type of occlusion, such as an embolism. This is a highly important task in the field of radiology, consequently several problems have to be investigated in order to provide the desired results. Furthermore, the target group are medical doctors who should be provided with an easily usable and understandable program, which is a challenging task, since medical doctors and computer scientists may operate on different interpretations of what is important and should be focused on.

The first goal of this thesis is the automation of the currently manual process of vessel detection and classification. Whereas the initial focus was on lower human arteries, we aim for generality. Since a rather skilled medical doctor needs about one hour to classify the arteries of lower human extremities, it would be a desirable and comfortable improvement to have the modern computers doing the work with a minimum of user interaction. We will describe an approach for automated vessel detection and take further possibilities about the availability of underlying hardware into account. For example, improvements according to computation time are assessed by optimization with assembler, *GNU Scientific Library (GSL)* and *Compute Unified Device Architecture (CUDA)*. Further improvements in regard to parallel execution of processes according to available system memory of the computer are taken into account by streaming technologies.

The second part of the thesis deals with vessel visualization with the initial main goal to detect calcifications and further suspicions, such as a pulmonary embolism. In this part a novel method, called *Centerline Reformation (CR)*, is described. The advantages of *CR*, in contrast to the currently common methods such as *CPR* and *Multipath Curved Planar Reformation (mpCPR)*, are that not necessarily connected vessels and arbitrarily spatially oriented vessels can be rendered and visualized. The main difference between *CR* and *CPR* lies in the applied sampling method in order to display the lumen of the vessel. *CPR* primarily samples along horizontal lines, which leads to incorrect sampling whenever an almost horizontal vessel is encountered. Furthermore, *CPR* samples the whole image, whereas *CR* samples according to the extent of the vessel resulting in a more convenient display of the vessel without its surrounding parts. Additionally, it should be mentioned that all these methods, *CPR*, *mpCPR*

and *CR* work in real-time, even if implemented in a non-hardware accelerated way. Optionally, halos can be added around the vessels when using *CR* to enhance depth perception whenever displaying lots of overlapping and occluded vessels and subjective selection of vessels by interaction is possible.

Finally, focus and context rendering is investigated, with the aim to support the user with additional information from surrounding areas for user-specified vessel bundles. In principle, focus rendering is based on specific domain semantics, for example the minimum length or the minimum average width of a vessel segment. The user has the possibility to interactively select specific values of these two constraints and, furthermore, can select the desired vessels easily by just striking and left clicking with the mouse. The *CRs* of the corresponding vessels are displayed in real-time and the user can visually explore them by zooming and panning. In addition, a specific context can be optionally rendered to preserve the overall perception and orientation within the whole data-set.

1. Introduction

1.1 Motivation

Due to the increased computation power of modern computers and the possibilities to incorporate modern medical image data acquisition techniques, the field of visualization in medicine has become very attractive and popular. The main goal of visualization is the presentation of information in such a way, that medical doctors can utilize it, because sometimes the visually pleasing visualizations are of no use. Furthermore, some parts can be automated in order to provide the medical doctor with additional information and data in less time, which can be essential in cases where an urgent diagnosis is required.

Angiography is a very important section in medicine, and in this thesis the focus is on automating the detection of vessels and the visualization of the lumen of selected vessels of interest. With the aim to visualize arbitrary vessel trees, application areas such as pulmonary embolism detection are accessible. Moreover, the manual detection and isolation of vessels are very time consuming tasks, for which reason automation would be a great advantage. Although this still requires some user input, the aim is to reduce this to a minimum and to provide a robust and accurate process.

The following sections describe the medical background of the main application fields investigated in this thesis. Volumetric data-sets containing and outlining various diseases have been used for either automated detection of vessels or visualization of the desired regions, such as stents, narrowed vessels or an embolism.

1.1.1 Atherosclerosis

Atherosclerosis is a vascular disease of primary medium or large arteries which is characterized by vascular inflammation and thickening of the artery wall resulting from a buildup of plaque such as lipids, cholesterol, calcified deposits and other cellular residues, as stated in [29] and outlined in Fig. 1.1. Commonly this buildup restricts the blood flow. One can distinguish between two different types of plaque, namely hard and soft plaque. The former leads to hardening of vessel walls whereas the latter is more likely to break apart and enter the blood stream. Such blood clots can totally or partially block the blood stream which causes organs to starve due to lack of oxygen and blood. Analytical models concerning atherosclerosis are extensively described in [53].

Basically, the symptoms of atherosclerosis are angina pectoris, if arteries around the heart are afflicted (coronary artery disease) or sudden numbness as a sign of atherosclerosis in arter-

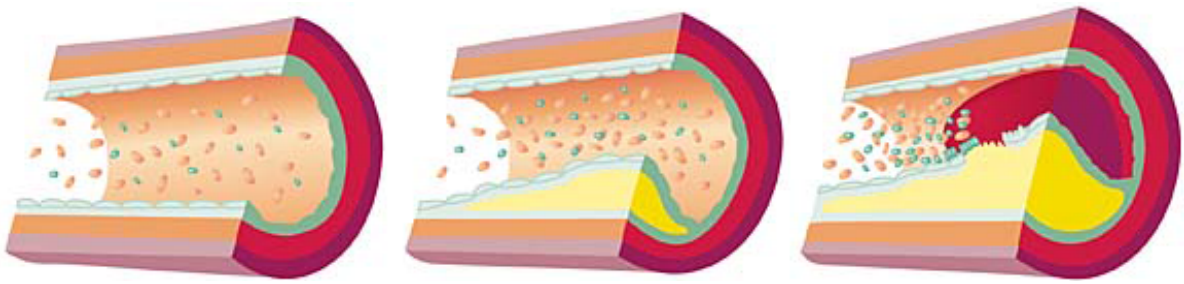


Figure 1.1: *Development of atherosclerosis. The left image shows a healthy vessel where the blood stream is not affected by any obstacle, whereas the central image displays the residue of plaque along the vessel wall leading to a narrowed vessel with increased blood pressure. The right image presents a clot hindering the blood stream from continuing. (Image source:* `http://www.diabetes-deutschland.de/archiv/4818.htm`*)*

ies leading to the brain (transient ischemic attack) or even pain during walking if encountered in arteries supporting arms or legs (peripheral arterial disease). Hence the symptoms heavily depend on the arteries affected.

Common potential risk factors of atherosclerosis are active smoking (increases the risk of atherosclerosis mainly in lower extremities), diabetes (which possibly leads to affliction even of small vessels), hypertension, advanced age, obesity, hypercholesteremia and no physical activity, as described in [31; 47; 3].

Diagnosis is performed non-invasively by means of medical imaging or invasively by inserting a catheter into an artery to inspect the region of interest. Since the latter approach is invasive, extensive effort has been made to extend the possibilities of medical imaging using *Computed Tomography Angiography (CTA)* or *Magnetic Resonance Angiography (MRA)* for data acquisition in addition with a contrast-agent to enhance vessels. Moreover, a stenosis can be detected with a stethoscope inspecting the sound of the blood resulting from turbulent flow through partly occluded vessels.

Clinical treatment of atherosclerosis such as surgical interventions can be mainly divided into two types, angioplasty which is a method to enlarge the narrowed arteries, and bypass surgery to create a new blood support in the case of even more narrowed vessels. Furthermore, two different techniques in angioplasty are used, which are shown in Fig. 1.2. The first method widens the narrowed artery with a balloon, whereas the second one places a stent to ensure a persistent continuous blood flow, as described in [31]. Additionally, an example data-set is displayed in the left image of Fig. 1.2, in order to present a stent in the lower extremities of a human body. When planning such surgeries the length and size of the stent must be accurately determined and of course its exact final position should be known too. Hence, visualization techniques can be helpful or even necessary in such cases. Furthermore, a bypass surgery is illustrated in Fig. 1.3.

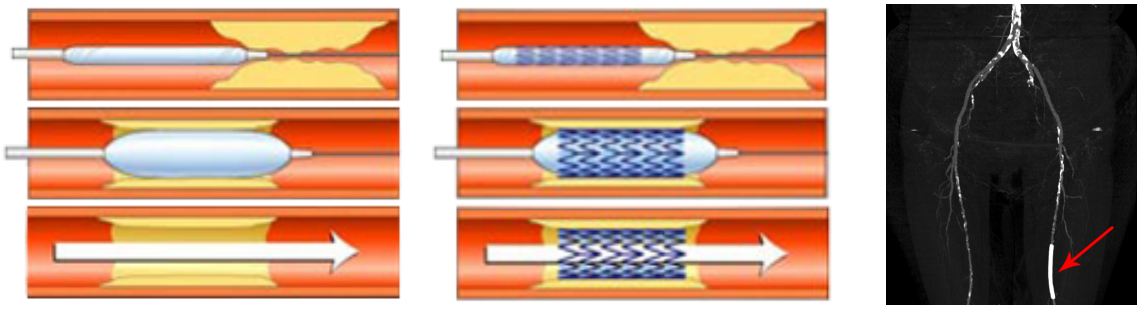


Figure 1.2: *Treatment of atherosclerosis. The left image shows a balloon angioplasty to widen the narrowed vessel, whereas the middle picture presents the technique of stent implantation. In the right image a stent (red arrow) is acquired by a CT scanner and visualized using MIDA. (Source of the left and middle image:* `http://www.herz-praxis.ch/`*, right image created using the AngioVis Toolbox)*

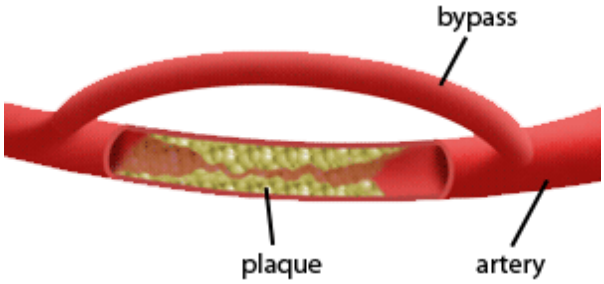


Figure 1.3: *Illustration of a bypass which has to be applied due to heavy calcification of the artery (Image source:* `http://www.reshealth.org/`*)*

1.1.2 Pulmonary Embolism

As described by Goldhaber [13], *Pulmonary Embolism (PE)* is a common and dangerous disease which has attracted only recently more interest and effort of medical doctors. Since *PE* is difficult to detect and diagnose, fast medical support and urgent interventions are necessary in order to save the life of the patient. Basically, an embolism is a clump of hardened plaque which has detached itself from a vessel wall and entered the blood stream, as illustrated in Fig. 1.4. This clot can block small vessels or hinder the blood from continuing at all, which can cause severe damage and pain.

Potential risk factors should be known by a greater range of people to successfully prevent the cause of a pulmonary embolism. Vulnerable people should undergo diagnosis more frequently with the aim to detect possible *PE*. As described in [13; 24; 41], potential environmental risk factors are very long travels in air planes or buses, obesity, hypertension, smoking and immobility, whereas a naturally conditioned risk factor is age. Illnesses such as cancer, diabetes and thrombophilia facilitate a possible onset of *PE*. Most of the *PE* are a consequence of thrombi originated in the deep veins of the legs or in the pelvic region, as stated in [13; 24]. In the case of deep vein thrombi detaching, they are transported through the venous system to

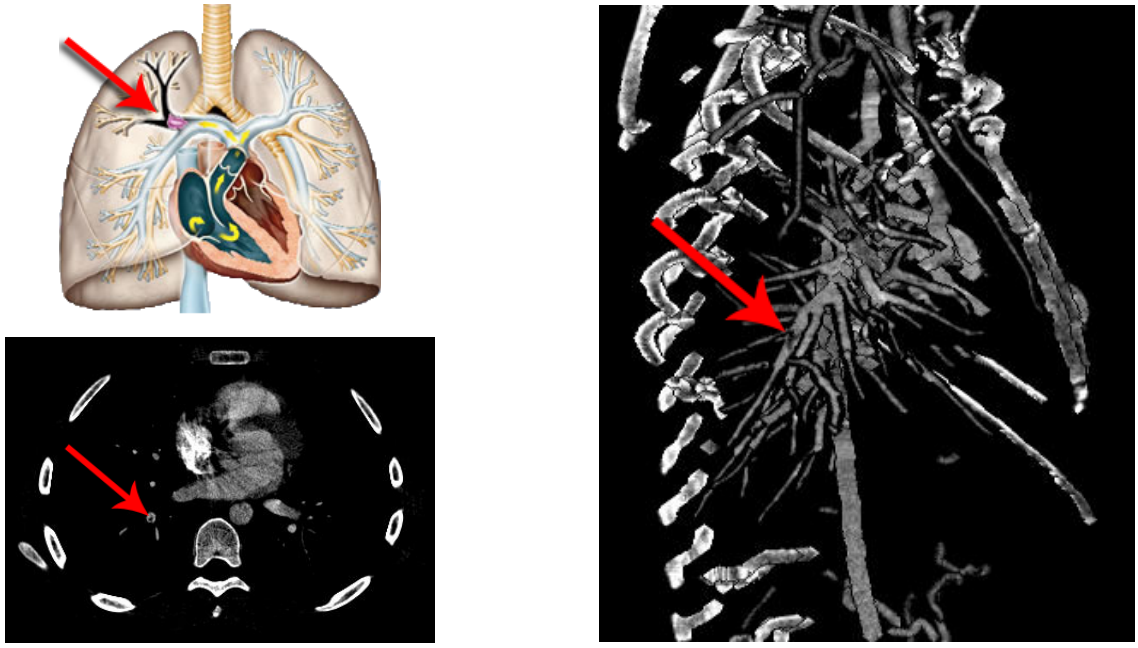


Figure 1.4: *Illustration and example of a pulmonary embolism acquired by CTA. The upper left image shows an illustration of a pulmonary embolism (the clot is shown by the red arrow and the directions into which the clot is moving through the blood stream and the heart are indicated by yellow arrows). The lower left image shows the slice view of a pulmonary embolism (notice that it is very difficult to detect the embolism, even with contrast stretching). The right image presents the CR of all vessels resulting from the vessel detection pipeline (note that even bones are detected since they have not been removed beforehand). (Image sources: upper left image* `http://www.apotheken-umschau.de/Lungenembolie`*, lower left and right image created using the AngioVis Toolbox)*

the pulmonary arterial system and eventually reach the lungs where they may block a whole tree of vessels, resulting in a reduced blow flow, or in the worst case none at all. Some consequences of *PE* are less gas exchange due to significant alveolar dead space caused by vascular obstruction, loss of gas exchange surface and possibly lung haemorrhage, as mentioned by Goldhaber [13].

Diagnosis of *PE* is a challenging task, since in contrast to other illnesses common symptoms are not present, but fast treatment must be provided. It gets even worse if patients have two joint diseases, because *PE* might not be recognized during a first attempt of diagnosis. As stated in [13; 54], the traditional gold standard for diagnosis has been invasive contrast pulmonary angiography, but nowadays it is only done when further suspicion remains after inspecting a *Computed Tomography (CT)* scan of the lung. Apart from that, [54] mentions that it is both an invasive technique and requires a rather skilled radiologist and a compliant patient. Furthermore, Goldhaber [13] states that potentially *MRA* will be widely used in the future as a main diagnosis tool. However Wells [54] supports the view that ultrasound is very accurate for lower extremities. Additionally, as described later in this thesis, some special visualization techniques, such as *CPR* or *CR*, with the aim to display the lumen of vessels, will significantly

improve detection of embolisms in volumetric data-sets, as outlined in Fig. 1.4.

The usual treatment of *PE*, as mentioned by Kyrle and Eichinger [24], are thrombolysis, embolectomy, anticoagulation or the insertion of a vena cava filter. For patients with heavy *PE* and cardiogenic shock, the first two types of treatment should be applied whereas the vena cava filter is advisable only in cases of contraindications to anticoagulants. Furthermore, additional catheter-based therapies are described in [51].

1.1.3 Aneurysm

Aneurysm is a blood-filled extension of a vessel, mostly occurring in arteries, but sometimes even in veins, as illustrated in Fig. 1.5. The cause of an aneurysm is either a disease, for instance atherosclerosis, or a weakening of the vessel wall. With increasing size of an aneurysm, eventual rupture with following haemorrhage will be more likely. This leads to a significant increase of morbidity. Additionally, a clot can be released from the inside of the aneurysm leading to severe danger such as thrombosis, as described in [27].

As mentioned by Sharma et al. [46] for a complete diagnosis of aneurysms *Computed Tomography (CT)*, *Magnetic Resonance Imaging (MRI)* and *Digital Subtraction Angiography (DSA)* are used to advise a medical doctor in making a decision for further treatment and surgery planning. Whereas *CT* reveals more details about calcifications and the extent of the aneurysm as well as the inner clot, *MRI* provides information concerning possible haemorrhage and the transition between the aneurysm and the surrounding parts. Moreover *DSA* is mostly applied, although recent techniques, such as *CTA* and *MRA* have made the former method obsolete, due to its disadvantage of being invasive. An aneurysm is displayed in Fig. 1.6 where the left image shows the *MIP* of the aneurysm data-set and the right image presents the lumen of all detected vessels utilizing *CR*.

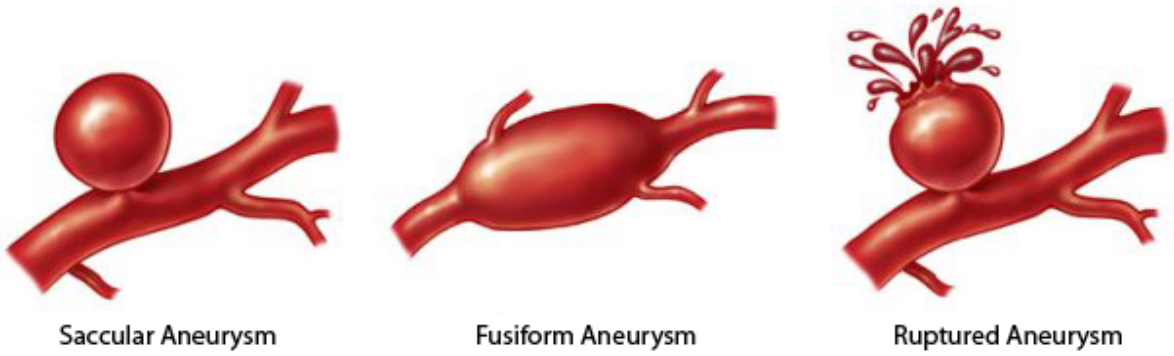


Figure 1.5: *Illustrations of aneurysms. The left image shows a saccular aneurysm (which is commonly located in the brain, called cerebral aneurysm), whereas the middle image presents a fusiform aneurysm (for example an aortic aneurysm). In the right image the severe danger, namely an eventual rupture, is illustrated. (Image source:* `http://neuro.wehealny.org/endo/cond_aneurysms.asp`*)*

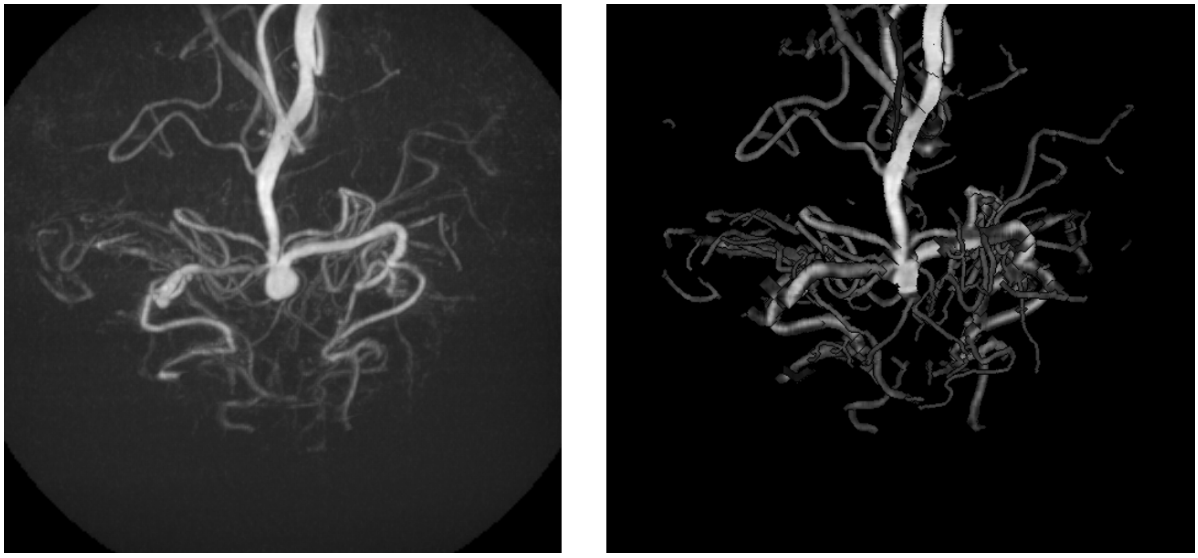


Figure 1.6: *Example of a saccular aneurysm. The left image shows the MIP of the aneurysm, whereas the right image presents the CR of all detected vessels. (Images created using the AngioVis Toolbox)*

Potential risk factors of an aneurysm are smoking, hypertension, diabetes, hyperlipidemia and advanced age, as outlined in [2; 15; 4]. Baman et al. [2] state further on, that the average size of an aneurysm does not depend on the previously mentioned factors.

Common treatment of aneurysms consists either of endovascular techniques, such as angioplasty using stents, or open surgery methods such as clipping off the aneurysm, as stated in [46]. In cases where clipping is not possible, wrapping or trapping or even a bypass technique are applied, whereas another method, namely coiling, aims to eliminate the aneurysm as a consequence of a clotting reaction initiated by coils inserted into it, as described in [50].

1.2 Data Acquisition

Medical image data can be acquired for various purposes, such as therapy and operation planning, monitoring the progression of diseases and, of course, diagnosis, which is the primary application field. Initially, a patient describes his or her problem and a medical doctor needs to interpret the symptoms. If some suspicious symptoms are revealed, medical imaging would possibly present necessary and helpful information to relate the symptoms to the cause. Furthermore, medical imaging supports the doctor in decision making, because various visualization techniques may illustrate the data in a very convenient and understandable way to focus on the suspicion or the regions of interest.

Several methods of image data acquisition exist, as described in [38]. The most common among these are *X-ray*, *Computed Tomography (CT)* and *Magnetic Resonance Imaging (MRI)*, but there are also several others such as *Positron Emission Tomography (PET)* and

Single-Photon Emission Computed Tomography (SPECT), which are widely used in nuclear medicine. Furthermore, *Ultrasound (US)* can be used as a source for medical image data as well.

The application field of Angiography resides in imaging of arteries. Often, it is an invasive procedure, since a catheter is inserted into the patient in order to reach and visualize the lumen of the desired vessels. Computer imaging and visualization modalities can enhance and improve angiography and, additionally, they may become a non-invasive modality. Moreover, they may result in a more comfortable and clinically faster localization of the region of interest and possibly in a much smaller and more precise intervention, since the planning can be more accurately performed.

Some image modalities for angiography are *Digital Subtraction Angiography (DSA)*, *Computed Tomography Angiography (CTA)* and *Magnetic Resonance Angiography (MRA)*, as stated in [48; 38]. Each of them will be described briefly in the following sections.

1.2.1 Digital Subtraction Angiography

Digital Subtraction Angiography (DSA) is an invasive method, as stated in [48; 38], since a catheter is inserted into the arterial system. Furthermore, to emphasize the arterial system a contrast medium is injected (usually iodine-based). As described in [38], *DSA* tracks the blood flow by first taking an image of the object before the contrast agent is injected. This image is referred to as *mask image*. Afterwards a series of images, called *frames*, is acquired while the contrast agent moves through the arteries. The final output of the *DSA* is generated by subtracting the frames from the mask image. This leads to the exclusive display of the blood flow, as mentioned in [48] and examples images are presented in Fig. 1.7. Advantages of this method are very high resolution, high contrast and objects possibly obstructing vision are removed with the mask image [38].

Figure 1.7: *Examples of Digital Subtraction Angiography (DSA) images of human lower extremities. (Images source: Straka [48])*

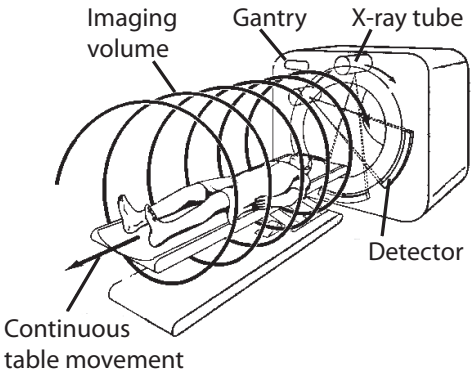
1.2.2 Computed Tomography Angiography

Basically, *Computed Tomography (CT)* produces several individual X-ray images in modern scanners, namely the axis slices, which are composed into one volume data-set, as described in [38]. The X-ray source together with the detector are rotated around the patient who is continuously moving in an orthogonal direction in order to get a full data volume, as illustrated in the left image of Fig. 1.8. The slice images are reconstructed by measuring the X-ray attenuation of a full rotation. The tomographic reconstruction is based on the *Radon transform* which is described in [33].

Initially, the scanners used to be slow and hence artifacts caused by motion occurred. Today, as described in [38], the state-of-the-art systems are helical or spiral scanners with low acquisition time. Furthermore, this evolution gives the possibility to inject a contrast agent intravenously and scan a desired region of interest. This method is called *Computed Tomography Angiography (CTA)* since vessels are emphasized by the contrast agent in order to make the vessels more visible and distinguishable from other types of tissue. Moreover this technique has advanced to be a robust and accurate non-invasive imaging method for detecting and inspecting arterial diseases, as stated by Straka [48].

An advantage of *CTA* is the intravenous injection of the contrast agent in opposition to the usual injection into an artery in the case of *DSA*, which is obviously more invasive. Since the volume data-set of *CTA* can be large, the laborious task to issue a radiologic interpretation is very time consuming, because numerous slices have to be examined in detail. Hence, intensive effort has been made to automate or at least advance this task to a level which requires a minimum of user interaction.

As mentioned in [48], data-sets are not isotropic since the spatial resolution within the slice is typically less than 1mm, but the resolution between slices is lower. The attenuation of the X-rays, namely the density, for a given material, such as tissue, is given in *Hounsfield*



Tissue	HU
Air	–1000
Lung tissue	–900 to –170
Fat tissue	–220 to –30
Water	0
Pancreas	10 to 40
Liver	20 to 60
Heart	20 to 50
Kidney	30 to 50
Contrast-agent enhanced blood	150 to 500
Bones	45 to 3000
Metal	>3000

Figure 1.8: *Illustration of a CT-scanner in the left image. The right table shows the Hounsfield Units (HU) of various tissue types, as stated in [38; 48]. (Left image source:* `http://www.researchmalignantmesothelioma.com/ct-scan.html`*)*

Units (HU). The density varies only slightly in *CT* images, the table in Fig. 1.8 shows a general definition of densities and corresponding types of tissue. Some types of tissue have overlapping values, which makes it hard to distinguish between them, as for example contrast-agent enhanced blood and bone, causing automated bone removing to be a challenging task.

Furthermore, as stated by Straka [48], since sampling can be described as a function that weighs the attenuation of X-rays, an averaged density value within the support of the point-spread function of the scanner is given. This leads to blurring of the acquired volume and averaging the attenuation values of different tissues in border regions. This artifact is called *partial volume effect*. Additionally, it should be stated that metal parts, such as stents or implants, cause significant artifacts too.

It should be briefly mentioned that the intensity of the radiation plays an important role as well, because the higher the dose of radiation the more impaired the patient's tissue might be. Consequently, efforts are made to reduce the amount of radiation needed while still preserving quality and accuracy of the acquired volume data.

1.2.3 Magnetic Resonance Angiography

Magnetic Resonance Angiography (MRA) is a derivative of *Magnetic Resonance Imaging (MRI)* with the additional use of a contrast agent. Furthermore, *MRA* is a non-invasive technique too and allows visualization of blood in vessels with a special setting of parameters, as described in [48]. *MRI* is based on various properties of tissues under the influence of a strong magnetic field which is aligned along the axis of the scanner. It is noteworthy that the image quality heavily depends on the strength and homogeneity of this magnetic field.

As Straka [48] mentions, principally the spin of hydrogen nuclei in human tissue is exploited to generate the image. While under the influence of an external magnetic field the spins of the atomic nuclei align themselves either parallel or anti-parallel, since they can be considered as small magnetic dipoles. Pulses of electromagnetic energy, *Radio Frequency (RF)* pulses, are applied perpendicularly to the initial magnetic field, resulting in non-aligned high energy states of some initial magnetically aligned nuclei. Those high-energy nuclei emit energy at measurable rates during their relaxation and realigning times, which is recorded and reveals information about the environment. The volumetric information measured are usually the proton density and T1 and T2 relaxation times which further need to be reconstructed for each voxel.

In order to reconstruct the spatial relationship of the assessed signal, two additional orthogonal magnetic gradients need to be applied. Hence most *MRI* scanners have three orthogonal gradient coils to be able to generate magnetic field gradients along three orthogonal directions, namely X, Y and Z as described in [32; 38]. Furthermore, this reveals the option to create arbitrary slices in contrast to *CT*-scans where this is not possible.

Another technique for reconstructing the spatial information is described in [38; 48], namely using the discrete Fourier transformation. In this case only one gradient field is required. The *RF* intensity is encoded into the intensity of the signal and the position is encoded into the frequency of the signal.

Furthermore, it should be mentioned that contrast agents with magnetic properties are used, usually Gadolinium compounds, in addition with an image weighted according to the

first relaxation time (T1), hence called T1-weighted image. As a result, enhanced structures appear bright in the T1-weighted images and vascular structures become significantly more visible with the contrast enhancement, as stated in [38].

In [38] several other derivatives of *MRI* with their application fields are described, such as *functional MRI (fMRI)* with the focus on cerebral blood flow and *Diffusion Tensor Imaging (DTI)* which utilizes the diffusion property of water to detect the fiber direction of heart muscles or neural pathways.

1.3 Data Processing

To create meaningful images which allow inspection of the desired regions of eventual suspicions, the acquired data has to be post-processed in order to provide the necessary information. As described by Straka [48], the post-processing can be roughly divided into the following steps:

1. Data acquisition using one of the modalities already mentioned
2. Reconstruction of the volumetric data out of the acquired axial slices
3. Segmentation of various types of tissue (bone, vessel, soft tissue)
4. Vessel extraction
5. Additional immediate and interactive vessel visualization to identify the regions of interest or to get a first impression of eventual suspicions
6. Resampling of the volumetric data and image generation
7. Storing the images using special medical software
8. Final interpretation of the generated images and further treatment planning

1.4 F3D Library

The name *F3D* is an abbreviation for *File 3D*. Basically it is a programming library for handling 3D volumetric files with a broad variety of algorithms concerning operations on data volumes. Since in this thesis the main prerequisite is to work on volumetric data, this library is well suited. Additionally, this library has two major goals which are utilized and investigated in this thesis.

The first target is to implement most algorithms for volumetric data in a streaming fashion. In contrast to common implementations where the whole data volume is loaded into the system memory at once, streaming approaches of *File 3D (F3D)* operate on a per slice basis. On the one hand a much better memory consumption and utilization is achieved, since only one slice at a time is loaded into the system memory. On the other hand, not all operations require or can operate on only a single slice, if, for example, neighborhood information has to be taken into account, for example, the Gaussian or Hessian filter. To overcome this restriction, slabs of slices have been introduced, which can store a specified number of slices. Thus a trade–off between memory allocation and computation time has to be taken into account, since if

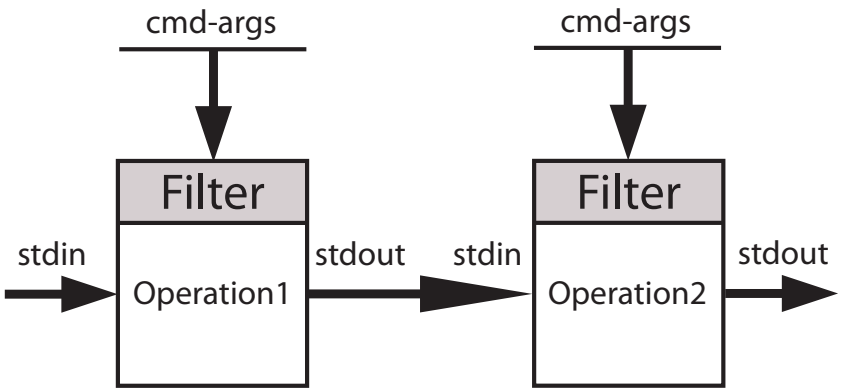


Figure 1.9: *Illustration of the operation paradigm implemented in the F3D library, where each filter performs one operation. Specific parameters are controlled via command-line arguments and the input and output volumes are passed respectively via standard input and output. Due to the fact, that each filter runs in a single process, no interprocess communication is necessary between each operation.*

less information is provided the more computation time it will take, and vice versa, if more information is given or stored, the faster it will be.

Furthermore, the *F3D* library aims to make every operation atomic. This means that every operation is represented within a single filter and should work with a minimum of user interaction by command-line arguments. Furthermore, the standard input and standard output are utilized to provide the operation with the necessary data volume files. The advantage of this approach is the possibility to connect several operations by simply connecting the output of one filter to the input of the next one, where each filter operates within a separate and single process, as outlined in Fig. 1.9. Additionally, process grids and other process pipelines can be modelled without any further efforts, such as interprocess communication. These properties are used to implement a specific task scheduler for the vessel detection pipeline which is described later in Section 3.1.1. Basically, this scheduler will handle the execution of operations in a specific manner.

Additionally, most operations can be accelerated by using at least one of the following three techniques. The first one is optimization in assembler using SSE, SSE2 instruction sets and much more recent ones, which perform multiple operations simultaneously. The second technique as well as the third approach utilizes the *Graphics Processing Unit (GPU)*, whereby the former is realized by using *OpenGL (GL)* and the latter by using the recent *CUDA* for further optimization. The desired type can be selected by the user via the standard switches if implemented for the operation.

Summing up, the *F3D* library is a highly generic library with several optimization techniques and a huge selection of various operations on volumetric data. It is designed with a high degree of modularity and can be extended well. Its most significant advantage is the design to support various ways to implement operations on volumetric data in a streaming fashion.

1.5 The AngioVis Toolbox

The primary goal of the *AngioVis Toolbox (AVT)* is to support a medical doctor with several features and information concerning angiography to facilitate decision making or surgery planning. Logically it can be divided into four main parts, namely the background identification stage, the bone remover, the vessel tracker and the batch image generator. Fig. 1.10 illustrates the application field of the *AVT* in the clinical workflow.

During the background identification stage, the data of interest, from the acquired *CTA* data-set, is separated from the table the patient is lying on. The user selects circles which identify the fore- and background and according to them, the volume data gets cropped. Although this is a quite straight forward approach, some background artifacts can still remain due to overlapping or being too close the desired data.

The second step is the bone remover which is kind of semi-automated, since not all bones can be separated automatically, because of problems caused by the contrast agent applied to enhance vessels. Since the enhanced vessels have about the same density as bone, it is difficult to distinguish between bone and vessel, as shown in Fig. 1.11. Hence, this has to be resolved manually by the user and is a topic for further concerns and future investigations, since this is a very time consuming process which needs a lot of expertise with the *AVT* program.

After the background has been removed and the bones have been separated, the vessel tracker is the next in stage. The first part of the thesis focuses on this, since it investigates automated vessel tracking to relieve the medical doctor of this very time consuming task,

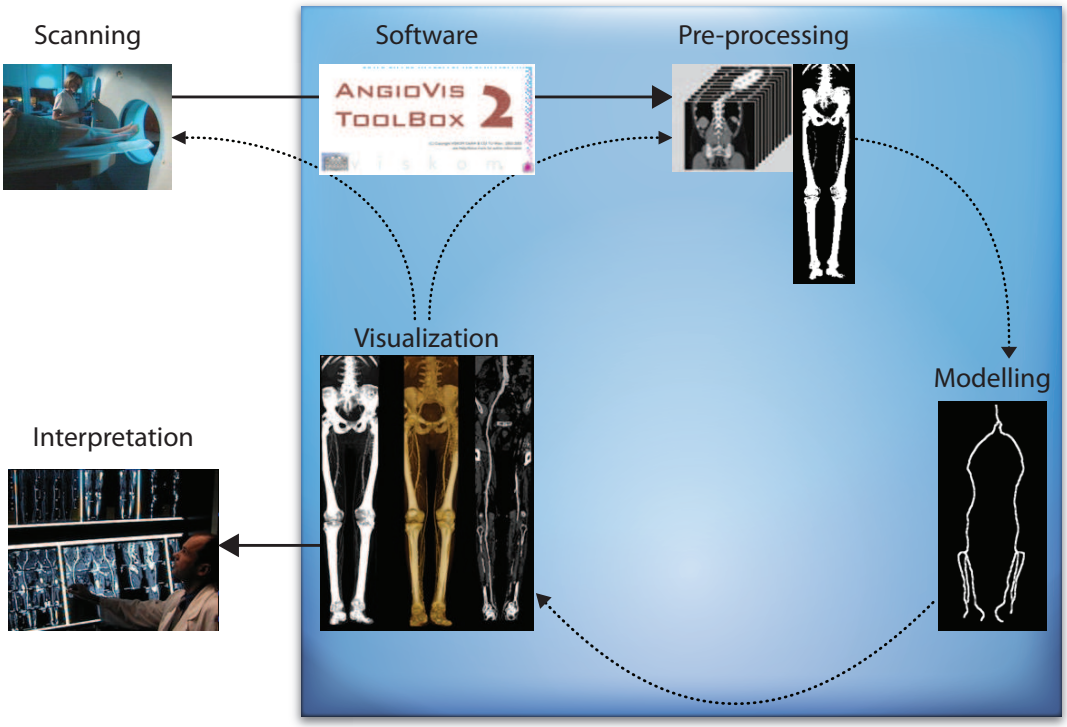


Figure 1.10: *Illustration of the application field of the AVT in the clinical workflow*

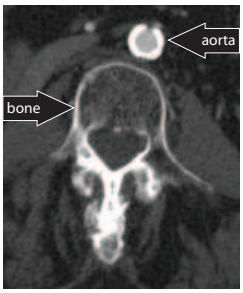


Figure 1.11: *Example of a vessel, the aorta, with almost the same density as bone. In this case manual distinction is necessary for correct classification. (Image created using the AngioVis Toolbox)*

if manually done. After the vessel tracking is finished, the user can select the vessels of interest. These vessels are interactively visualized using a specific technique, namely *CPR* or *CR*, in order to display the lumen of those vessels, as presented in Fig. 1.12. Furthermore, a medical doctor can select vessels according to specific domain semantics, which operate in the sense of a filter. Moreover, the selection of vessels is interactively rendered and the user can additionally zoom and pan to view the desired vessels from various directions and angles.

The fourth and additional or optional step is the batch image generator, which is utilized to generate a series of images from the current data-set. These images can be used by medical doctors to support their decision making and suggestions for further treatment strategies.

More technically, the *AVT* is programmed in C++ using QT widgets for the interface. Moreover, it can easily be extended via a plugin system, which loads a list of shared object files at the program start-up and thus offers the possibility to extend the system with future solutions concerning specific medical problems. `www.angiovis.org` provides more information and program related content about the *AVT*.

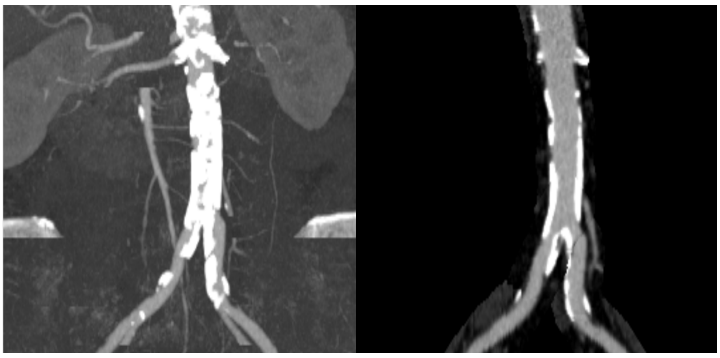


Figure 1.12: *Example of a heavily calcified aorta. In the left image the vessel seems to be almost completely narrowed, because of using MIP, whereas in the right image Centerline Reformation is utilized to show the lumen of the vessel. (Images created using the AngioVis Toolbox)*

2. Related Work

2.1 Vessel Enhancement

Since imaging technologies have improved a lot in recent years due to the increasing power of modern computers, they have become an indispensable tool. Images acquired by *Magnetic Resonance Imaging (MRI)* or *Computed Tomography (CT)* and further imaging possibilities are widely spread in the medical domain. Since such images provide information about patients, they have come to play a crucial role in diagnostic processes. Furthermore, operation and therapy planning could benefit from additional information, and visualizations of the human vasculature would play an important role. Mostly the steps from acquisition, using *MRI* or *CT*, to the evaluation of the images, are performed manually, which takes a lot of time. Hence, a lot of effort has been dedicated to automate those processes, as stated in Kirbas and Quek [22].

The description of blood vessels from medical images plays an important role in medical diagnosis of vessels, such as planning stent operations resulting from heavy calcifications or others such as neural pathway extraction in *DTI*, as described in [32]. Nowadays nobody can think about planning an operation without any additional information about the disease of the patient, and that is why accurate image acquisition and visualization of the parts of interest play an important role. In this section we focus on extracting the human vasculature out of a volumetric data-set.

A common way to interpret the images of a *CT* or *MRI* is the *Maximum Intensity Projection (MIP)*. The drawback of this method is that other structures can obscure vessels and that small vessels with low contrast will not be visible [12]. One way to overcome this problem is to use the recent *Maximum Intensity Difference Accumulation (MIDA)* as described in [5] or to use more advanced vessel enhancement techniques, which will give a better overall result. As an example, the «Pock filter» should be mentioned, as described by Pock et al. [37].

After preparing the acquired volumetric data-set in the pre-processing step, the pipeline continues with vessel detection and extraction. It finally ends with a tree-like representation of the vessels, which can be further utilized, for example, by converting it into a graph representation and loaded with the *AVT*. Basically, the pipeline can be divided into three parts, namely pre-processing, segmentation and conversion to the desired representation.

In the following sections related work for several parts of the pipeline will be presented, with the focus on detecting vessels and determining their centerlines. Furthermore, properties related to stability and smoothness regarding the centerline as well as streaming approaches, in order to avoid memory consumption overflow, will be discussed.

2.1.1 Segmentation

Several techniques for vessel segmentation from 2D image data or 3D volumetric data have been investigated and developed, each with its own advantages and drawbacks. Kirbas and Quek [22] distinguish between six main categories of segmentation approaches, as briefly described in the following:

1. **Pattern recognition techniques**. This approach deals with automatic detection or classification of objects or features by using pattern recognition approaches [22]. The approach implemented and applied in this thesis belongs to this category, namely the scale-space technique.
2. **Model-based approaches**. Here a vessel model is created and applied to the data. Depending on the model, vessels of different shapes are extracted. The model needs to suit the object to extract well in order to get acceptable results, as stated in [22]. Furthermore, active contour models, called *snakes*, should be mentioned, which are described in [21] in detail. Another method which should be brought up is the *Fast Marching Method* which uses a wave propagation approach and is extensively delineated in [43; 44].
3. **Tracking-based approaches**. This technique applies a local operator to a vessel, determined by an initial starting seed-point, whereas in contrast pattern recognition approaches apply local operators to the whole data. The vessel is tracked by analyzing pixels perpendicular to the tracking direction as stated by Kirbas and Quek [22]. One noteworthy disadvantage is the lack of full automation, since a seed-point must be chosen manually.
4. **Artificial intelligence-based approaches**. In this approach vessels are extracted according to a previously determined knowledge base. This base can either be a general model of vessels or some image processing algorithms. This technique is accurate but the complexity of the computation is much higher in contrast to other methods described in Kirbas and Quek [22].
5. **Neural network-based approaches**. This category uses neural networks, which are widely applied in pattern recognition, for vessel detection. On the one hand, neural networks have the advantage of self learning and the ability to use non-linear classification boundaries acquired from the training of the network. On the other hand, every time a new feature is added to the training set, the whole network needs to be retrained, as mentioned in [22].
6. **Miscellaneous tube-like object detection approaches**. Following Kirbas and Quek [22] this category deals with approaches not necessarily designed for vessel detection such as the scale-space analysis, which is, for example, the basis of the Pock's filter, described in [36; 37], and further vessel enhancement filters that extract tubular structures as described and discussed in [11; 12; 8].

Each approach has its strengths and weaknesses, as for example the neural network technique which applies well for contour detection, but needs to learn from a training set. But if features change, the training set needs to be rebuilt. Furthermore, if the knowledge base of

some artificial intelligence approach is altered, a relearning process is necessary, which may consume a huge amount of time [22].

After structures or objects have been enhanced using one of the methods explained above, a threshold is applied as a post-processing operation in order to extract only the significant structures. There are various thresholding techniques which try to reduce the noise resulting from a threshold operation with just one value. The method utilized and adapted in this thesis is Hysteresis Threshold, described by Csetverikov [9]. The advantage of this method is to pass only edges of special properties.

Since the main goal of the desired overall implementation is to be as highly automated as possible, because the clinicians should not and cannot interact with the application for every patient, the decision goes for the scale-space approach which will be discussed in greater detail in the next section.

2.1.2 Scale Space Analysis

2.1.2.1 Gaussian Filter

As mentioned by Pock [36], nothing in the real world exists without the notion of scale. Every object can be described at a specific scale and this can be extended to objects consisting of several scales, for example the branches of a tree. Going from real world object description to image analysis, an object must be of some scale or defined under a specific scale. Hence, objects corresponding to a specific scale must be detectable in an image (either 2D or 3D, which is not so important to distinguish in principle).

The first step in scale space analysis is filtering the initial image with a Gaussian filter. The one-dimensional equation of the filter is described by Equation 2.1. Higher dimensions of the Gaussian filter are recursively created using the one-dimensional filter kernel, which is an important property according to computation time and streaming approaches, namely separability.

$$G(\mathbf{x}, \sigma) = \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}} e^{-\frac{(x_1^2+x_2^2+\ldots+x_n^2)}{2\sigma^2}} \tag{2.1}$$

The separability property of the Gaussian filter results from its symmetry. In other words, this means that applying a two-dimensional filter to an image equals the application of two one-dimensional filters to that image, one for each dimension. Going into higher dimensions this property states, that convolving a n-dimensional continuous signal with a n-dimensional Gaussian kernel, $G_N(x_1, x_2, \ldots, x_N, \sigma)$, equals convolving the former signal with the number n times the one-dimensional Gaussian kernels $G(x_1, \sigma)G(x_2, \sigma)\ldots G(x_N, \sigma)$. The advantage of this property leads to a computation improvement with regard to performance and furthermore this can be utilized in streaming approaches. Note the decomposition into separable steps if using one-dimensional Gaussian kernels.

Fig. 2.1 shows the detection of several helices of different sizes by applying a Gaussian filter, outlined in the first row, followed by a Hessian filter, shown in the second row, using different sigma values as parameters. Note how different helices of different scales are stressed

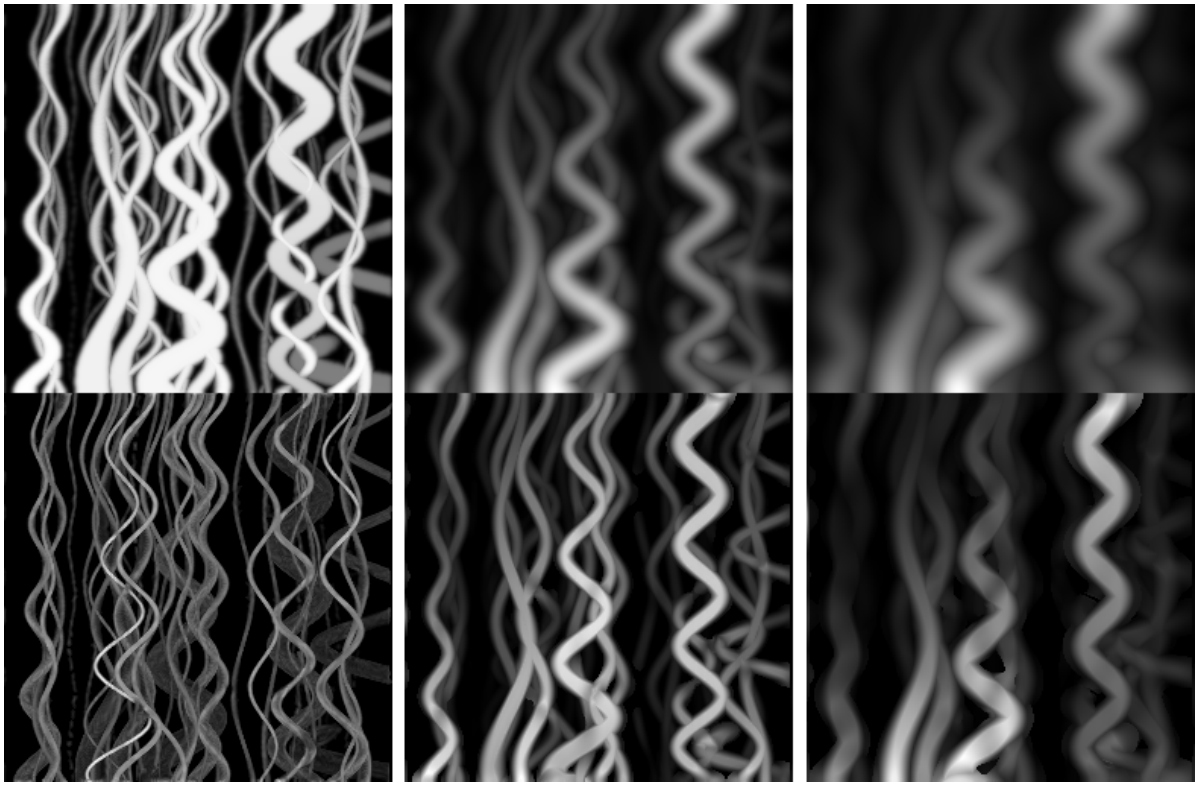


Figure 2.1: *Example of scale space detection of artificial helices of various sizes. From left to right with increasing sigma and thus helices with increased thickness are detected. The top row shows the Gaussian filter, whereas the bottom row presents the Hessian filter. Note how helices, which do not equal the desired size, which is the sigma value specified, vanish almost completely. (Images created using the AngioVis Toolbox)*

and emphasized in contrast to the ones, which do not fit to the desired scales. The latter helices are not at all emphasized and thus will not contribute to the final result.

It should be mentioned, that it is important to find filters which are invariant under translation, rotation and scaling, because the detected features should stay the same, no matter how they are oriented. Hence we have to find filters which detect such invariant features. The Gaussian filter is such a filter, which can be utilized to detect objects of different sizes, since object which are smaller than the specified sigma will be blurred in such a way, that they will be hardly visible anymore, as mentioned in [36].

2.1.2.2 Hessian Filter

As stated in Sato et al. [42], second derivatives are useful for line enhancement filters, but they are even more generally applicable, like for detecting any tube-like structures or shapes. An intensity function, $I(x)$, is convolved, denoted by the $*$ symbol, with the second derivative of the Gaussian, see Equation 2.1, to reduce the noise and to get response according to a specific

line width. The one-dimensional response of the line filter is given by

$$R(x,\sigma) = \{-\frac{d^2}{dx^2}G(x,\sigma)\} * I(x). \tag{2.2}$$

To obtain positive values for a bright line the sign of the Gaussian derivative has been inverted, as mentioned in [42]. Furthermore, the one-dimensional approach can be extended to multi-dimensions using the Hessian matrix, that describes the second-order structure of local intensity variations around every point of a multi-dimensional image, as described in [42; 36]. For a three-dimensional image, $I(\mathbf{x})$, the Hessian matrix, $\mathcal{H}(\mathbf{x})$, is defined as

$$\nabla^2 I(\mathbf{x}) = \mathcal{H}(\mathbf{x}) = \begin{bmatrix} I_{xx}(\mathbf{x}) & I_{xy}(\mathbf{x}) & I_{xz}(\mathbf{x}) \\ I_{yx}(\mathbf{x}) & I_{yy}(\mathbf{x}) & I_{yz}(\mathbf{x}) \\ I_{zx}(\mathbf{x}) & I_{zy}(\mathbf{x}) & I_{zz}(\mathbf{x}) \end{bmatrix}, \tag{2.3}$$

where $I_{xx}(\mathbf{x}) = \frac{d^2}{dx^2}I(\mathbf{x})$, $I_{yz}(\mathbf{x}) = \frac{d^2}{dydz}I(\mathbf{x})$ and so on denote the partial second derivatives of the image $I(\mathbf{x})$ and $\mathbf{x} = (x, y, z)$.

To determine local extrema the eigenvalues λ_1, λ_2, λ_3 and eigenvectors $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$ of $\mathcal{H}(\mathbf{x})$ are computed, such that $\lambda_1 \leq \lambda_2 \leq \lambda_3$ and $|\mathbf{v}_i| = 1$. They correspond to the principal curvature of $I(\mathbf{x})$. Further Pock [36] states that

$$I(\mathbf{x}) \text{ has local maximum} \leftrightarrow \lambda_i < 0 \quad \forall i = 1 \ldots 3 \tag{2.4}$$

$$I(\mathbf{x}) \text{ has local minimum} \leftrightarrow \lambda_i > 0 \quad \forall i = 1 \ldots 3. \tag{2.5}$$

Furthermore, as described in [36; 12; 11], using the eigenvalues of the Hessian matrix $\mathcal{H}$ different types of shapes can be distinguished according to an underlying elliptic model, where the eigenvectors define the orientation and lengths of the primary axes. These types of shapes are outlined in Table 2.1.

Table 2.1: *Shapes in 3D according to eigenvectors and eigenvalues of the Hessian matrix $\mathcal{H}(\boldsymbol{x})$*

λ_1	λ_2	λ_3	**Polarity**	**Shape/Structure**
noisy	noisy	noisy	–	noisy, no direction
$\ll 0$	≈ 0	≈ 0	bright	plate-like
$\gg 0$	≈ 0	≈ 0	dark	plate-like
$\ll 0$	$\ll 0$	≈ 0	bright	tubular
$\gg 0$	$\gg 0$	≈ 0	dark	tubular
$\ll 0$	$\ll 0$	$\ll 0$	bright	blob-like
$\gg 0$	$\gg 0$	$\gg 0$	dark	blob-like

One drawback of the elliptical model is that not all shapes can be modelled with this approach. Moreover, since the Hessian matrix takes only a small neighborhood into account, structures of larger diameter result in a problem. This was addressed in [36] by a multi-scale technique, where the maximum response over various scales is selected. As shown in [42] this improves the visualization.

2.1.3 Threshold

The results obtained from the Hessian-based enhancement contain too much information (false positives), hence the volumetric data needs to be thresholded, in order to extract the desired vessels. Those will be further delegated to skeletonization to retrieve the final centerlines. Hence, binary volumetric data is necessary, since skeletonization just needs to distinguish between foreground and background points. Therefore a threshold operation must be applied. Since a threshold possibly removes a lot of information we must carefully choose an appropriate threshold value. If a too high value is selected, too much information will be lost and most of the vessels would neither be detected nor connected. But on the other hand, if a too small value is taken, too much unwanted information and noise would remain, possibly resulting in very noisy centerlines.

The simplest method is a threshold operation with just one value. Although this is a quite common and fast approach, it can lead to at lot of noise, since undesired local maxima, which have values above the threshold, will still be present after the threshold operation. In the case of vessel extraction this is not satisfying, because on the one hand, those remaining local maxima may not correspond to a desired vessel and hence are just noise. On the other hand, points which indeed correspond to a vessel, will vanish, as they are under the chosen value, leading to unconnected vessel parts. An example of this is presented in the middle image of Fig. 2.2, note how noisy this image looks alike, since there exist undesired small segments. Furthermore, this noise can lead to problems using skeletonization afterwards, because this process is very prone according to noise, which will further influence the resulting centerlines. Hence this has to be avoided, because smooth centerlines are desired, in order to visualize the lumen of the vessels accordingly.

In order to reduce the noise and to improve connected components, such as vessels, Csetverikov [9] introduced a method called *Hysteresis Threshold (HT)*, which needs two threshold values as input. The first value is used as a low threshold to determine all points

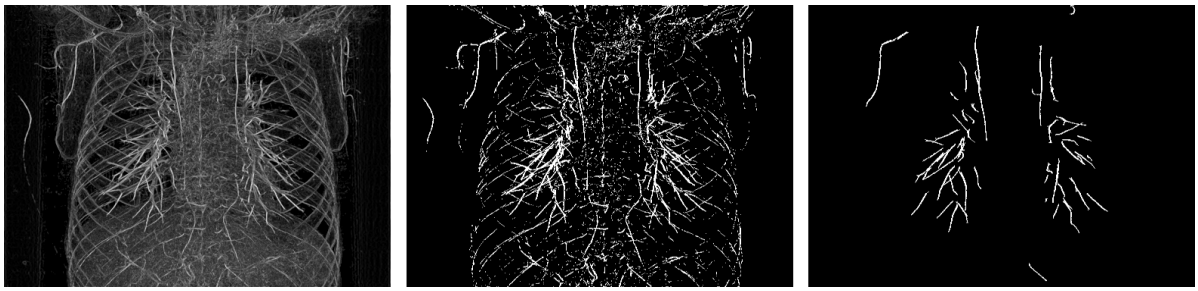


Figure 2.2: *Example of normal threshold versus Hysteresis Threshold. The left image shows the result from scale space vessel detection in a CTA data-set of a human lung by the Hessian-based enhancement. The middle image presents the result after applying a regular threshold operation, whereas the right image uses HT and is far less noisy, due to the adjustable two threshold values. Furthermore, it is noteworthy to mention that less noise leads to smoother centerlines. (Images created using the AngioVis Toolbox)*

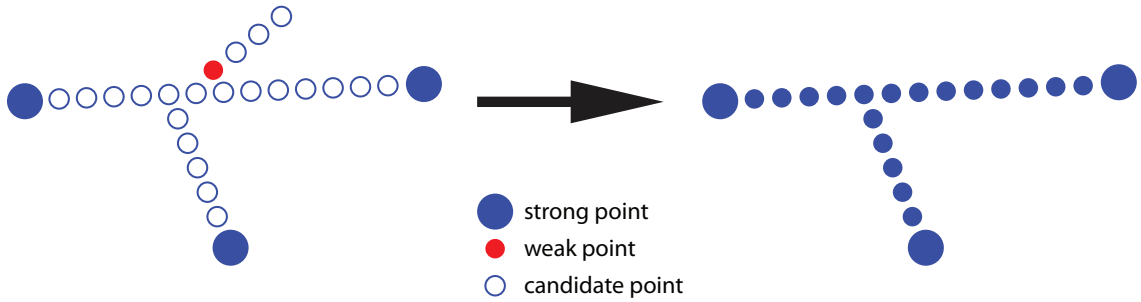


Figure 2.3: *Illustration of Hysteresis Threshold. Initially all points are marked as strong, weak or candidate. For each candidate point, it is recursively determined if a path to a strong point exists by visiting only candidate points on such a path.*

below it, called weak points (pixel in the 2D case and voxel in the 3D case, but since the principle of the threshold operation is the same in both cases, the general term point is used if not distinguished explicitly). The second threshold is taken as high value to determine so called strong points. Initially all strong points are marked with one and all weak points are marked with zero. All other points between those two threshold values are called candidate points. The algorithm mentioned in [9] follows for each candidate point the sequence of connected local maxima (candidate points), until a strong point is reached or no local maximum is accessible anymore. For each candidate point, a strong point can be reached, via propagation over its neighbors with either 4,8-neighborhood in the 2D case or 6,18,26-neighborhood in 3D. If a strong point is indeed reached, the candidate point gets marked as strong, otherwise with as weak, as illustrated in Fig. 2.3. Finally all points are classified, which means that no candidate point can remain.

It should be stated further, that the connectivity of the objects heavily depends on both chosen threshold parameters, since the better they are chosen the more accurate and smoother the resulting centerlines would be. Now the question may arise how we find the proper threshold values for the desired structures to extract. Choosing the same values for various different situations will not always suit best and manually or empirically chosen values will need a lot of time to be determined.

As mentioned in [6], several methods can be investigated to find suitable values for the low and high threshold values. One approach described therein is performed by so called hypothesis testing which utilizes the background conditional probability density function to determine the strong points by the means of a significance test, whereas the null hypothesis means that the current point belongs to background. In order to determine a weak point, the vessel class conditional probability density function is used, whereby the current point is hypothesized to correspond to a vessel.

The second method described in [6] is an empirical percentile based technique. As stated, the low threshold value separates high confidence background points from the rest, whereas the high threshold value separates high confidence object points from the rest. Consequently, the thresholds can be determined by utilizing percentiles of how much of the data is occupied by background and vessel points.

The simple threshold approach has one advantage in contrast to Hysteresis Threshold, namely it is a point operation, which means that it only needs to know the value of the current point and no neighborhood information. This leads to a straight forward implementation in a streaming fashion, whereas *HT* is a global operation and cannot be implemented in a streaming per slice fashion, because of the recursive propagation according to the candidate points. One way to implement a kind of streaming version of *HT* will be described in Section 3.1.4.

2.1.4 Skeletonization

Skeletons are useful in several domains and applications, such as visualization and image processing. In the medical domain the skeleton of a vessel is needed for simulating blood flow, virtual navigations through vessels or virtual endoscopy. Another field of application is surgery planning, since surgeons often need an accurate extraction of the specific region (vessels and organs) and therefore skeletons can be useful. Kinematics make use of skeletons, where joints and bones are the skeleton of a model and further used for animations. In morphing operations, skeletons are used for smooth transitions from one object into another. Even in object decomposition, as stated by Cornea et al. [7], skeletons are applied.

The medial axis of an object is defined in [7] as the locus of points that have at least two closest points on the boundary of the object. In the 3-dimensional case it is called medial surface. Furthermore, Cornea et al. [7] define a skeleton as the locus of centers of maximal inscribed open balls in 3D or disks in 2D. If $S \in \mathbb{R}^3$ is a 3-dimensional shape, an open ball with radius r is defined as $B_r(x) = \{x \in S | y \in \mathbb{R}^3, d(x,y) < r\}$. The distance between two points is computed by $d(x,y)$ and the ball is open since it does not include the boundary. Furthermore, an open ball is called maximal if it is not completely enclosed by another ball in S. Then the set of the centers of all maximal balls in S is called skeleton.

Amenta et al. [1] define a medial ball as a maximal empty ball, which is not fully contained in another empty ball. A ball is empty with respect to a surface S, if its interior contains no point of that surface. The medial axis transformation of the surface S is defined as the set of medial balls and, furthermore, the medial axis is made up of the set of centers of the medial balls of surface S.

In [7; 49] several properties of skeletons are described. Some of them are of significant importance for the method implemented in this thesis. Furthermore, some properties will lead to benefits regarding vessel visualization and Centerline Reformation, which will be discussed later. Most of the important properties are the following:

1. **Homotopic** (Topology preserving). Homotopic or homotopy equivalence means that two spaces can be transformed into one another by expanding, shrinking and bending operations [14].
2. **Invariant under isometric transformations**. If we have a transformation in which the length between two points is preserved, then the skeleton of the transformed object equals the transformed skeleton of the initial object. If the transformation is denoted by T, the skeleton by $skel$ and the object with O then $skel(T(O)) = T(skel(O))$ [7].
3. **Reconstruction**. Given the definition of the skeleton as the centers of the maximal inscribed balls, as mentioned by Cornea et al. [7], the reconstruction would be the union

of these maximal balls at each skeleton point. The radius of each maximal ball to the closest point on the boundary of the object is specify by a distance transform. Therefore the medial axis transform, as described in [1], is necessary in order to reconstruct the object. If the reconstruction is denoted with $rec(skeleton)$, the precise reconstruction would be $rec(skel(O)) = O$. As further mentioned in [7], the accurate reconstruction is generally not possible when the union of the maximal balls is used alone, since the skeleton is only a sub-set of the medial surface. Hence, the possibility to reconstruct an object from an abstraction (in our case the skeleton) can be an indicator of the quality of the abstraction. This means that, if the degree of reconstruction is low, the skeleton does not represent the object well.

4. **Thin**. Skeletons should be one-dimensional and one pixel thick. This means that every pixel has two neighbors, except for an end-point or a branch-point. This property is essential for the graph conversion as described later in Section 3.1.7 and for *CR* stated in Section 3.2.1.
5. **Centered**. In some application a perfectly centered skeleton is required. This is only possible if the skeleton lies on the medial surface as denoted by [7], since the latter is centered within the object.
6. **Connected**. This results from homotopy. If the object consists of one component then the skeleton is composed of one component too, since it preserves the topology.
7. **Robust**. The skeleton should be robust with regard to noise on the boundary of the object. However, Cornea et al. [7] state that a robust skeleton does not necessarily need to be centered. Telea and Vilanova [49] propose a single threshold to remove small branches resulting from noise.
8. **Smooth**. The skeleton should not contain abrupt changes.

A significant disadvantage of skeletons is the huge sensitivity regarding modifications of the boundary. Even very small changes can lead to large alterations of the skeleton. Fig. 2.4 shows an illustration of a skeleton and one when the boundary is slightly altered. Notice the large change of the skeleton in the right image. In many cases this is not desired, since the

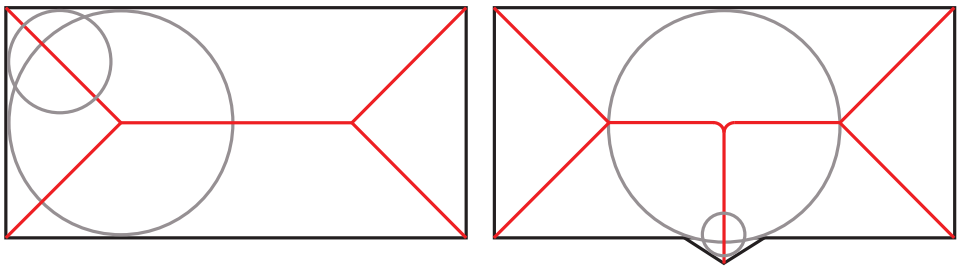


Figure 2.4: *Illustration of a skeleton (outlined in red) of a rectangle. Note that if the boundary is altered slightly the skeleton changes accordingly, as shown in the right image. The skeleton can be seen as a collection of centers of inscribed circles which have at least two points in common with the boundary of the object.*

skeleton should be robust under little changes, or even noise, of the boundary. For example, when the skeleton is used in virtual navigation it is not visually pleasing if the path is noisy. Also in vessel detection smooth centerlines of the vessels are expected and desired.

Since there exist various methods for computing skeletons from 2D or 3D objects, Cornea et al. [7] distinguish between the following categories of algorithms:

1. **Thinning and Boundary Propagation**. In this method boundary points are sequentially removed until no more change occurs. To determine if a point can be removed without altering the topology of the object, various constraints are investigated, as the Euler invariance or the notion of a simple point [26; 39].
2. **Using a Distance Field**. In this category a distance field is taken into account. Additionally some methods use this together with the thinning approach [39; 45; 7].
3. **Geometric Models**. As stated by Cornea et al. [7] this method applies to polygonal meshes or point sets. One approach uses Voronoi diagrams in order to get an approximation of the skeleton, whereas another approach uses a Reeb graph [35; 7].
4. **General-Field Functions**. In this approach a potential field function, an electrostatic field function or a visible repulsive force function is used. The skeleton is extracted by connecting the local extremes of the generated field [7].

Lee et al. [26] propose a thinning method, where points are subsequently peeled off to produce the skeleton of an object. To preserve topological thinning the notion of a *simple point* is introduced. Namely, a boundary point is simple if it can be removed without changing the topology of the underlying object, in particular the number of connected components. To determine whether a point is simple or not a method called *octree-labelling* is described, that only requires inspecting the neighborhood of the point, which makes it quite efficient. Furthermore, the *Euler characteristic* is used, which is a topological invariant and a number that describes the shape of a topological space, to additionally determine whether a point can be removed or not. But, as stated, using the Euler characteristic as the only deletion criterion is insufficient in some cases. The Euler characteristic is defined for a surface S, as $\chi(S) = \mathcal{O}(S) - \mathcal{H}(S) + \mathcal{C}(S)$, where $\mathcal{O}(S)$ equals the number of connected objects, $\mathcal{H}(S)$ is the number of holes and $\mathcal{C}(S)$ represents the number of cavities of the surface S. Hence, a boundary point can be removed without altering the topology of a surface, if it is Euler invariant, simple and no end-point. To prevent the reduction of skeletons to a single point while preserving topology of the object, end-points have to be encountered and preserved, since they are simple points and would thereby become removed, as additionally stated in [39]. Furthermore, a parallel approach is outlined by Lee et al. [26], which needs a special procedure, called *sequential re-checking* in order to obtain correct deletion of points.

Moreover, Pudney [39] describes a technique, which utilizes a distance field, namely *Distance-Ordered Homotopic Thinning (DOHT)*. Thinning is performed using a queue, leading to the advantage to process only those points whose deletability must be determined, in contrast to methods which sweep through the entire image every time. Moreover, as mentioned, the notion of centers of *maximal balls* is introduced, which are global features in contrast to local features. An example of a local feature is the preservation of the end-points

of a curve, which can possibly result from noise, since only the immediate neighborhood is inspected to determine such a point. Utilizing the distance field, the centers of maximal balls are identified as points p with distance values $d(p)$, for which $d(p) \neq d(q) - dist(p,q)$ with $\forall q \in N_{26}(p)$ denoting the 26-neighborhood. As stated in [39] the chamfer distance transform can be used to approximate the euclidean one for the computation of $dist(p,q)$, since it uses only integer values, as outlined in Equation 2.6. For further details about the chamfer distance transform and several issues about efficient implementations is referred to [23].

$$dist(p,q) = \begin{cases} 3 & \text{if only one coordinate changes (isothetic moves)} \\ 4 & \text{if two coordinate change} \\ 5 & \text{if three coordinate change} \end{cases} \tag{2.6}$$

The algorithm proposed in [39] works as follows. First, points are deleted if they are simple and not a center of a maximal ball with a distance value greater than a user-defined threshold. However, the skeleton produced does not satisfy the *thin* property, which leads to another pass, where points are deleted if they are simple and not an end-point. This produces a thin skeleton.

Telea and Vilanova [49] propose a fast and robust method for centerline extraction of objects by detecting the skeletons for each dimension separately and combining them in an explicit step. Furthermore, a thinning operation is applied followed by a reconnection step, since the resulting skeleton does not need to be connected. After this the centerline is ranked and pruned according to the ranking. Besides, preserving end-points can lead to many undesired small branches, for what reason a threshold is introduced in order to remove them.

Puig [40] describes a discrete medial axis transform for discrete objects. Differences between the continuous and the discrete case are investigated and mathematically described in detail, in order to produce a medial axis. Furthermore, the proposed algorithm uses a distance field and implements a seed strategy. The starting seed pixel is specified beforehand and belongs to the skeleton of the object. Initially specified for 2D images, a 3D extension is also presented and discussed.

Pascucci et al. [35] present an on-line technique of computing a special graph, namely a *Reeb graph*, which encodes the topology of the original shape. The topology of a compact manifold is captured by following the evolution of the level sets of a real-valued function, $\mathcal{F}$, defined on this manifold, as additionally described in [7]. Critical points of the function $\mathcal{F}$, where the gradient of $\mathcal{F}$ equals zero, are called nodes of the Reeb graph and further on, they are connected by arcs. The actual algorithm for creating the Reeb graph starts with processing the vertices and triangles in the order they are read. Initially, the Reeb graph is empty and every time a new vertex or triangle is processed, the Reeb graph is updated according to all simplices encountered so far. A new node is added for each new vertex and for all non-visited edges of each triangle, an arc is created. Furthermore, adding the interior of a triangle could lead to merging of paths that correspond to the edges of the respective triangle.

Another point of interest is a streaming implementation of skeletonization. Since skeletonization always requires the whole data volume to be in memory this can lead to a bottleneck. To prevent this, a streaming approach will be described and discussed in Section 3.1.6.

2.1.5 Streaming

As stated by Varchola et al. [52], streaming can be mainly divided into two different categories. The first is characterized by an infinite stream, as from an acquisition device used for multimedia. Furthermore, these streams end up in a presentation device, namely a monitor in the case of a video stream or a sound system in the case of an audio stream. The second category of streams belongs to visualization and data processing where it is understood as a technique to model operations on out-of-core data. They present a method where the smallest entity is a slice of the volumetric data.

In image processing one distinguishes between three types of operations, namely point operations, local operations and global operations. In a point operation the output voxel's value depends only on the value of the input voxel of the same position, whereas in local operations neighborhood information is taken into account, like in convolution with different kernel sizes. Another property of such kernels is called separability, which states that applying a 3D filter equals the subsequent application of a 1D filter for each coordinate (x, y, z respectively). Gaussian filtering is separable, for example. The last and most complex type of operations is the global one. In this case the region of influence of one input pixel cannot be definitely determined or is simply the whole volume, for example the distance transform operation. As further mentioned in [52], numerous global operations can be performed using a limited number of passes through the volume.

Law et al. [25] mention that parallel algorithms must be investigated, either distributed or multi-threaded, in order to provide an interactive visualization system which can deal with huge data. They bring up several properties concerning such systems, namely *minimize disk accesses* due to the large reading and writing times of disks, *cache* pieces of data in order to reduce the accessing time of frequently used data and *parallelism* in order to utilize multiple processors and increase computation performance. Another noteworthy property is *demand driven*, which means to process data only when it should be processed, in contrast to operate always on the whole data-set. Moreover, such visualization systems should be *hardware architecture independent* to provide a broad field of application.

Furthermore, a technique of partitioning a data volume into pieces, which are subsequently processed, is described in [25]. The main goal of the proposed method is the avoidance of memory swapping. Hence, three key principles for streaming architectures are mentioned, namely *data separability* which means to efficiently divide the data into pieces, *mappable input* which controls the data throughput of the streaming pipeline and *result invariance* which means that the result should not be related to the number of pieces and the execution mode (single or multi-threaded). Moreover, the second principle states which amount of input data is required to produce a given amount of the output. Result invariance facilitates the possibility to partition data into sub-pieces and perform subsequent operations on them, leading to the same result as if the operations would have been applied to the whole data. These three properties are essential for designing an abstract and extendable streaming architecture.

Pajarola [34] describes a streaming approach for point processing data supporting non-recursive local stream-operators, which are restricted to investigate only the immediate neighborhood. The main idea is to process data by sweeping a window, called active working set, over a data stream and perform operations only inside this window. Hence, this leads to an

efficient data processing resulting from coherent memory accesses as well as a significant reduction of occupied system memory. A local operator is defined as $\Phi(p_i)$ where p_i are points which are contained in the working set. Furthermore, a local operator $\Phi(p_i)$ is streamable if the computation does not need any recursive invocation on any neighbor points and before a point is passed to the following operator, it must be completely processed and updated in order to ensure the correct work flow.

Furthermore, Isenburg and Lindstrom [16] mention an approach for streaming triangle meshes. Since resolving triangle to vertex references needs a coherent underlying mesh layout to ensure optimal cache utilization, Isenburg and Lindstrom [16] investigate a special streaming format to support a huge number of triangles. Their basic idea is to store the data in an interleaved fashion, where vertices become active when introduced and get inactive when finalized (when they are not used by any subsequent triangle anymore). They use spectral sequencing for arranging the vertex and triangle arrays to obtain a good coherence of the layout. The reconstruction of the mesh is performed using a window of a specific size, which sweeps over the mesh of full resolution. In contrast to rendering sequences, where local coherence plays an important role, it has a smaller impact in streaming, but global coherence of the layout is much more crucial.

As additionally mentioned in [52; 34; 16], streaming has the inherent advantage of reducing memory consumption. This is particularly necessary when engaging huge data volumes or performing a lot of operations on volumetric data in parallel, since the system memory is limited. Especially the techniques described in [52; 34] scale well when modelling large and complex networks of operations, as the vessel scale detection pipeline.

2.2 Vessel Visualization

Visualization in medicine has become a very important field of interest for many people of various professions. From the medical doctor's point of view, visualization is a powerful tool which facilitates the doctor's diagnosis of diseases and provides the doctor with more insight and information. In contrast, computer scientists might think that the most visually pleasing visualization will suit a medical doctor most, which is often a misunderstanding of the terminology and requirements. In most cases very simple techniques such as slices are more convenient for a medical doctor than 3D volume rendering approaches, no matter how visually pleasing they may be.

The main goal of visualization, as stated in [38], is to bring information to medical doctors in the way they can utilize it. Hence, it is very important for a computer scientist to know the requirements of a medical doctor and of course vice versa. A medical doctor should know at least, how to define the requirements in an understandable way to clarify what exactly should be the aim of a technique or tool.

Straka [48] distinguishes between two categories of visualization. The first category is called *scientific visualization* with the main focus on photorealistic visualization of a scene under consideration of the spatial relationship between objects therein. The second class is called *information visualization* which primarily deals with visualizing information of any kind without a spatial relationship, for example charts, histograms, etc. In this category intuitive visualization of the desired information is of greater importance than photorealism and certain features can be depicted and emphasized using illustrative or non-photorealistic visualization techniques.

Medical visualization lies between those two categories. On the one hand realism is required but on the other hand it goes side by side with visualizing the desired information in a meaningful way. One has to find the most suitable solution for a medical doctor, which can be a challenging task.

In this thesis the main focus lies on vessel visualization. Although there are many ways how to visualize them, not all of them are suited for diagnostic purposes in radiology. Here we specialize on reformations to detect and visualize possible calcifications of vessels. This is a highly important and established section, since, for example, planning a stent surgery needs to be accurate. In this case, visualization serves as a tool to support the medical doctor in making decisions and planning the surgery. However, it should never be used as the only tool without considering additional opinions.

As already outlined, locating pulmonary embolisms is a very time consuming task which furthermore requires a lot of skill of the doctor to classify them correctly. Visualization techniques would provide a real advantage, since they would significantly support the doctors in localization and decision making.

2.2.1 Volume Visualization

Since volumetric data consists of several slices resulting from the acquisition method, the simplest approach is to present each slice separately. These slices can be independently inspected by medical doctors to detect possible suspicions or to use them as support for surgery planning.

However, since the number of slices grew rapidly with developing acquisition technologies, this method can become extremely time consuming, since a vast number of slices has to be investigated. Therefore, techniques which render either the whole volume or specific regions have been developed and established.

A straightforward and commonly used method is *Maximum Intensity Projection (MIP)*, where rays are shot through the data volume, either parallel to the viewing direction when encountering orthogonal projection or along the view vector, determined by the vector through the eye-point and each pixel of the image plane, when coming to perspective projection. At each sampling position along the rays, only the maximum value is taken and displayed. On the one hand performance is a significant advantage of this method, since it is very fast to compute, but on the other hand it has one drawback, namely depth perception is lost to a noticeable degree. Although this method is quite simple, it is widely spread and applied, since a quick overview of the data is always required and essential.

Levoy [28] mentions a method called *Direct Volume Rendering (DVR)*. A color and a transparency value are assigned to each voxel regarding their density [48]. These properties of the voxels get accumulated along each ray and the final value is presented in the image. Furthermore, the surface normal vector at each voxel is utilized for computing additional lighting and shading information. The notion of transfer functions was introduced in order to specify colors and transparency for a specific range of densities of the volumetric data. Utilizing transfer functions, structures of interest such as tissue or bone can be revealed and viewed separately or together. For a detailed description of several types of transfer functions is referred to [38].

Bruckner and Gröller [5] describe a technique, that is called *Maximum Intensity Difference Accumulation (MIDA)*, which provides the user with increased depth cues, in contrast to the common *MIP* method. The basic principle is to emphasize the changes of maxima along each ray, particularly from a low to a high value, by increasing their contribution to the final result. This method is capable of performing front to back compositing very fast. As it is a hybrid technique which lies between *MIP* and *DVR*, it has the inherent advantage, in contrast to *DVR*, that it does not require an explicit specification of a transfer function. Moreover, depending on how a specific parameter is chosen, the appearance of the result can be adjusted between *MIP* or *DVR*.

Another approach of volume rendering are surface rendering techniques, for example iso-surface rendering. Given an iso-value, the corresponding surface is extracted from the volumetric data-set and a polygonal mesh is reconstructed in order to render the surface. Several techniques for polygonal mesh reconstruction have been investigated. The usual approach is the *marching cubes* method, which is described in [38; 30] in detail. An advantage of the surface rendering approach is the possibility to utilize the *GPU*. On the one hand, the well established and advanced illumination and rendering techniques can be applied. On the other hand, a major drawback of the approach is that no interior of the underlying volumetric data is displayed, which is necessary, for example, in embolism detection. Hence, this types of visualizations are not used for inspecting the interior of regions.

2.2.2 Reformation

Reformation can be seen as a visualization technique with the essence in displaying eventually interpolated values over a cut surface. This surface can be specified by the medical doctor in order to get insights into specific regions of possible suspicions. Additionally, it can be useful in surgery planning and it can assist a medical doctor in decision making. Since one plane is not satisfying in many cases, planar reformation is extended to specify many planes, namely multi-planar reformation. This method evolved from using orthogonal planes to oblique cut planes later.

One disadvantage of planar reformation is the specification of *planar* cut planes, which cannot be modelled along arbitrary curves in space, such as tubular structures, like vessels for example. Since the lumen of tubular structures is a very desired visualization tool for detecting and classifying calcifications, another more flexible method must be investigated. This technique should be accurate and cut the structure along a specified centerline. Even interactive zooming and rotating along the centerlines should be supported to give the medical doctor the possibility to explore the tubular structures from several different points of view.

2.2.3 Curved Planar Reformation

To overcome the shortcomings of planar cut planes, Kanitsar et al. [18] proposes a method called *Curved Planar Reformation (CPR)*, which reveals the lumen of a tubular structure by *sampling* the volume data along the centerline of the object parallel to the view plane. The goal of *CPR* is to present the lumen of the entire tubular structure within one image. This feature can be of great use in the medical field of angiography to inspect vessels. For example, in the case of atherosclerosis, heavy plaque can be detected and inspected using *CPR* and further, an eventual surgery for placing a stent or a bypass can be planned utilizing this technique.

A case study of *CPR* is given by Kanitsar et al. [20] and therein they present *CPR* as a possible visualization technique of *CTA* data. Further on they describe the centerline extraction as a graph theoretical problem and introduce the notion of a *cost-function* which determines the likelihood of a voxel to be inside a vessel. Subsequently, they reduce the problem to a shortest path problem according to a user selected start and end-point. As a side effect of this approach the centerline does not necessarily need to be centered, as stated in [20]. To overcome this, a vessel-center approximation algorithm is applied afterwards.

In [18; 17] is distinguished between three different categories or types of *CPR* techniques according to geometric properties, which are outlined in Table 2.2 and illustrated in Fig. 2.5. Whereas projected *CPR* preserves the spatial orientation which is useful for exploration, it

Table 2.2: *Geometric properties of the CPR methods*

CPR	Spatial perception	Isometry	Occlusion
Projected	high	–	maybe
Stretched	medium	yes	–
Straightened	low	yes	–

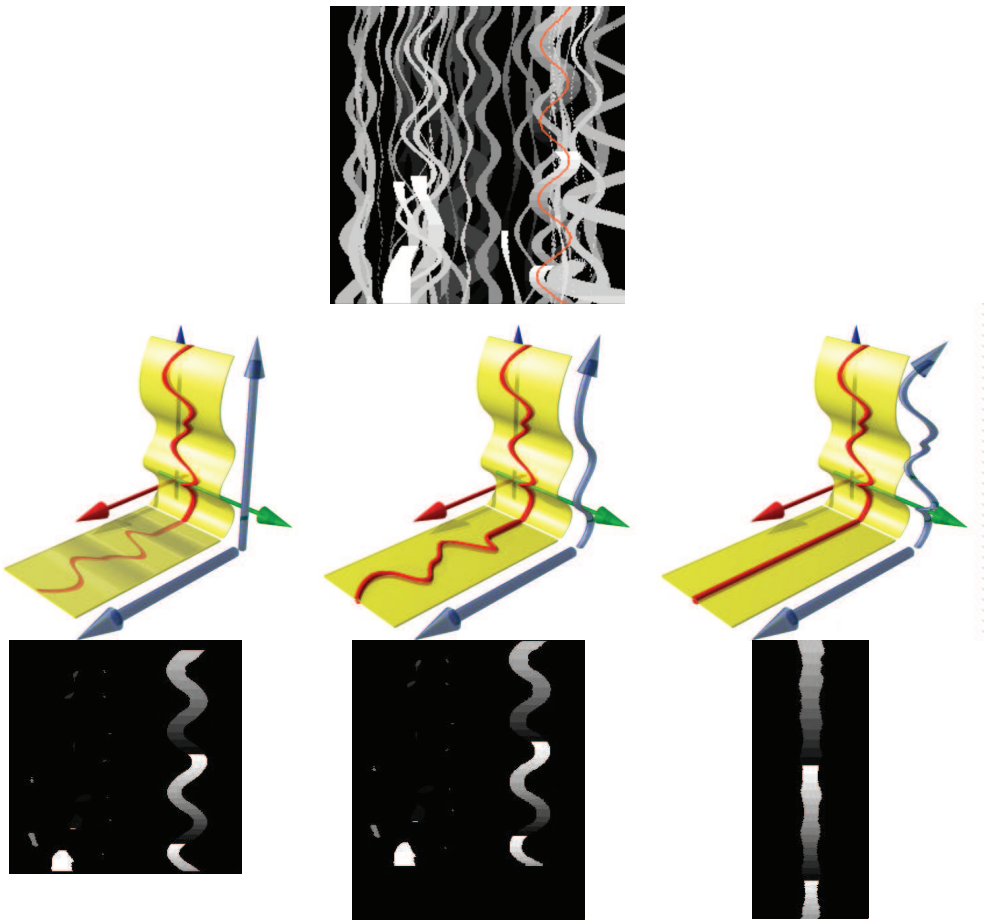


Figure 2.5: *Illustration of the different CPR techniques. The top image is the MIP of the artificial data-set, further highlighting the selected helix. The second row shows, from left to right, the projected CPR, the stretched CPR and the straightened CPR. The third row presents an example of each type of CPR. (Image sources: second row Kanitsar et al. [18], first and third row created using the AngioVis Toolbox)*

lacks isometry, which is essential in planning stent surgeries due to preservation of length. In the following sections each type will be discussed in greater detail.

Kanitsar et al. [18] define a *vector-of-interest*, which is in some cases parallel to the xy-plane. Furthermore, if the vector-of-interest moves along the centerline, it defines a line, the so called *line-of-interest*. The following methods differ according to how the line-of-interest is projected into the image plane and how the volume data is sampled along this line with regard to the vector-of-interest.

2.2.3.1 Projected CPR

In this case each point of the centerline, the line-of-interest, is projected to the corresponding line in the image which is parallel to the viewing plane, as shown in the left image of the second row in Fig. 2.5. The vector-of-interest is parallel to the viewing plane. Furthermore,

if the up-vector of the camera is parallel to the z-axis then the vector-of-interest directly maps to each line of the image. On the one hand this method preserves the spatial relations due to the parallel projection, but on the other hand, as stated in [18; 17], isometry is not preserved as an effect of the distortion of the central axis' length. Another disadvantage is the way of sampling the volume data. For vertically aligned structures it is sufficient if the vector of interest is orthogonal to the central axis, which leads to artifacts in the case of a tubular structure being parallel to the vector-of-interest. To overcome this, we can sample according to a circle, but this leads again to artifacts, since wrong values are displayed, because values will be overwritten in the image, if we take the samples closer to the camera. In the case of peripheral vascular structures, horizontal vessels only rarely occur, but in the general case with arbitrarily spatially orientated vessels, we need a more general method of sampling along the centerline.

Furthermore, since the whole image is sampled along the line-of-interest and the vector-of-interest, surrounding parts are sampled too. This leads to possible undesired artifacts, because the data outside the tubular structure is sampled and may overlap with other tubular structures. Additionally, this may lead to performance issues if the volume data is huge, because of sampling a lot of unnecessary data. Moreover, if we want to have a specific context rendered around the structure, for example using *MIP* or *MIDA*, of the remaining volume, this may be difficult to establish.

2.2.3.2 Stretched CPR

In this method, the line-of-interest is curved in one dimension as stated in [18; 17]. All points of the centerline are mapped to the image in such a way, that the distance between two consecutive points is preserved, as illustrated in the central image of the second row in Fig. 2.5. Hence, isometry is maintained and as a result, the image size depends on the degree of curvature of the centerline. The higher the curvature, the smaller the image and vice versa. For a detailed description see [18; 17].

One drawback of this approach is that spatial orientation is not fully preserved, since each point of the centerline is mapped to the image. In contrast to the projected *CPR*, this method is utilized in planning stent surgeries since the length and the placement of the stent need to be known accurately beforehand [18; 17].

2.2.3.3 Straightened CPR

This type of *CPR* straightens the tubular structure completely along the centerline, as seen in the right image of the second row in Fig. 2.5. The height of the image corresponds to the complete length of the centerline, as mentioned in [18].

On the one hand, a serious disadvantage of this method is the lack of spatial orientation, since the centerline of the tubular structure is completely enrolled. On the other hand this reveals the advantage of preserving isometry and gives an overview of diameter changes in the tubular structure. Furthermore, as stated by [17], this method is highly vulnerable to noise of the centerline.

2.2.4 Multipath Curved Planar Reformation

Since the visualization of one tubular structure out of a tree is often not sufficient for diagnostic purposes, [18; 19; 17] use the *CPR* technique multiple times to support branched structures, namely *Multipath Curved Planar Reformation (mpCPR)*. It should be anticipated that the segments of the tree should not overlap each other in order to perceive the correct depth relationship, since this method was initially designed to visualize peripheral arteries of lower extremities, as shown in the right image of Fig. 2.6.

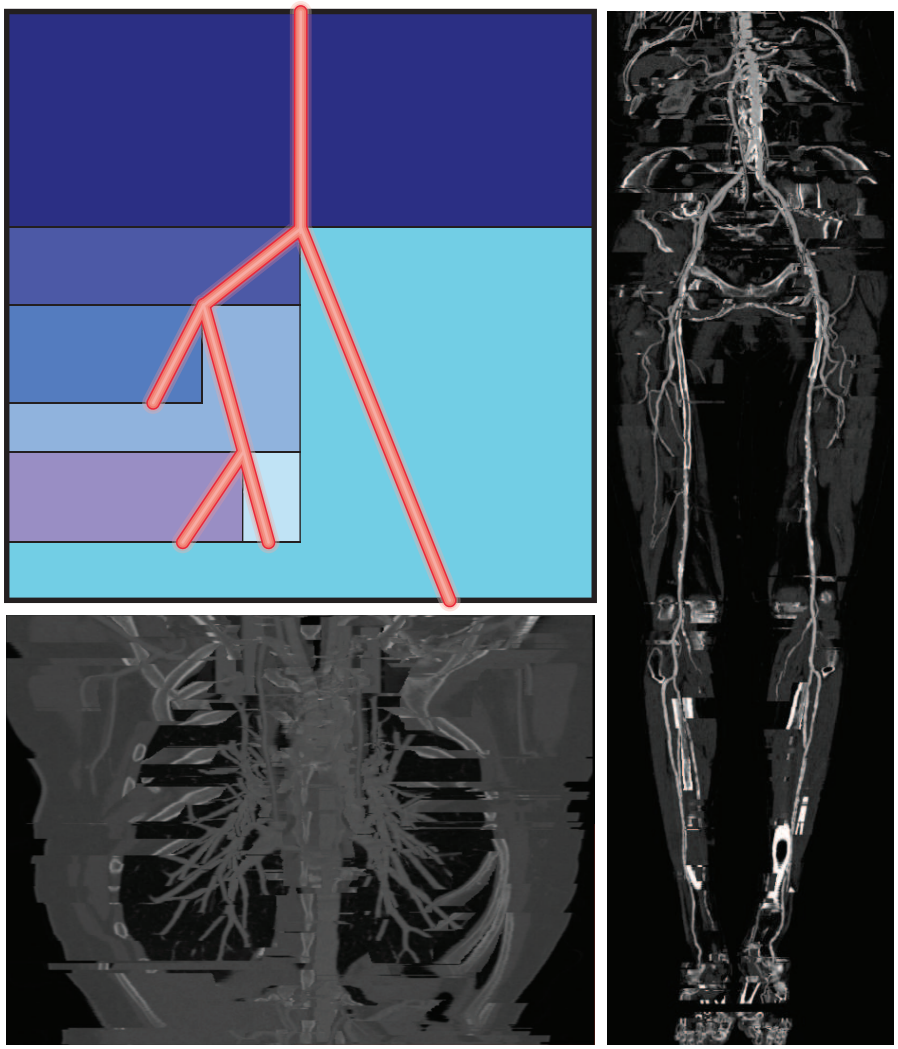


Figure 2.6: *Illustration of mpCPR. Note how the upper left image is partitioned for each branch of the tree. Furthermore, for all parts the CPR is computed and finally they are composed into the resulting image. The lower left image shows the mpCPR of a CTA data-set of a human lung and the right image displays the mpCPR of the lower extremities of a human CTA data-set. (Lower left and right image created using the AngioVis Toolbox)*

Basically, the method works as described in [17]. The image is divided according to the branches of the vessel tree. For each tree segment the *CPR* is computed and finally the individual parts are merged together to produce the final output. One drawback of this method is that the neighboring parts do not match at the borders, since, as already mentioned, *CPR* always samples the whole image and in this case the whole image part. Furthermore, if instead of the surrounding parts *MIP* or *MIDA* is desired, special effort has to be taken into account.

Visualizing arbitrary not connected vessels turns out to be a problem with this method, since the prerequisite of a connected vessel tree for partitioning the image is violated and due to the lack of depth perception of overlapping vessels, the resulting image would not be meaningful, as displayed in the lower left and right image of Fig. 2.6.

3. Method

3.1 Vessel Extraction

In the following sections each step of the vessel detection pipeline will be discussed and our own or adapted implementations will be presented and described in detail. In each step several pitfalls and problems will be pointed out, as well as already used or possible optimization techniques will be additionally mentioned. Moreover, for every step an eventual streaming implementation will be presented, and the advantages and disadvantages will be discussed.

As the main goal of the pipeline is automation, every step must operate more or less with a minimum of user interaction. Since it is hard to estimate which parameters are necessary, this will be discussed for every step. Another goal is stability and, of course, fail-safety of the pipeline, since errors cannot be addressed and solved by non-domain experts, such as medical doctors, since they do not know about the internals of the implementation. Hence, errors should be avoided as much as possible, but if occurring, rolling back to a previous error free state should be possible. Moreover, for every step, as future aspect, optimizations will be discussed and investigated.

More detailed, everything is programmed in C++ utilizing the *F3D* library and the *AVT*. As already mentioned, each operation of the pipeline is implemented as a single filter, subsequently called task, which leads to the inherent possibility to model several execution assemblies, described as the notion of the task scheduler in the following.

3.1.1 Task Scheduler

Before going into detail of every step of the vessel enhancement pipeline, the abstraction for modelling the processing pipeline will be described, namely the task scheduler. Each step is modelled as an independent task with only a sparse number of methods to support the basic operations such as initializing a task, running a task and of course handling task finishing signals. Further on, each task stores a pointer to an optional parent task to have the possibility to create tree-like structures of tasks and to notify the parent of eventual signals. To distinguish between tasks and subtask, and to support progression handling, a dedicated method is added to a task to signal that a subtask is done.

In order to model assemblies of tasks, a scheduler needs to be implemented. It should be anticipated, that the task scheduler is implemented in the *AVT* program to support pipeline integration with the user interface. Basically, there are two different types of task schedulers, namely a parallel and a sequential one. The former executes a list of tasks in parallel, which

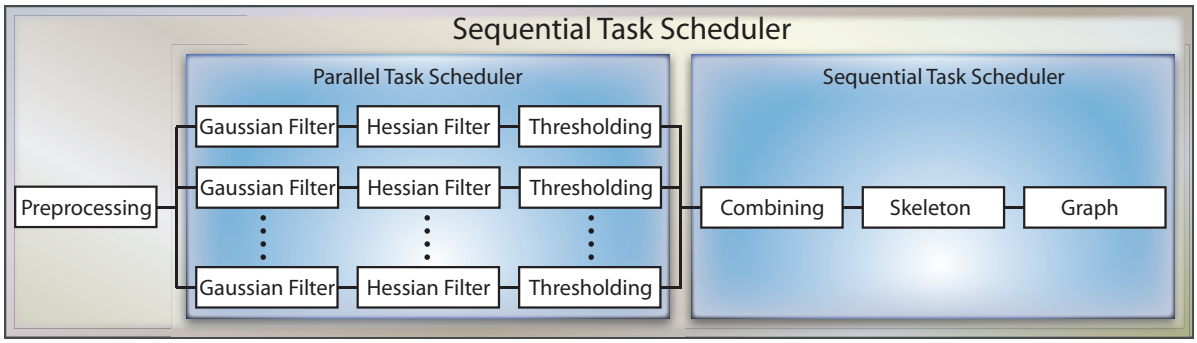


Figure 3.1: *Illustration of task schedulers used in order to model the vessel detection pipelines. Globally, a sequential task scheduler is needed to ensure the correct workflow of the pipeline. The only parallel scheduler is applied for the scale space vessel detection, since here every triple of operations is executed in parallel.*

means that every task in the list is started immediately, whereas the latter starts the tasks in the list consecutively. Of course, synchronization is taken into account by the schedulers, to ensure the correct order and update of the progress whenever a subtask finishes within a task scheduler. To support modelling more complex execution systems, the schedulers are tasks themselves, which allows to model a scheduler of different schedulers.

Supporting those two different types of schedulers, task chains, grids (as the scale space vessel detection) and sequences can be modelled without any further concerns. Furthermore, whenever a task is encountered, a scheduler can be added instead too. Utilizing this flexibility, the vessel detection pipeline can be modelled as illustrated in Fig. 3.1.

Since each operation is modelled within a separate filter, another type of task must be implemented, namely a lower-level one, which utilizes the underlying operation systems inter-process communication. In our implementation the Nokia QT (see `http://qt.nokia.com/` for details) is used together with their implementation of a process. The processes can be connected together by their standard input and output and furthermore each process can be specifically controlled by their corresponding command-line parameters. Every time a process is finished, an event is triggered and handled by the corresponding underlying task. Such events are mainly the finishing of those processes, whereas only the last process signals that the task is done, the others are just seen as subtasks. This modelling allows to handle accurate progression in the current pipeline and, of course, eventual aborting by terminating all running tasks by the user.

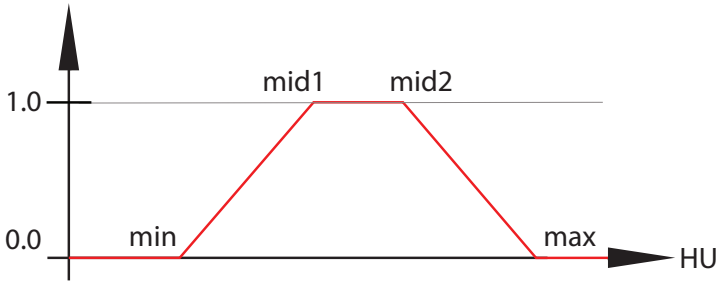
3.1.2 Preprocessing

To prepare the volumetric data for the vessel detection pipeline, a preprocessing is necessary to adjust the density values. Since the data is internally, in the *AVT*, represented in an integer format (16 bits, unsigned short), it needs to be converted to a floating-point representation, utilizing the *F3D* binary *f3d2f3d*, since the subsequent tasks operate on floating-point data.

Furthermore, density values can be remapped to initially emphasize vessels, since if the data has been acquired using *CTA* the density values are normalized in *Hounsfield Units (HU)*. Hence, bones and other undesired and clearly distinguishable types of tissue can be initially suppressed. In the next operation, a trapezoid function is applied to remap the Hounsfield Units to vessel probabilities by using the binary *f3dramp*. This special step was initially mentioned by Straka [48], since the Hessian filter should not be applied to the densities directly, because the vessels are not always the brightest structures and, furthermore, vessels have a high density variability resulting from contrast agents or calcifications which makes the vessel enhancement irregular. The trapezoid function is defined by

$$f(x) = \begin{cases} 0 & \text{if } x \leq min \\ (x - min)/(mid1 - min) & \text{if } x < mid1 \\ 1 & \text{if } x < mid2 \\ (max - x)/(max - mid2) & \text{if } x < max \\ 0 & \text{if } x \geq max \end{cases} . \tag{3.1}$$

Input are the *Hounsfield Units (HU)* of the volumetric data and output is the probability of the vessels, which lies between zero and one. An outline of the function and the used values is shown in Fig. 3.2.



Border	HU
min	1100
mid1	1200
mid2	1600
max	1700

Figure 3.2: *Illustration of the remapping trapezoid function. The left image shows the trapezoid function which is used for remapping vessel densities to vessel probabilities, whereas the right image shows the empirically determined values used for remapping.*

All the steps described above are implemented in the streaming fashion of the *F3D* library, operating on a slice basis, which allows to execute them in parallel. Moreover, this leads to an increase in computation power utilization, due to the fact, that no processes stall until the previous one has finished. Additionally, a possibly huge memory consumption due to large data-sets is reduced, since only a small number of slices is stored in the memory during the operations.

3.1.3 Detection

After the data has been preprocessed, it is ready for vessel scale space detection. As already mentioned this is performed by applying a Gaussian filter followed by a Hessian filter. Both have the same sigma value as the scale input parameter, which corresponds to the size of the vessels to be detected.

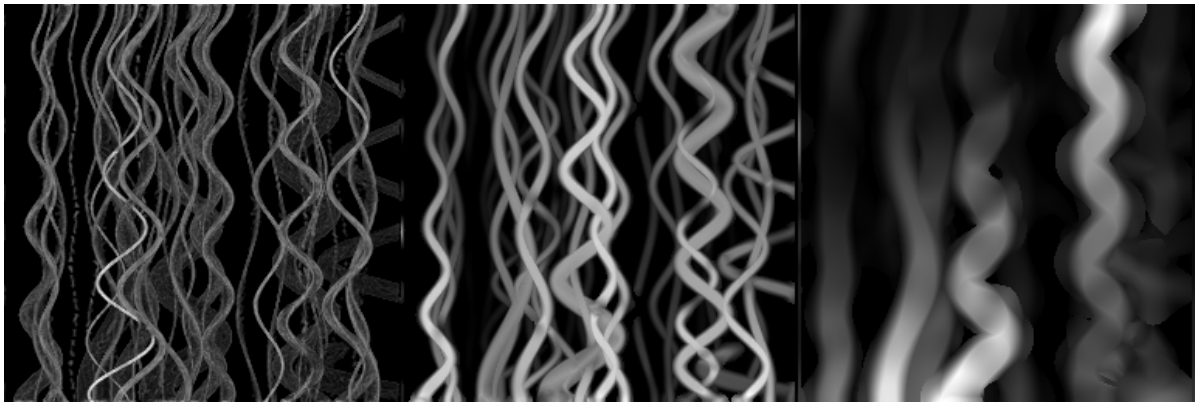


Figure 3.3: *Example of a Hessian filter of an artificial data-set. Note how with increasing scale, from left to right, larger tubular structures are emphasized. (Images created using the AngioVis Toolbox)*

Fig. 3.3 shows the vessel enhanced data of an artificial data-set, which contains several various sized vessels. Note how the vessels corresponding to each scale are enhanced, whereas the rest ones are hardly noticeable. Since this step requires some computation time, it is desirable to execute the sequence of Gaussian and Hessian filters in parallel, for each specific user defined scale value. Afterwards, a threshold operation is applied to each filtered volume in order to pass only the desired and enhanced vessels. After that, they are combined into one volume which is used for further processing, as explained in Section 3.1.5.

Since the input volumetric data can be huge, it is not feasible to load the whole data volume into the system memory. Imagine that if we wanted to compute ten scales, whereas the input data has about 1 GB, we would need 10 GB memory at the same time. This is hard to establish even on modern computer architectures or it requires extensive memory swapping, which leads to a significant performance impact. Since each operation stores the whole volumetric data in the system memory, it produces the result at once, which leads to a bad computation power utilization, because only one process is working at a time. Thus, step by step processing of the triple Gaussian, Hessian and threshold will be performed, which contradicts the initial design principle of each operation. Hence, the memory consumption without utilizing streaming approaches depends on the size of the data volume, the number of operations and the structure according to which these operations are organized. To overcome the memory bottleneck, we utilize the separability property of the Gaussian and Hessian filters by operating on a per slice basis or in this case on a slab of slices, since some neighborhood information has to be taken into account. If a single slice has been processed in the sense that it is not further required by the slab, it is immediately written to the output, the standard output in most cases. This leads to optimal workload since the Hessian filter can start working if the first slice is processed by the Gaussian filter. Hence, only the absolutely necessary slices are stored in the memory, which reduces the overall memory consumption a lot. The memory used is no longer dependent on the size of the data volume more than on the number of slices needed by the Gaussian or Hessian algorithm, which is a great improvement.

3.1.4 Thresholding

Hysteresis Threshold is adapted as described in the related work Section 2.1.3. The aim is to delete candidates which are not connected to strong points (pixels in 2D, voxels in 3D). Hysteresis Threshold is a global operation, which means that repeated recursive propagation through the whole data volume is necessary. For that reason it is difficult to establish a streaming implementation. The usual approaches to sweep a window of a specific number of slices through the volume along one direction would not lead to a correct result, as illustrated in Fig. 3.4, since the wave needs an unpredictable amount of back and forth movement until it will be completely classified. The point is that it is unpredictable, since all tubular structures of the image cannot be investigated in order to determine the amount of movement needed. On this account a solution is needed to simulate the recursive propagation in a streaming fashion.

A straight forward solution would be to read and write points of arbitrary slices. In this case the recursive approach can be used, as outlined by the *FollowEdge* method in Algorithm 1. Since strong points are spread to their neighbors through the whole data volume, this will soon lead to an excessive read/write overhead, resulting in infeasibility, because writing operations tend to be quite slow. For this reason a more sophisticated approach is needed.

Another solution is to do as few writing operations as possible. To ensure this we first need to remove the recursion of the *FollowEdge* method, which makes the method obsolete at all. Algorithm 2 shows the resulting streaming implementation of Hysteresis Threshold. In

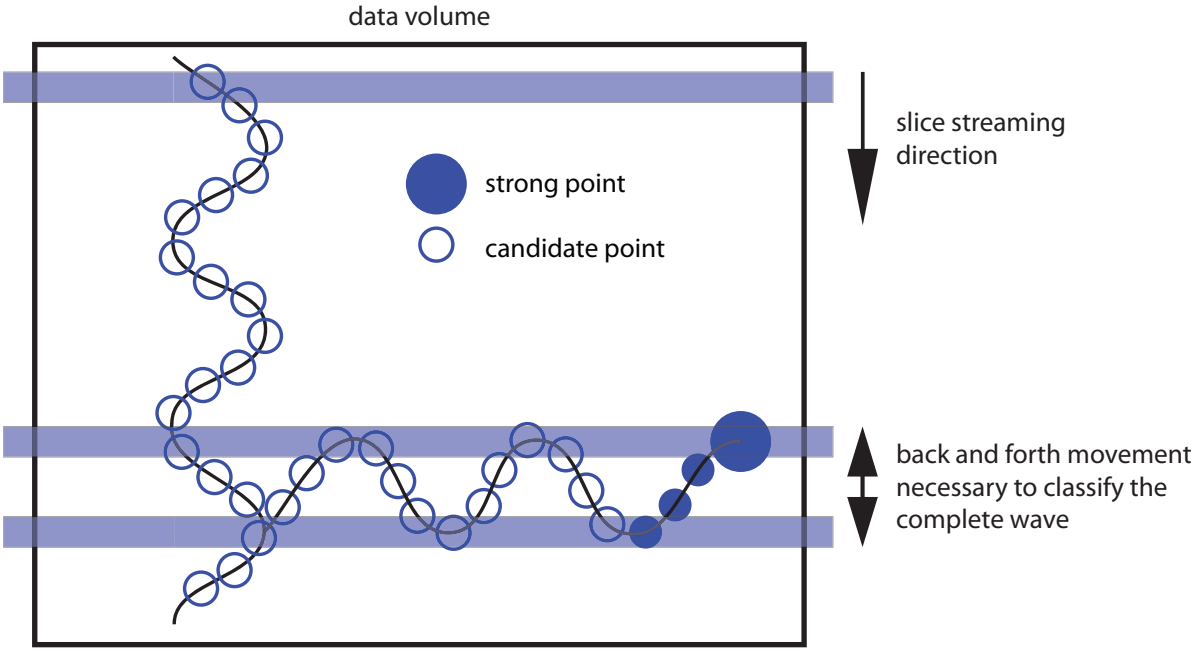


Figure 3.4: *Illustration of a problem for a streaming implementation of HT. Since only one strong point exists, several back and forth movements are necessary in order to classify every candidate point.*

the first pass we classify each point as strong, weak or candidate. In the second pass the slice offsets in the stream are determined. The main pass is the third pass, during which we loop through all slices, and for each slice all strong points are stored in a list for propagation, called *Local Propagation List (LPL)*. Now, for every strong point in the *LPL* an additional list of candidates to propagate is used, called the *Candidate Point List (CPL)*, which initially contains just the current strong point of the *LPL*. Finally, recursion is simulated by removing the first point of *CPL* and append it to another list, called the *Global Propagated Point List (GPPL)*. Then the neighbors of the current point are added to the *CPL* if and only if they are not already contained in either the *CPL* or the *GPPL*. These constraints are necessary to avoid infinite looping since we always operate on data which is not updated, because no writing operation has been performed yet.

Every time the *CPL* becomes empty, the next strong point in the *LPL* is removed and propagation is simulated again. This is done until the *LPL* of the current slice becomes empty. Then we proceed to the next slice. Mind that no writing operation has been performed yet, only reads have been done, which leads to a significant improvement in performance. Furthermore, what is noteworthy is that the usage of the *LPL* preserves integrity along the path since the propagation is done in a depth first search fashion. This may also lead to an opportunity to remove reading operations using a slab of slices instead of always one. Imagine on the other hand if we added all strong points of a slice immediately to the *CPL*, then we might have to read different parts of the data volume every time, reducing or even nullifying the usage of a slab, since paths may vary.

Since all strong points resulting from the propagation are stored in the *GPPL* it needs to be cleared at regular intervals, otherwise an out-of-memory exception might occur for large data volumes. This step is called update pass. We just add a threshold according to the number of points contained in the list and if exceeded, the whole data volume is looped through and the strong points contained in the *GPPL* are written. Generally the streaming approach is significantly slower than the one storing the whole volume in the system memory, since a lot of additional reading and writing operations need to be done, but with some investigations the streaming approach can be of use too.

Since we always operate on a per slice basis, the *GPPL* needs special effort to further optimize the searching whether a point has already been propagated. Even during the update pass, it should be investigated if a slice contains any strong points at all, since it might be skipped, resulting in a further reduction of unnecessary reading/writing operations. On this account, an array of lists for the *GPPL* is used, where the index of the array is the number of the current slice. Thus searching whether a point has been propagated is reduced to searching in the slice related to the point. Moreover, in the update pass slices can be skipped very fast by only checking whether the corresponding list is empty. All these optimizations lead to an increase in performance of the streaming implementation.

As a future aspect, a slab can be used for propagation instead of reading one slice all the time. Furthermore, a more sophisticated structure for searching whether a point is in the *LPL* may lead to a performance gain. Maybe the same structure as for the *GPPL* should be applied also for the *LPL*, but this needs to be investigated further.

3.1.5 Combining

In this step all volumes resulting from the scale space detection are combined into one, since skeletonization can only operate on one volume. For every voxel of the resulting data volume, the maximum scale value is taken and written into an optional data volume, called the sigma-volume, as it contains necessary information for further processing, such as the width or radius of the vessel in the corresponding voxel. The resulting volume is built by the union of the Hysteresis Threshold of the volumes from the scale-space detection. The streaming implementation is outlined in Algorithm 3, where out_p denotes the value of a voxel in the output file and $sigma_p$ the sigma value (scale value) of the voxel in the sigma-volume.

Since every volume is just read once and needs to be processed only along one axis, this part is suited for streaming. Simultaneously a slice of every scale volume is read and the maximum sigma value of each voxel of these slices is determined. Furthermore, the value of the voxel of the scale volumes, which has the maximum sigma, is written to the output volume. Usually this value is one, resulting from threshold operation. Fig. 3.5 shows the result of the combination of all volumes resulting from the scale space analysis.

Another noteworthy aspect concerning streaming is, that the combining would not be realizable storing all the volumes to combine in the system memory. Since the data is accessed uncompressed, each volume will use, dependent on the extent of the volume, about 1 GB. If ten scales have been processed, the combination step will need 10 GB of system memory, which is not tractable, since swapping will reduce performance far too much.

Figure 3.5: *Example of combining several thresholded volumes of a CTA data-set of a human lung. The images in the first row show the thresholded results from the multi scale vessel detection. The left image of the second row presents the combined volume, whereas the middle image shows the corresponding sigma volume. Note that brighter areas mean larger vessels, since the sigma value is higher. The right image displays the resulting skeleton. (Images created using the AngioVis Toolbox)*

3.1.6 Skeletonization

Advancing from the combined volume of the thresholded scale-space detection, the centerlines must be determined in order to generate the vessel tree. The main challenge for this step was to satisfy the *robustness* property, which is the most important one, since the vessel detection must be robust with regard to noise. Therefore, two methods have been implemented, where the first one is described by Lee et al. [26] and the second one is mentioned by Pudney [39]. The resulting difference of the two methods is presented in Fig. 3.6.

In the first applied implementation, both constraints, the Euler invariance and the notion of a simple point are used, as mentioned and described earlier in Section 2.1.4. Our implementation is derived and adapted from the *Insight Toolkit (ITK)* skeletonization algorithm, namely *BinaryThinning3D*. Basically, this skeletonization method peels off a point for every border direction by checking if the point is not an end-point, whether the point is Euler-invariant and finally if the point is a simple point, then it can be removed or added for sequential re-checking in the case of parallel peeling. This sequence of steps is repeated until no changes occur. The output is a one voxel thin centerline in 26-connectivity, which is utilized in the following step in the vessel detection pipeline, namely the graph conversion.

The second implemented method, *Distance-Ordered Homotopic Thinning (DOHT)*, is described in [39]. The main difference to the previously mentioned method is the use of a distance field and a queue for the points whose deletability needs to be investigated. In contrast to the method proposed by Lee et al. [26], which is a one-pass algorithm, *DOHT* is a two-pass algorithm. In the first pass, all points which are simple and not a center of a maximal ball are deleted. The centers of the maximal balls are determined applying a user defined threshold according to their distance. Since this produces a thick centerline, a second pass is needed in order to produce a thin centerline by deleting all points which are simple and not end-points.

As presented in Fig. 3.6, the method proposed by Lee et al. [26] produces less noisy center-

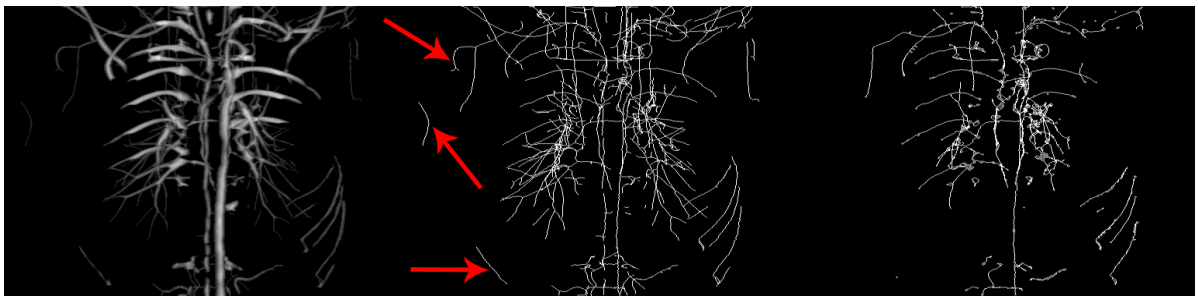


Figure 3.6: *Comparison of the two implemented skeletonization methods. The left image presents the chamfer distance transform of the pulmonary data-set computed in vessel insides. The middle image shows the technique described by Lee et al. [26], whereas the right image presents the method by Pudney [39] with a maximal ball threshold of 27. Note, how the vessels indicated with the red arrows are not present in the right image, since their maximal balls are below the threshold and will go away in the first pass of the algorithm. (Images created using the AngioVis Toolbox)*

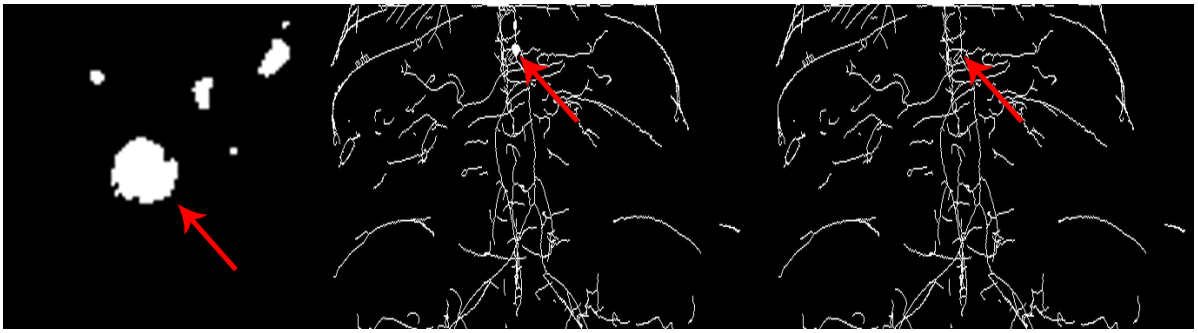


Figure 3.7: *Example usage of local morphological operations on a human aorta in order to reduce the noise resulting from combining. If a background point has more than four foreground neighbors, it becomes a foreground point too. The right image shows an axial view of the aorta, whereas the middle image presents a frontal view of the result of the skeletonization without using noise reduction, leading to a clump. In the right image, noise is reduced beforehand. (Images created using the AngioVis Toolbox)*

lines and has fewer little branches at the end of the vessels. Those result from the computation of the maximal balls, because the initial pass, where points are peeled off according to the centers of such maximal balls, leading to noisier centerlines. Another disadvantage is the selection of a threshold for the maximal balls. Since in the first pass no check is performed whether a point is simple, structures smaller than the user selected threshold can possibly vanish, as illustrated in Fig. 3.6 by the red arrows. Furthermore, since vessels vary a lot in their diameter, determining an appropriate threshold can be very time consuming and requires additional user interaction, which is not a proper solution since the user needs not be familiar with the topic or with the results and relation of some special parameters.

Skeletonization needs the whole data volume in the system memory. As already mentioned, this can lead to a bottleneck for huge data-sets. To overcome this, a streaming version of skeletonization is desired. If we take a closer look at the implementation of the method described by Lee et al. [26] we can realize that in every boundary peeling phase the whole volume is run through. This can be easily adapted for streaming purposes, using a slab for neighborhood information, which makes the implementation quite efficient. Only one case needs some minor effort. When a boundary point is selected for deletion, instead of adding it to a list, it becomes marked as a boundary point by assigning a mask to the value in the data volume. During sequential re-checking, the whole data volume is run through again, checking for each marked boundary point if it is indeed deletable. If yes, it is set to background, if no, it is marked as foreground. Since the input volume results from a threshold operation beforehand, only fore- and background are distinguished, which makes the marking of boundary points possible.

A problem occurred during skeletonization of the aorta, as shown in Fig. 3.7, where some clumps are produced. One possible reason for those stems from combining the volumes of the scale space detection, where the sizes of the detected vessels do not necessarily fit together well. Hence, holes and heavy noise around the aorta are introduced, since the data is not

combined smoothly and seamlessly due to the obtained different sizes from the scale space detection and further from the threshold operations. To overcome or reduce the influence of this problem, morphological operations, such as dilation, erosion, closing or opening should be further investigated to remove the noise and prepare the data for skeletonization. In our approach, a rather simple solution is chosen. Controlled by a threshold value, a background voxel will be set to the foreground if it has more than or equal to the threshold value foreground neighbors. As empirically tested a value of four gives acceptable improvements.

Summing up, skeletonization is a very challenging task in 3D which still requires a lot of issues to be investigated and, especially in medical visualization, we need robust centerlines of vessels to clearly visualize the lumen. Since skeletonization is prone to noise we need a smooth object beforehand. We described a very simple approach to reduce noise, but as future aspects more sophisticated methods can be applied.

3.1.7 Graph Conversion

The final step in the vessel detection pipeline is the conversion of the skeleton to a graph representation for visual exploration and visual queries related to specific domain semantics. Medical doctors will operate with these semantics in order to select specific vessels and explore dense vascular structures. Moreover, a graph representation allows to visualize reformations with regard to the lumen of selected vessel bundles. Since the *AngioVis Toolbox (AVT)* supports vessel trees in the XML format, the conversion produces such a one and furthermore this is a widely spread and readable format. Basically, the graph conversion consists of four steps with an additional one which is needed to convert the graph from a memory representation to a desired output format. Since this might be changed, it is implemented as a separate part.

In the first step the number of neighbors (using 26-connectivity) for each foreground voxel is assigned to the voxel's value and branches and end-points are determined, both called *Control Point (CP)*. Since the skeleton satisfies the thinness property, a point with more than two neighbors is a branch and a point with only one neighbor is an end-point. All others must have exactly two neighbors and are called edge-points. As an optimization, the address of each *CP* is assigned to the voxel's value, in order to access them during the segment building step rapidly, as explained later. To ensure that data is neither lost nor altered in a way that segment building will be restricted or even impossible, it can be shown that all other points are edge-points with a voxel's value of two, since they have two neighbors, or background with a value of zero. Taking the premise that no pointer address will ever have the value of two (which never is the case), the distinction between control and edge points is still preserved. Fig. 3.8 illustrates the initial distinction between the several types of points.

The second step builds the segments by starting from each *CP* and recursively tracing a path along edge-points until another *CP* is reached. Since the control points are stored in the volume as pointer addresses, they can easily be identified during the recursive propagation, as illustrated in Fig. 3.8, where a segment is built starting from a branch-point. If the pointer address was not stored in the data volume, the resolution of control points would need an explicit lookup through a list, which would result in a significant performance impact due to a possibly huge number of control points.

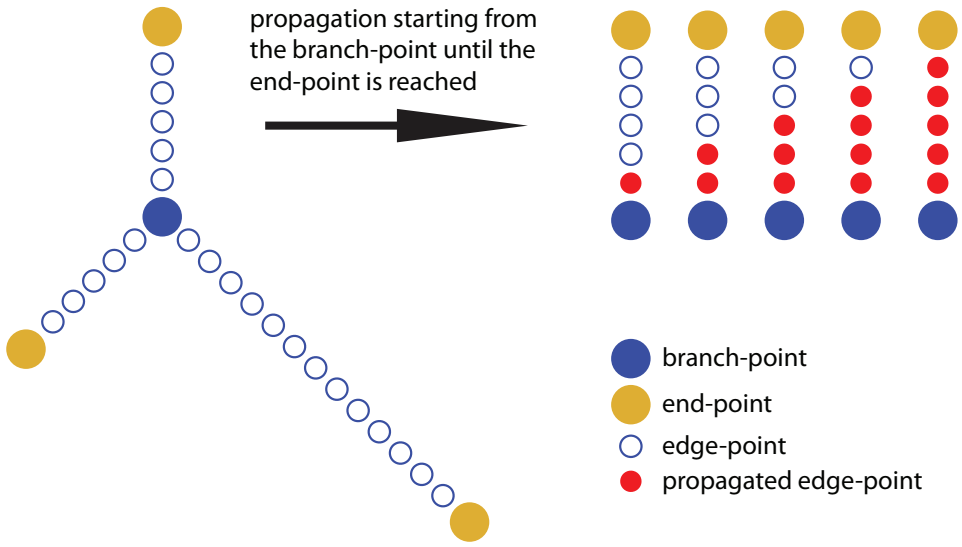


Figure 3.8: *Illustration of the propagation in order to determine the segments. Propagated edge-points get marked. In order to denote already built segments, each CP is used only once for propagation.*

After all control points have been used for propagation, in the third step all segments are built. Then, in order to reduce the number of small segments, they are pruned according to specific domain semantics, such as *minimum segment length*, which can be controlled by the user via command-line arguments. It should be mentioned that this pruning step is irreversible and cannot be controlled later via the interface in the *AVT*. For this reason this step is thought to act like a noise reduction of little segments, which are of no interest at all.

In the fourth step the size of the vessels, namely the radius obtained by the sigma-volume from the combining step of the vessel detection pipeline, is stored in the control points and edge-points of all segments. Furthermore, each segment is assigned another property, called *average segment sigma* or more explanatory average segment thickness, which is the average size of each edge-point within the segment including the start and end *CP*. This property is particularly useful during the visualization of a selection of vessels, since they can be differentiated according to this.

The last step is writing the graph to a file using a specific format. We chose XML, since *AVT* supports this file format and the former is very convenient. Every *CP* is stored with an index, its number. Every segment is stored using the indices of the start and end *CP* and, furthermore, all edge-points of each segment are stored too. For all control points and edge-points, the radii corresponding to the width of the vessel are additionally stored.

In order to produce fewer small and possibly not connected segments, an edge linking operation could be applied before, which can be addressed as a future aspect. This step could be fit before the pruning step, since small segments could be connected to other ones, and thus they need not be deleted anymore. Furthermore, since the operation takes place on the whole volume, streaming approaches should be investigated and implemented to reduce the memory consumption and to avoid possible bottlenecks.

3.1.8 Conclusion

We presented a framework which detects vessels from volumetric data-sets and which requires only a minimum of user input, since this is one of the initial prerequisites. Furthermore, its input parameters (the sigma values of the scales to compute, the low and high threshold for *HT* and the minimum segment length for the graph representation) are intuitive and easy to handle, which allows non domain experts to use this framework within the AngioVis Toolbox. Since clinicians are the target group, the main focus of interaction was to minimize the number of user parameters and to make them readily understandable.

Another aspect of the described framework is the accuracy and reliability, since users must rely on what they see. It is very important that the visualization is correct or that vessels are correctly detected. Although performance is not so crucial, in some cases such as pulmonary diseases it plays an essential role, hence the variety of described and implemented optimization techniques and algorithms.

Furthermore, stability of the framework is a desired property that is very important, since clinicians may not know what to do in the case of an error. This is investigated by providing the possibility to terminate and restart the vessel detection pipeline. Since every tool works independently they can be restarted without further problems.

Availability and scaling with regard to the underlying hardware are other very useful properties of frameworks. Since clearly not all hospitals operate on the same type of computer, the framework has to run on all types, or at least on the most important. Additionally, scaling in the sense of utilizing the full hardware will give significant improvements as mentioned.

Since all steps are implemented as more or less independent filters, each of them can be exchanged with more advanced versions, without altering the whole pipeline. Of course the interface must stay the same, but in almost all cases the data volume is passed between two consecutive filters. This makes the pipeline easily extendable which further results in an more comfortable and flexible maintenance.

3.2 Vessel Visualization

As already mentioned, the main goal of the AngioVis Toolbox is vessel visualization. The user has the possibility to select several techniques (*MIP*, *CPR*, *mpCPR*) and the option to zoom and pan around the z-axis. Furthermore, the contrast can be stretched in order to enhance structures. The workspace consists of linked views, mainly an overview window which displays the *MIP* of the data-set, and a visualization window. Selecting is done in the overview window either by clicking or visual querying and the corresponding selected vessels are rendered immediately in the visualization window.

In the next sections, our novel method Centerline Reformation will be described in detail and selection techniques such as visual querying are explained. Finally, focus and context rendering will be presented.

3.2.1 Centerline Reformation

Here we propose a novel method that is capable of reformatting multiple and not necessarily connected segments of arbitrary vessel trees in contrast to the already mentioned techniques, *Curved Planar Reformation (CPR)* and *Multipath Curved Planar Reformation (mpCPR)*. The desired vessel tree is determined by a user according to relevant domain semantics, such as the minimum segment length or the average segment thickness.

As presented in Fig. 3.9, the common *CPR* has drawbacks when sampling horizontal structures, since the vector-of-interest is usually parallel to the x-axis. Even if the vector-of-interest

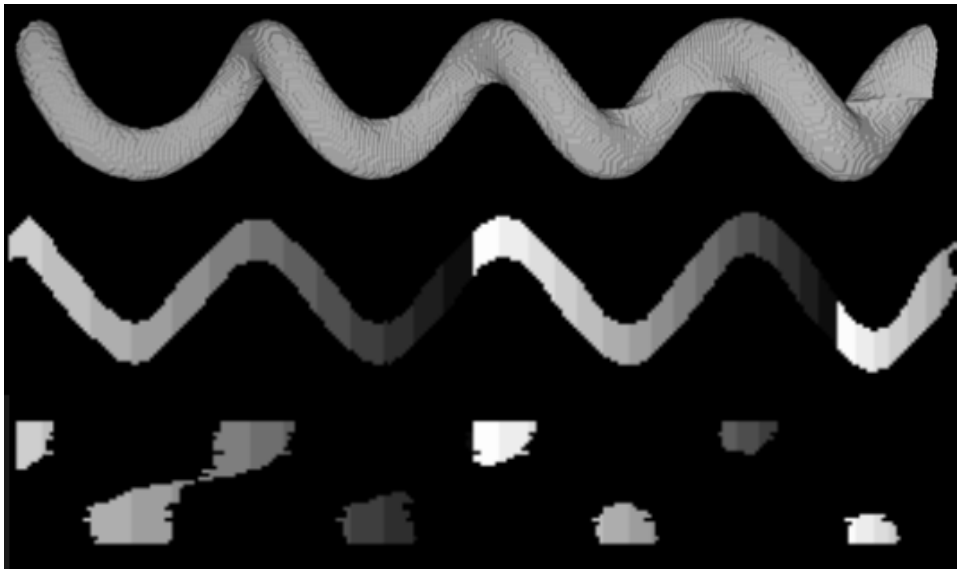


Figure 3.9: *Problem statement of CPR according to horizontal structures. The top image shows a 3D view of a horizontal artificial helix, whereas the central image shows the helix rendered using CR. The bottom image presents the helix using CPR. (Central and bottom image created using the AngioVis Toolbox)*

is taken perpendicularly with regard to two consecutive control points, it will lead to drawbacks when sampling rotations, since some pixels might be missed. Hence, a more general sampling technique is required. Basically, Centerline Reformation can be divided into four main steps. Additionally, we assume that each voxel of every centerline stores the radius of the corresponding vessel computed in preprocessing and that orthogonal projection is used.

First, the *Initial Boundary Set (IBS)* for sequential thickening is specified by projecting every detected vessel centerline onto the image plane. The depth and the sampling position of the volume data of the projected centerline are stored in a special buffer at the projected pixel. Furthermore, the indices of pixels of the initial boundary set are stored in a queue.

In the second step, the boundary set is sequentially thickened according the radius of the vessel which is specified by the projected voxel of the current centerline. Hence, sequential thickening is performed in 2D, namely in the image plane. The growing process is established by taking the first element out of the queue and propagating its sampling position to neighboring pixels, decreasing the radius to grow further by one for every iteration and terminating when zero is encountered.

In the optional third step, halos are created by means of sequential thickening again. If the first growing phase of a pixel reaches zero, the desired width of the halo is assigned as radius and the pixel is added to the queue again, but this time for a halo propagation.

The fourth and final step is the image creation where the data volume is sampled according to the pixels in the propagation buffer. Basically, there are three types of pixels, namely *empty*, *occluded* and *sample*. The final pixel's color in the image depends on the type of the pixel of the propagation buffer. In the case of a sample pixel, the data volume is sampled at the stored 3D volume data sampling position. If the pixel is occluded, a halo color is assigned and in the case of an empty pixel, either a specified background color is assigned, or a desired context is rendered.

To establish the propagation or thickening process, two special buffers are used, a *Front Propagation Buffer (FPB)* and a *Back Propagation Buffer (BPB)*. Each of them stores information about the volume data sampling position (or short *Sampling Position (SP)*), the depth of the corresponding voxel, the radius of the vessel, the boundary identifier (increased during each propagation step), the vessel segment identifier, the index of the voxel in the segment control point list (necessary for self occlusions), a flag if the current pixel is a halo and finally if the current pixel is a possible candidate for segment self occlusions.

In the following sections each step of the Centerline Reformation will be described in greater detail with the aim to discuss the differences of the technique in comparison to *CPR* and *mpCPR*, the encountered problems and their possible solutions.

3.2.1.1 Initial Boundary Set

Every voxel of a selected vessel segment of the 3D vessel tree is projected into the image plane and stored in the *BPB*. Mind that for every projected vessel segment the *CR* is computed separately and not for all together. For every projected voxel the corresponding pixel index of the *BPB* is stored in a queue, called *Boundary Pixel Queue (BPQ)*. Every boundary pixel of the *BPQ* is then propagated to the corresponding neighbors in the *BPB*. Before propagating to the neighbors, the current pixel determines whether it is closer to the view plane than the

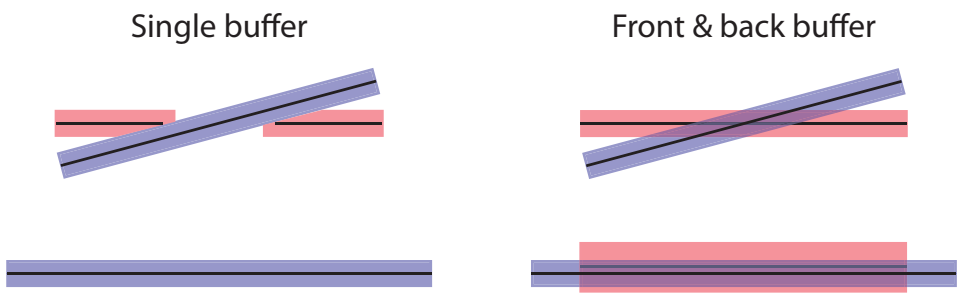


Figure 3.10: *Comparison between a single and two propagation buffers (centerlines are illustrated as black lines). The first row illustrates a partly hidden centerline (of the red vessel). Note how it comes to disruptions when the vessel behind becomes visible. In the second row the situation becomes even worse when using a single buffer, because the centerline of the red vessel is completely hidden by the blue vessel. In this case the red vessel behind would not be visible at all, which results in a significant visual difference as illustrated on the right side.*

currently stored pixel in the *FPB*. If this is true, the current pixel overwrites the *FPB* with the data stored in the *BPB*, otherwise nothing is altered. Moreover, if the radius of the current pixel equals zero, then the propagation stops.

Since the radius of how much a pixel of the *BPQ* needs to be grown is stored in the corresponding voxel of the centerline of the current vessel segment, we can assign this as the maximum width to grow for the corresponding pixel in the *BPB*. In addition, a user-controlled offset is added to the radius which provides the user with the possibility to enlarge the vessels of interest. This might be necessary in cases where the detected radii of the vessels are too narrow to visualize possible calcifications at the vessel borders.

The reason for using two buffers is a question of providing improved visibility. If a vessel is partly covered or obscured by another one, both vessels should be correctly displayed. To ensure this, every time a pixel becomes visible in the *BPB* it is written to the *FPB* too. Hence, if a vessel lies perpendicularly to the view plane, just the closest projected voxel is added to the *BPQ* and thus propagated, as illustrated in Fig. 3.10.

3.2.1.2 Sequential Thickening

The sequential thickening is performed in 2D, respectively in the image plane, according to each projected centerline. In order to give a more formal description of the thickening process, two types of distance metrics have to be mentioned, namely the Manhattan and the Euclidean metric. These are utilized for the growing phase, as illustrated in Fig. 3.11, and defined as:

1. **Manhattan Metric**. The Manhattan distance between two points is defined as the sum of the absolute differences of each of their coordinates, as outlined in Equation 3.2. Moreover, this distance metric can be seen when using 4-connectivity for sequential thickening, since the distance is determined as the sum of the isothetic moves between two points. If we only use integer coordinates for the pixels of the image plane, which is

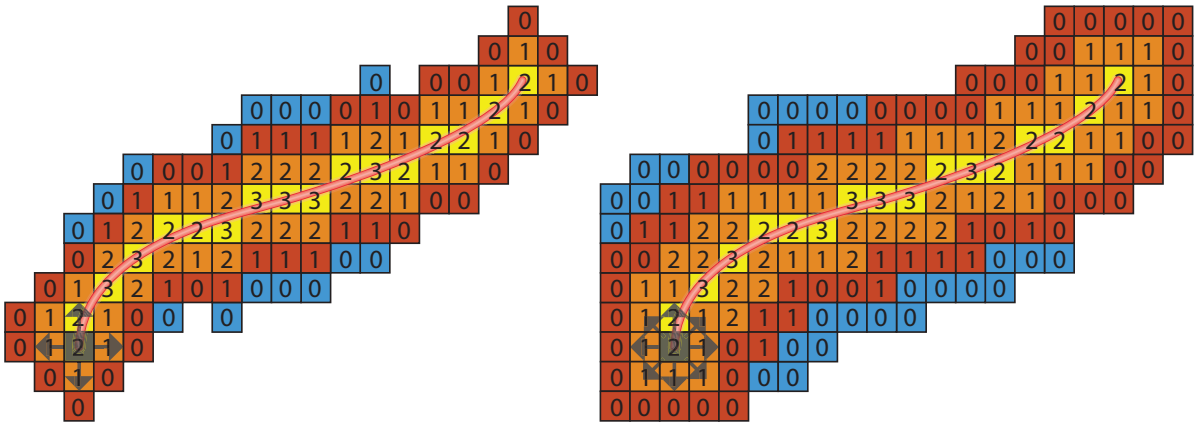


Figure 3.11: *Illustration of the sequential thickening process. Numbers indicate the radius to grow further. The left image shows the growing using 4-connectivity whereas the right image uses 8-connectivity. The order of propagation is from the lower left corner of the vessel to the upper right one. For each level of growing another color is used, whereby the IBS is highlighted in yellow.*

quite obvious, an isothetic move has the distance of one and the Manhattan metric will be an integer metric.

$$d(x, y) = \sum_i |x_i - y_i| \tag{3.2}$$

2. **Euclidean Metric**. The Euclidean distance between two points is stated in Equation 3.3. Since this metric produces floating-point distance values, as a possible integer approximation the chamfer distance metric can be used. In our case, the restriction to integer distances is done by assigning isothetic moves as well as diagonal moves the distance of one. Hence, we can relate this metric to sequential thickening using 8-connectivity, since all neighbors of one point are taken into account.

$$d(x, y) = \|x - y\|_2 = \sqrt{\sum_i (x_i - y_i)^2} \tag{3.3}$$

Propagation to the neighbors is performed using either the Manhattan or the Euclidean distance metric, but already touched neighbors from the same segment are not altered anymore. As presented in Fig. 3.12, using the Euclidean distance metric leads to significantly worse results due to overwriting the neighbor sampling positions of the same vessel segment. For each neighbor, the new *SP* is determined, the radius to grow further is decremented by one and the boundary identifier is incremented by one. The neighbor is assigned to the *BPB* and the index of the neighbor is appended to the *BPQ* if and only if the *BPB* does not already contain a pixel of the same segment. This is a very important constraint, since it ensures and preserves the correct sampling and visualization of the lumen of the vessel, as outlined in Fig. 3.13.

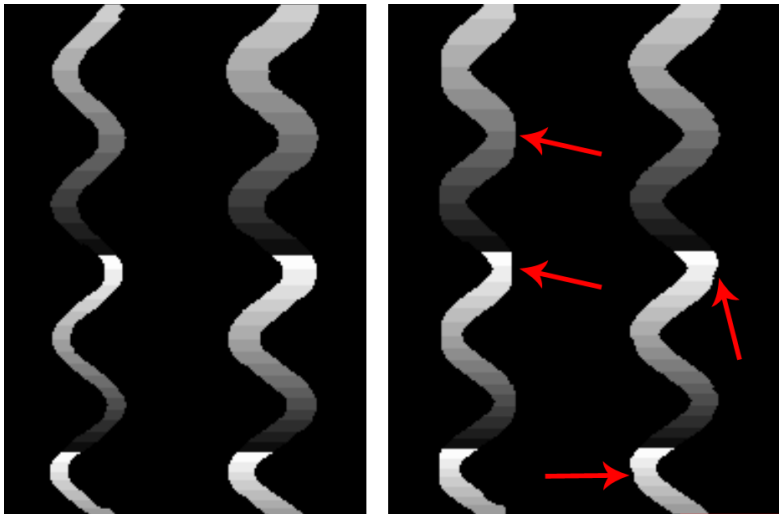


Figure 3.12: *Comparison between 4-connectivity and 8-connectivity used for sequential thickening. The two images on the left use 4-connectivity and show the same helix with different sampling offsets, whereas the images on the right apply 8-connectivity. Note how the pixels in the curves are overwritten in the right-hand images, resulting in cut-off curves. Although the curve fitting improves with increased sampling offset, the problem still remains (pointed out by red arrows). (Images created using the AngioVis Toolbox)*

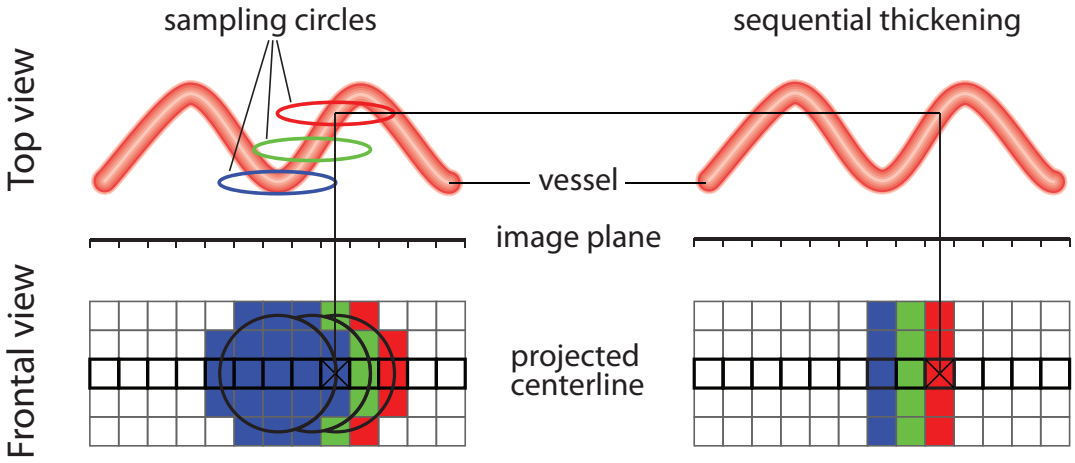


Figure 3.13: *Illustration of the sampling difference between common CPR and CR. The left image illustrates a sampling method where sampling circles, according to the radius of the vessel, slide along the centerline. This method is applied in CPR, but instead of a sampling circle the vector-of-interest is used, which is usually parallel to the x-axis. Note how the closest value is always taken, regardless of the distance related to the centerline. In contrast, the right image illustrates the approach of CR, where the value according to a distance metric to the centerline is taken. Furthermore, no pixel from the same segment, which has already been sampled, is touched anymore and distances are propagated during sequential thickening.*

During the next growing phase the pixels of the *BPQ* try to update the *FPB* before they propagate to their neighbors. Since we use orthogonal projection the depth of the neighbors of a pixel stays the same. Sequential thickening is applied by removing the first boundary pixel from the set and all subsequent pixels are appended at the end of the queue. Since the boundary is increased by one pixel every time, correct thickening, according to the extent of the vessel, is preserved.

The algorithm terminates since the radius to grow further is decreased by one every time and if it reaches zero, the pixel is not considered for further growing. Thus the set of boundary pixels will become empty and the algorithm will terminate. An outline of the implemented algorithm is shown in Algorithm 4.

3.2.1.3 Halos

As described in [10], depth dependent halos enhance the depth and overall perception of dense line fields. Whether displaying the lumen of a selected vessel bundle or neural pathways (for example resulting from Diffusion Tensor Imaging), both have in common, that a very high number of paths, lines or vessels are rendered. This necessarily leads to overlapping, partly covered and even objects lying densely together, with the visual side effect that depth perception is reduced, as displayed in Fig. 3.14.

For this reason halos can be optionally added to each vessel segment, in order to address this problem during visualization of dense vessel bundles. Moreover, the user can define the

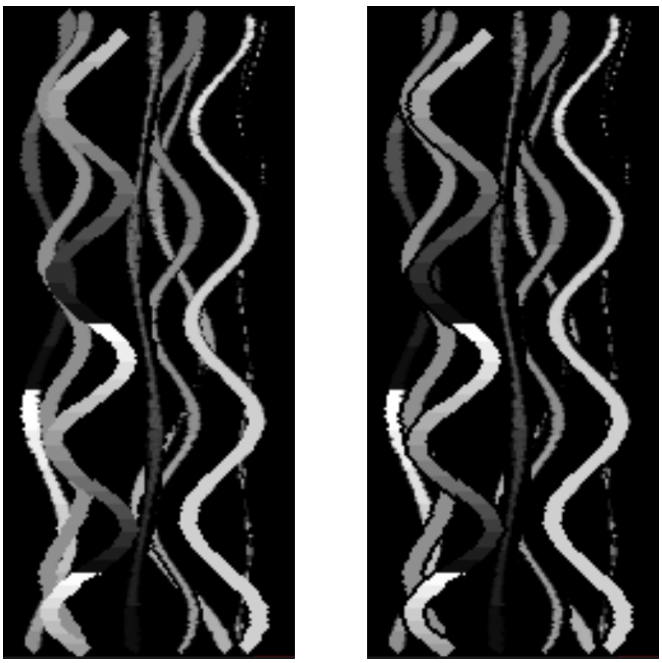


Figure 3.14: *Example of helices without (left) and with (right) halos to emphasize depth perception. Note how halos increase spatial orientation in dense fields of tubular structures. (Images created using the AngioVis Toolbox)*

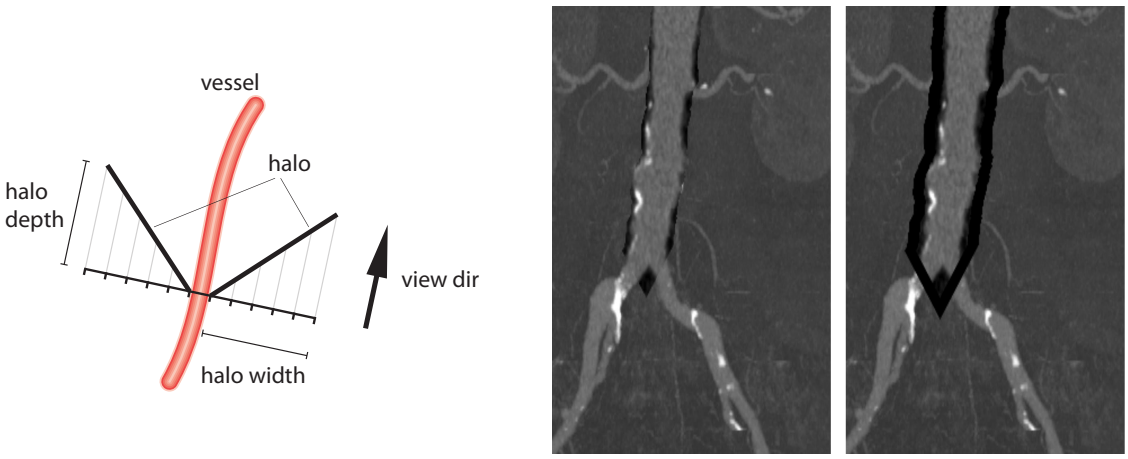


Figure 3.15: *Illustration of halo depth computation. The left image illustrates how the depth of the halo is computed with increasing halo-width. The central image (small halo) and right image (large halo) present examples of different halo widths. (Middle and right image created using the AngioVis Toolbox)*

width of the halos to more or less emphasize depth perception. The halos are established, after a pixel from the *BPQ* reaches zero radius. Then this pixel gets marked as halo, the desired halo width becomes the new radius for growing and it is appended to the *BPQ* again. The actual calculation of the new depth of the halo is performed by adding the identity view vector to the sampling position of the current pixel. Fig. 3.15 illustrates on the left the growing of the halos around the lumen of a vessel segment and the depth computation. Since sequential thickening is also used for halos, they are continuously created around vessels, resulting in smooth transitions between partly occluded vessels.

3.2.1.4 Self Occlusion

A self occlusion is defined as an occlusion of a segment with itself, for example a loop, as shown in Fig. 3.16. The reason why this is a special case and must be handled separately resides in the thickening process, where we stated that pixels in *FPB* are not overwritten with pixels of the same segment, even if they lie closer to the view plane. As already outlined and illustrated during the explanation of sequential thickening, why this constraint is necessary to achieve correct results, it leads to a problem concerning visibility of only one vessel segment. Hence, we have to relax the described constraint in order to get the correct visibility at the points of occlusions.

On this account, after the centerline has been projected onto the image plane, a visibility step is performed, in order to reveal and detect possible self occlusions. It has to be mentioned that all projected pixels are stored in the *BPQ* and the index of a pixel in the queue is referred to as *Pixel Index (PI)*. To determine visibility we loop through all pixels of the *BPQ*, in ascending order, checking if a pixel has a neighbor of the same segment, either closer or further away according to the view plane. This will be almost always true, since it is very unlikely that two consecutive pixels have the same depth in regard to the view plane and if so,

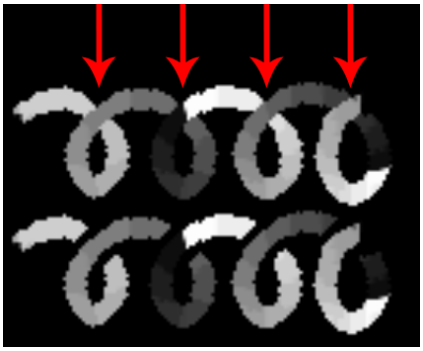


Figure 3.16: *Example of one segment self occlusions shown by reference to an artificial helix. The red arrows indicate the positions of occlusions. In the top image no halos are rendered, whereas in the bottom image halos are used to emphasize the depth perception at the intersections. (Image created using the AngioVis Toolbox)*

it cannot be distinguished which one is behind or in front. If a pixel encounters a neighbor of the same segment now, it is determined which one is further away from the view plane and this pixel gets, together with its preceding and following pixels of the *BPQ*, marked as possible candidates for occlusions, which is illustrated in Fig. 3.17. Normally all pixels will get marked as possible candidates for occlusions, except if they lie in the same plane. Hence, we previously introduced a constraint to ensure correct sampling of the lumen by stating to leave pixels of the same segment untouched and now we contrarily discard this constraint by relaxing it in a too general fashion.

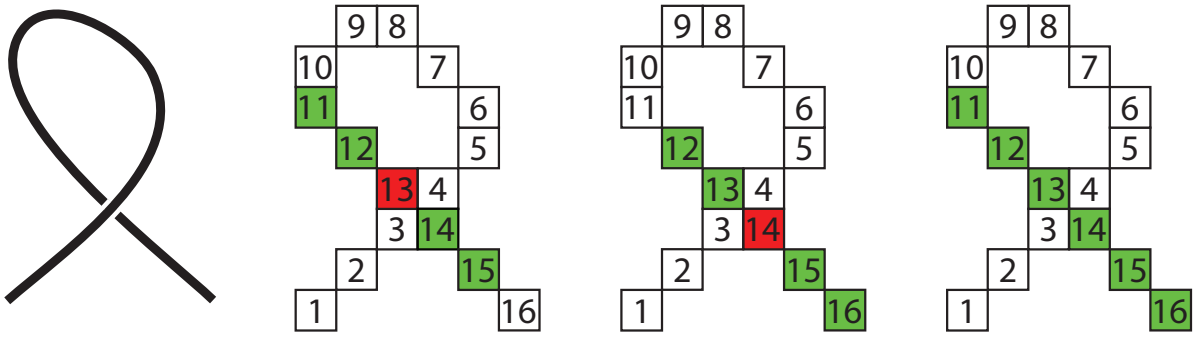


Figure 3.17: *Illustration of one segment self occlusion. The numbers indicate the index of each pixel in the IBS. From left to right, the first image shows an outline of the segment, where it is explicitly made clear at the intersection which part goes behind. If we start the visibility check in ascending order, at pixel three we encounter a neighbor which is behind and has a high enough PI to become an occlusion candidate. Furthermore, the neighbors of pixel thirteen (red) become occlusion candidates (green) too, according to the vessel radius stored in pixel thirteen. In the third image the same for pixel fourteen is encountered as before and finally in the right image all possible occlusion candidates have been correctly marked.*

To overcome this, the *PI* is additionally used to determine whether a neighbor is indeed a candidate or just a following pixel in the *BPQ*. As displayed in Fig. 3.17, the *PI* differs at the intersection points. Thus we add a threshold according to the difference of two pixel indices. If a neighbor pixel has a neighbor with a *PI* greater than the threshold and lies further away in regard to the view plane, then the neighbor becomes indeed a possible occlusion candidate. As we can see, this leads to correct visibility where only a few pixels at the intersection regions are marked.

Finally, the pixels marked as possible occlusion candidates propagate the mark during the growing phase to their neighbor pixels. This leads to a correct partial display of thicker vessels and even of halos too. Since the initial visibility is determined on the projected centerline, this approach has the drawback that, if occlusions occur during the growing phase, visibility is no longer guaranteed, but this is left open for future investigation.

3.2.1.5 Image Creation

The final step is the image creation. This is done by looping through the whole image buffer and checking if a pixel of the *FPB* is marked as sample, occluded or empty. If a sample pixel is encountered, the volume data is sampled at the stored position using trilinear interpolation. For an empty pixel, a specific background color or a context can be rendered and for an occluded pixel, the desired color of the halo is assigned to the image. The context rendering will be described in details in Section 3.2.3.

3.2.1.6 Limitations

Since visibility is determined according to the projected skeleton, occlusions resulting during the sequential thickening process are not taken into account. The problem resides again in the constraint which prevents overwriting pixels of the same segment. This limitation is illustrated in the left image of Fig. 3.18, whereas the right image shows an example using an artificial helix. Note that not even a halo is created in the regions of occlusions, since propagation is stopped when reaching pixels of the same segment.

Figure 3.18: *Illustration of the visibility limitation of Centerline Reformation. The left image illustrates the case of a self occlusion encountered during sequential thickening and the right image shows an example using an artificial helix. Note that not even a halo is created around the vessel. (Right image created using the AngioVis Toolbox)*

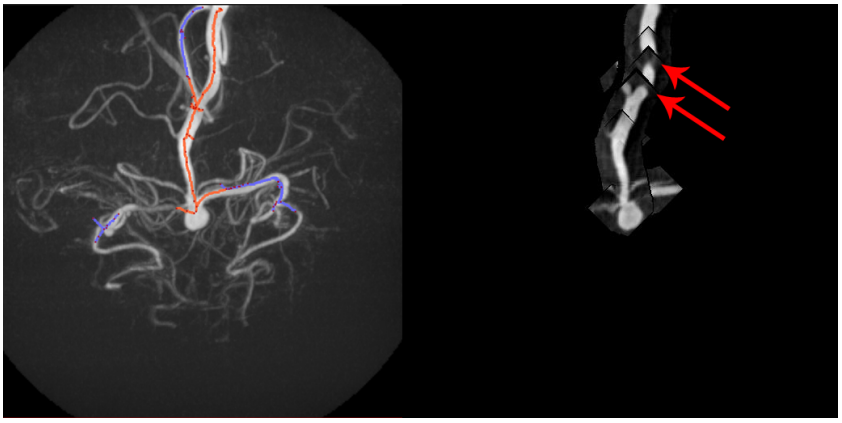


Figure 3.19: *Example of discontinuities of CR resulting from a very large sampling offset and a lot of small contiguous segments. The left image shows the selected vessels in orange and the unselected in blue, whereas the right image renders the selection using CR with a sampling offset of 20 pixels and a halo of 14 pixels. (Image created using the AngioVis Toolbox)*

Another limitation arises if a lot of small vessel segments are lying contiguously together. This can result in visual artifacts or discontinuities at the overlapping regions, if the sampling offset is increased by a significant amount, as shown in Fig. 3.19. This results from the fact, that different segments will overlap and the values according to the distance of the view plane will be taken. This may be solved by applying a reconnection step after the skeletonization in order to provide better connected centerlines, or by investigating the transitions between contiguous segments.

3.2.1.7 Conclusion

Our proposed novel method *CR* is capable of rendering arbitrarily branched vessel trees, regardless of the spatial orientation of the vessels, in contrast to *CPR* and *mpCPR* which are restricted to vertical non-overlapping vessels. Further on, it takes a different sampling approach which allows to display only the vessels without their surroundings, which leads to smooth halos around vessels.

Future aspects will deal with visibility and smooth transitions between consecutive segments. The former would improve segment self occlusions and the resulting visual depth perception, whereas the latter would lead to smooth transitions even if a high sampling offset and halo is desired. Moreover, to overcome the problem of possible discontinuities, the underlying graph of the projected centerline can be addressed and analyzed.

Furthermore, the fast marching method for propagation of distances should be investigated instead of sequential thickening [43]. Moreover, it takes the environment in propagation into account. For example, the front of the *CR* area can propagate with different speeds if another vessel is in the background or if the currently propagating front is located in calcification areas. This may also lead to new ways of specification of the haloed area. These modifications in turn may result in novel rendering techniques and will require further investigation and clinical evaluation.

3.2.2 Visual Exploration using Visual Queries

Visual queries are an intuitive selection technique with the intention to relate the selection to specific properties or domain semantics. Thus, the user can select a specific collection of vessels out of a dense bundle by controlling a number of parameters to reduce the range of the vessels to display for selection. Usually, visual querying is performed by moving the mouse and pressing the left mouse button to visually stroke over the desired vessels. The path traced by the mouse movement is illustrated using a brush, as outlined in Fig. 3.20. Without the possibility of visual queries, the user would have to select each vessel individually, which is impractical, since the number of detected vessels can be pretty high, due to noisy little segments or according to the underlying data.

To reduce the number of dense and complex vessel structures their length and thickness are utilized. The first one, namely the vessel length or called the minimum segment length, is defined as the number of edge-points the segment contains plus the start and end-point of the segment. This allows the user to vary between different lengths of segments and thus noisy segments can be removed. The second property is called the average segment sigma, or differently and more explanatory the thickness of the vessel. It is defined as the average of

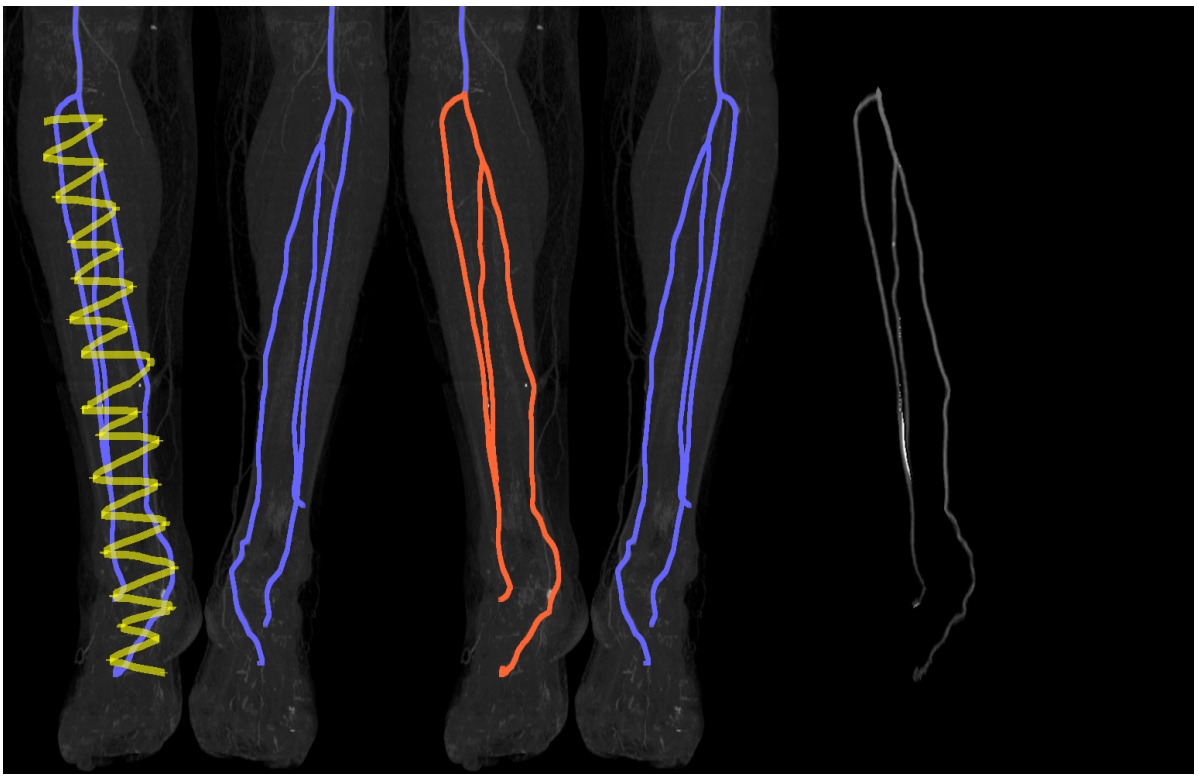


Figure 3.20: *Example of a visual query, shown by the yellow brush in the left image. The selected vessels become highlighted in orange, as outlined in the middle image, and are rendered using CR, as presented in the right image. (Images created with the AngioVis Toolbox)*

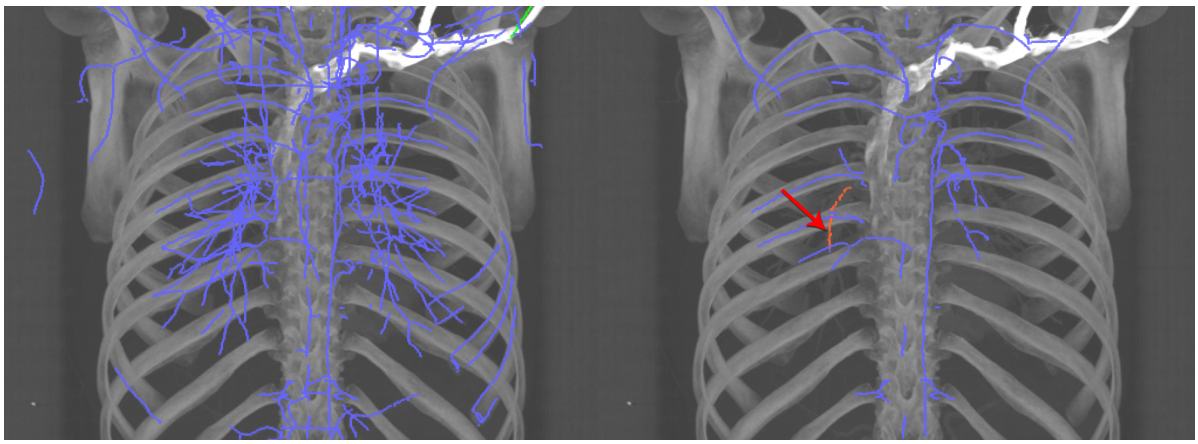


Figure 3.21: *Utilization of the vessel thickness in order to reduce the number of vessels displayed. Note how the position, where the pulmonary embolism resides, is revealed (pointed out by the red arrow) and thus can be more easily selected by visual stroking. (Images created with the AngioVis Toolbox)*

the sigma values of each edge-point of a segment including the start and end-point. Since the sigma equals the size of the scale, it specifies the thickness of the corresponding vessel. Using this parameter, vessels can be selected according to their thickness, as for example the aorta.

Moreover the user is supported with the opportunity to interactively change both domain semantics and sketch the selection of the filtered vessels, as illustrated in Fig. 3.21. Hence, specific vessels can be selected and visualized for further inspection out of a dense structure which allows fine-grain exploration of the region of interest.

3.2.3 Focus and Context Rendering

Focus and context rendering is an essential paradigm when exploring data. As the user is often overwhelmed with a flood of information, it is important to focus on special features or emphasize specific locations or selections. On the other hand the overall context around the stressed area should always be preserved and presented in order to give the user the possibility to orientate himself or herself within the explored data.

Therefore, the possibility to focus on specific regions, vessels in our case, is implemented using visual queries and the visualization is performed by utilizing Centerline Reformation. As already mentioned, *CR* samples according to the radius of the vessel with an optional halo around, leading to empty space elsewhere. This empty space can now be utilized further to render specific context information. This is done by either using the well-known method *Maximum Intensity Projection (MIP)* or the more recent approach, called *Maximum Intensity Difference Accumulation (MIDA)*, which is described in [5]. Whereas *MIP* is well established and fast to render, *MIDA* has the advantage of adding more depth cues without the need of an explicit specification of a transfer function as required in *Direct Volume Rendering (DVR)*. An example of context rendering is shown in Fig. 3.22.

Figure 3.22: *Example of focus and context rendering. From left to right. The first image shows the Centerline Reformation of a selection of vessels without any context. The second one displays the focus region with MIP as context, whereas the third uses MIDA. The fourth and fifth image present MIP respectively MIDA additionally with bones. Note how the focus region is still preserved and enhanced with a small halo around it. (Images created using the AngioVis Toolbox)*

Another point of investigation is performance. Since all steps are done in software utilizing the CPU and context rendering is a quite expensive task, it needs special effort and optimizations to finish in an acceptable amount of time. Since orthogonal projection is used and only rotations around the z-axis are possible in the AngioVis Toolbox, an optimization would be to pre-compute the sampling positions for each slice, since they stay the same except for the z-coordinate. To perform this, the polygon defined by the shape of the slice, usually a square or a rectangle, is rasterized using a commonly familiar polygon rasterization technique, as illustrated in Fig. 3.23.

Furthermore, this leads to equidistant sampling points and no need of trilinear interpolation, since the sample points are the voxels themselves, as illustrated. Once the sampling positions have been pre-computed, they only need to be shifted along the z-axis for all slices of the data volume. Actual compositing is done for either *MIP* or *MIDA*. This leads to a performance increase and moreover each slice can be computed in parallel once the sampling

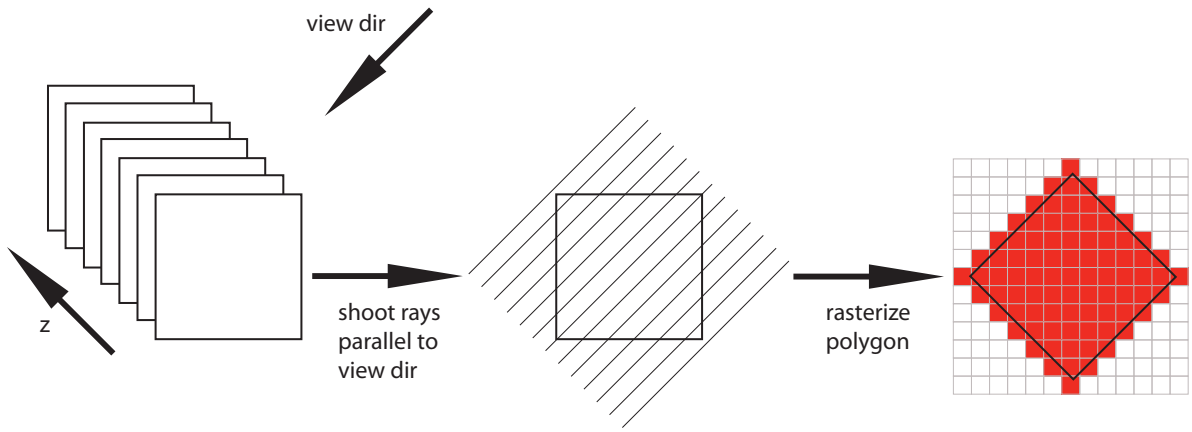


Figure 3.23: *Illustration of polygon rasterization in order to pre-compute the sampling positions required for rendering the desired context. Since orthographic projection is used, the sampling position can be used for all slices by simply shifting the positions along the z-axis.*

positions have been pre-computed, but this has to be investigated as future work in addition with utilizing the *GPU*.

4. Results

In this chapter, results of the vessel detection pipeline and Centerline Reformation will be presented. As the main goal of the vessel detection pipeline is automation, the parameter setting will be discussed and results will be presented by images outlining intermediate results of the pipeline. Furthermore, in order to outline the possibilities and capabilities of Centerline Reformation, which aims for generality, it will be opposed to *CPR* and *mpCPR* in several data-sets, and the differences will be pointed out. In the following sections results are presented first for an artificial data-set and then for visualization of various diseases discussed in the motivation of this thesis, namely atherosclerosis, pulmonary embolism and an aneurysm. Hence, we will present a very general overview of data-sets in order to show the possible application fields. If not explicitly stated, all examples use the vessel tree generated by the presented pipeline.

4.1 Artificial Data

Artificial or phantom data is mainly used to illustrate an algorithm or a technique on generated data. In this case, several artificial helices are used, which vary in their diameter, as shown in Fig. 4.1. To get an overview of the spatial orientation, a 3D overview of the data is presented there. The data-set contains densely arranged artificial helices. Furthermore, the color of these helices varies in order to emphasize correctness of an algorithm.

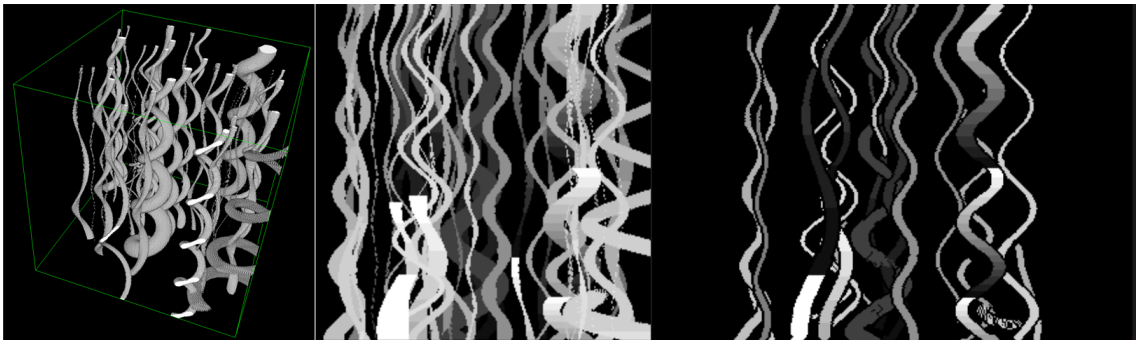


Figure 4.1: *Artificial data-set consisting of several various sized helices. The left image shows a 3D view of the helices, whereas the central image presents the data-set using MIP. The right image shows CR of all helices detected in the vessel detection pipeline. (The central and right image created using the AngioVis Toolbox)*

In Fig. 4.2 the problem of *CPR* encountering horizontal vessels is presented, using the helix data-set rotated 90 degrees around the y-axis. As shown in Fig. 4.3, when investigating dense bundles of tubular structures, *mpCPR* has the disadvantage of partitioning the image according to the branches of the resulting tree, leading to a negative visual impact according to, for example, depth perception. Moreover, as already mentioned in this thesis, halos are implemented to emphasize the depth perception, which is of great advantage when visualizing dense fields of tubular structures.

The intermediate and final results of the multi scale detection of the helical data-set are outlined in Fig. 4.4. In order to view a specific subset of helices, visual querying related to specific semantics is performed, as shown in Fig. 4.5. The two semantics used are the minimum length and the minimum thickness of a tubular structure.

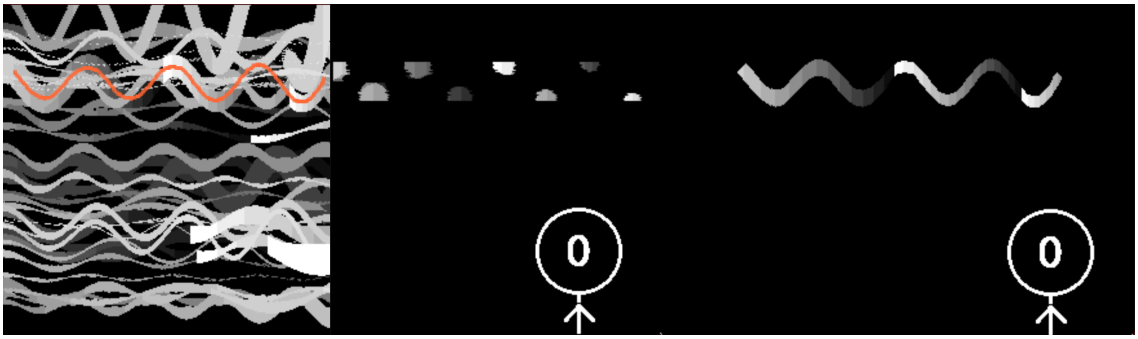


Figure 4.2: *Comparison of an artificial horizontal helix between CPR and CR. The left image shows the MIP of the data-set with the selected helix highlighted in orange. The middle image shows the CPR of the helix, whereas the right image presents CR. Note that CR presents the helix correctly due to the different sampling method. (Images created using the AngioVis Toolbox)*

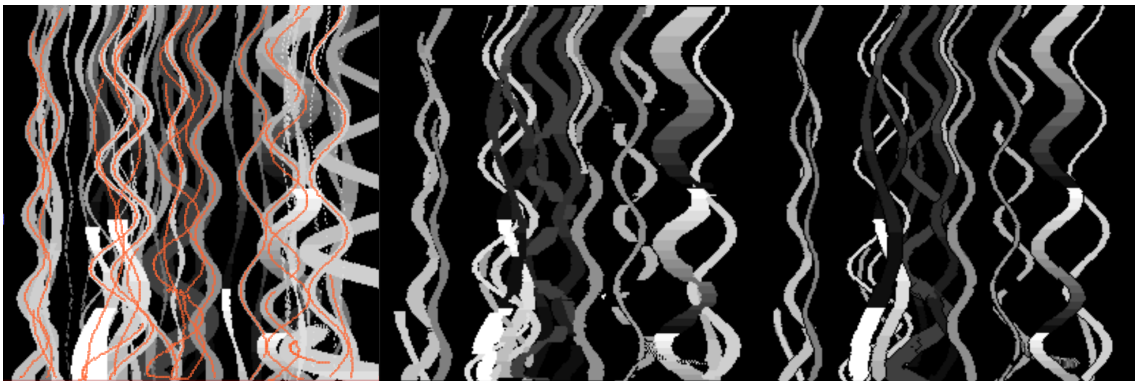


Figure 4.3: *Comparison between mpCPR and CR using an artificial data-set. The left image shows the selected helices. Note how mpCPR in the middle image shows less depth perception than CR in the right image. (Images created using the AngioVis Toolbox)*

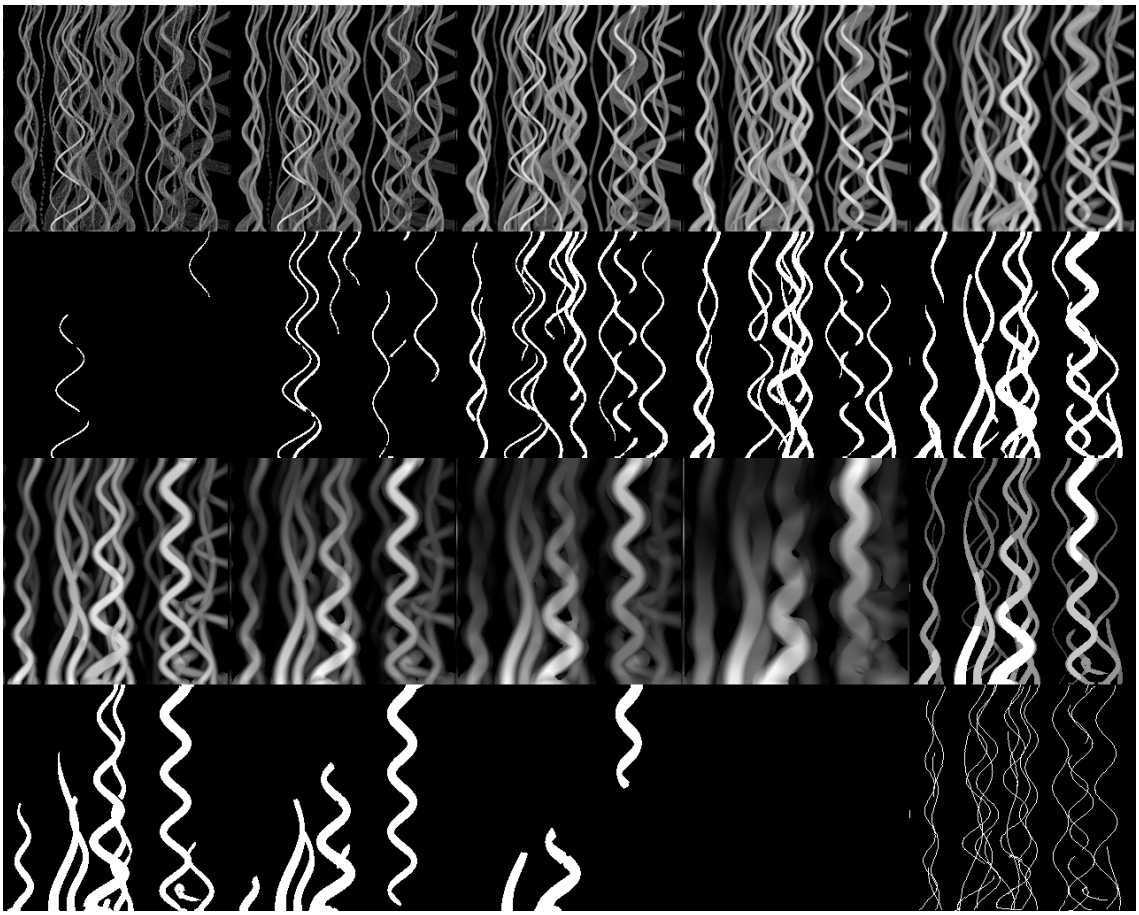


Figure 4.4: *Multi scale detection results of an artificial data-set. From top to bottom, the first and the third row show the result of the Hessian filter (with the scales 1, 1.4, 2, 2.8, 4, 5.6, 8, 11, 16), whereas the second and fourth row show the Hysteresis Threshold of the corresponding scale (using 50000 as low value and 70000 as high value). The right image of the third row shows the combined sigma volume and the image below presents the resulting skeleton. (Images created using the AngioVis Toolbox)*

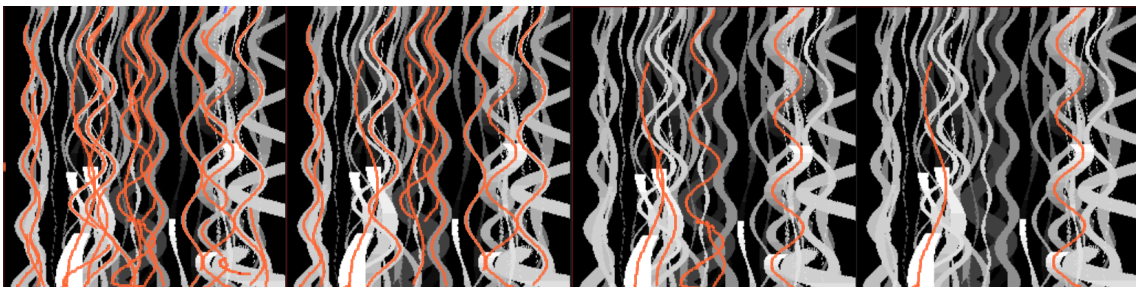


Figure 4.5: *Example usage of the minimum segment length and average segment thickness illustrated on the artificial data-set. From left to right, none, the minimum length, the average thickness and both semantics are used. (Images created using the AngioVis Toolbox)*

4.2 Atherosclerosis

To show examples of atherosclerosis or calcifications, a *CTA* data-set of a human is used, which shows a lot of calcification and has two stents, as shown in Fig. 4.6. Note how the blood flow is preserved, which can be seen well in the right image, using curved reformations. The scales of the vessel detection pipeline and their results are presented in Fig. 4.7. Featuring *mpCPR* and Centerline Reformation in Fig. 4.8 the difference between those two methods is displayed. Since the aorta looks heavily calcified and someone might suggest further treatment because the aorta seems to be blocked in the *MIP* view, this is not the case when using *CR*, as outlined in Fig. 4.9. Context rendering is presented in Fig. 4.10.

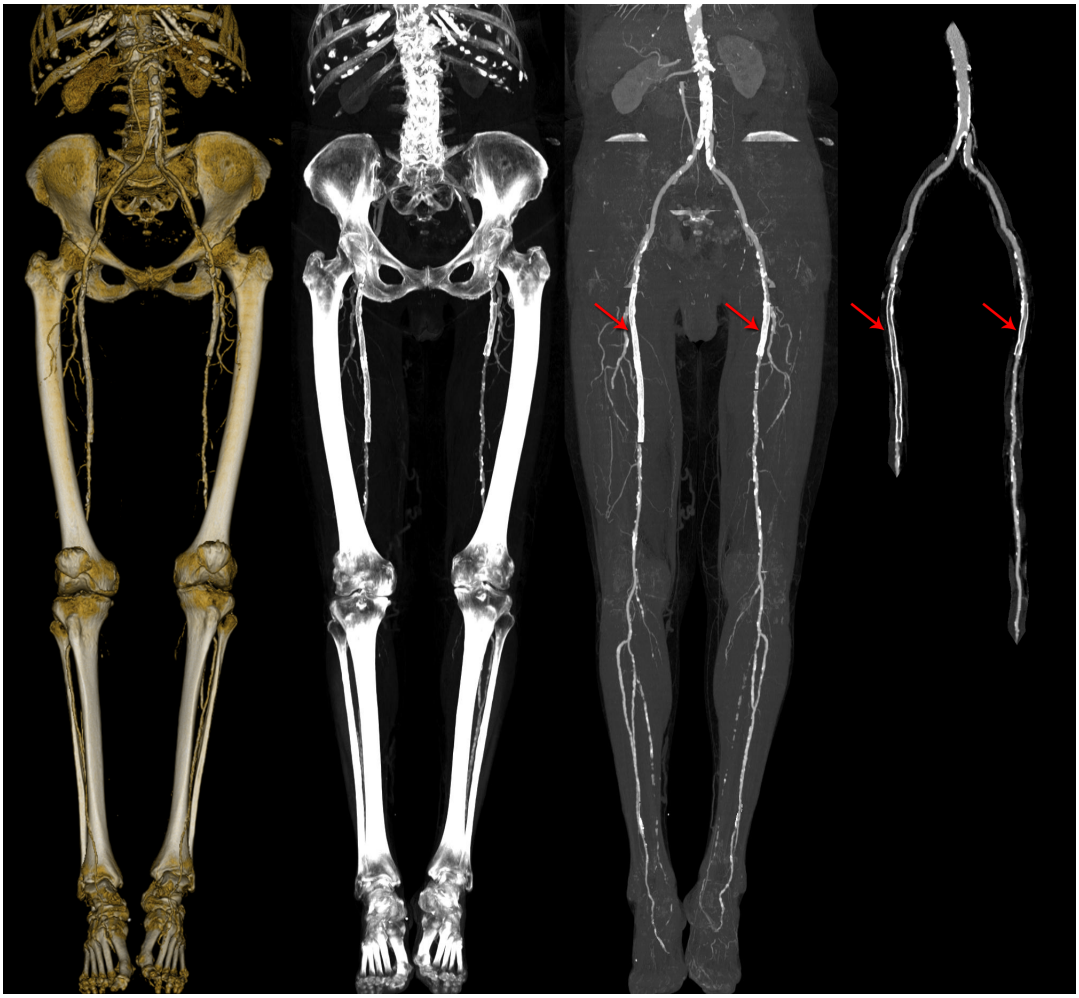


Figure 4.6: *CTA data-set of human body. From left to right, the first image displays DVR of the data-set, the second MIDA with bones, the third MIP without bones and the right image presents the stents, indicated by the red arrows, with CR in order to show that they indeed allow the blood to flow through the vessels. (Images created using the AngioVis Toolbox)*

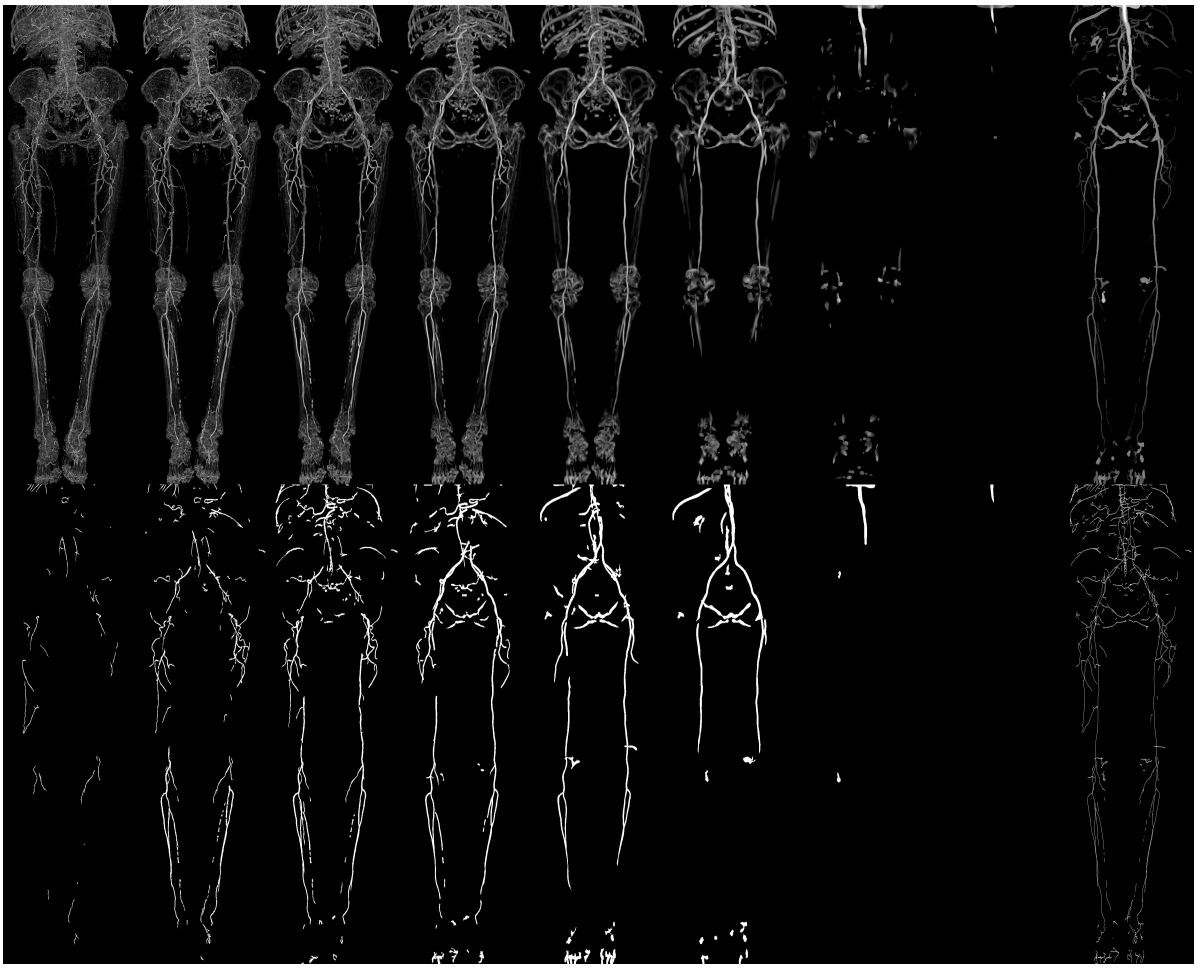


Figure 4.7: *Results of the vessel detection pipeline using a CTA data-set of a human. The top row shows the results of the Hessian filter (with sigma values of 1, 1.4, 2, 2.8, 4, 5.6, 8, 11), whereas the images below are the corresponding Hysteresis Threshold (with 0.6 as low value and 1.2 as high value). The rightmost image in the top row is the combined sigma volume and the image below is the resulting skeleton of all scales combined. All images have been rendered using Maximum Intensity Projection. (Images created using the AngioVis Toolbox)*

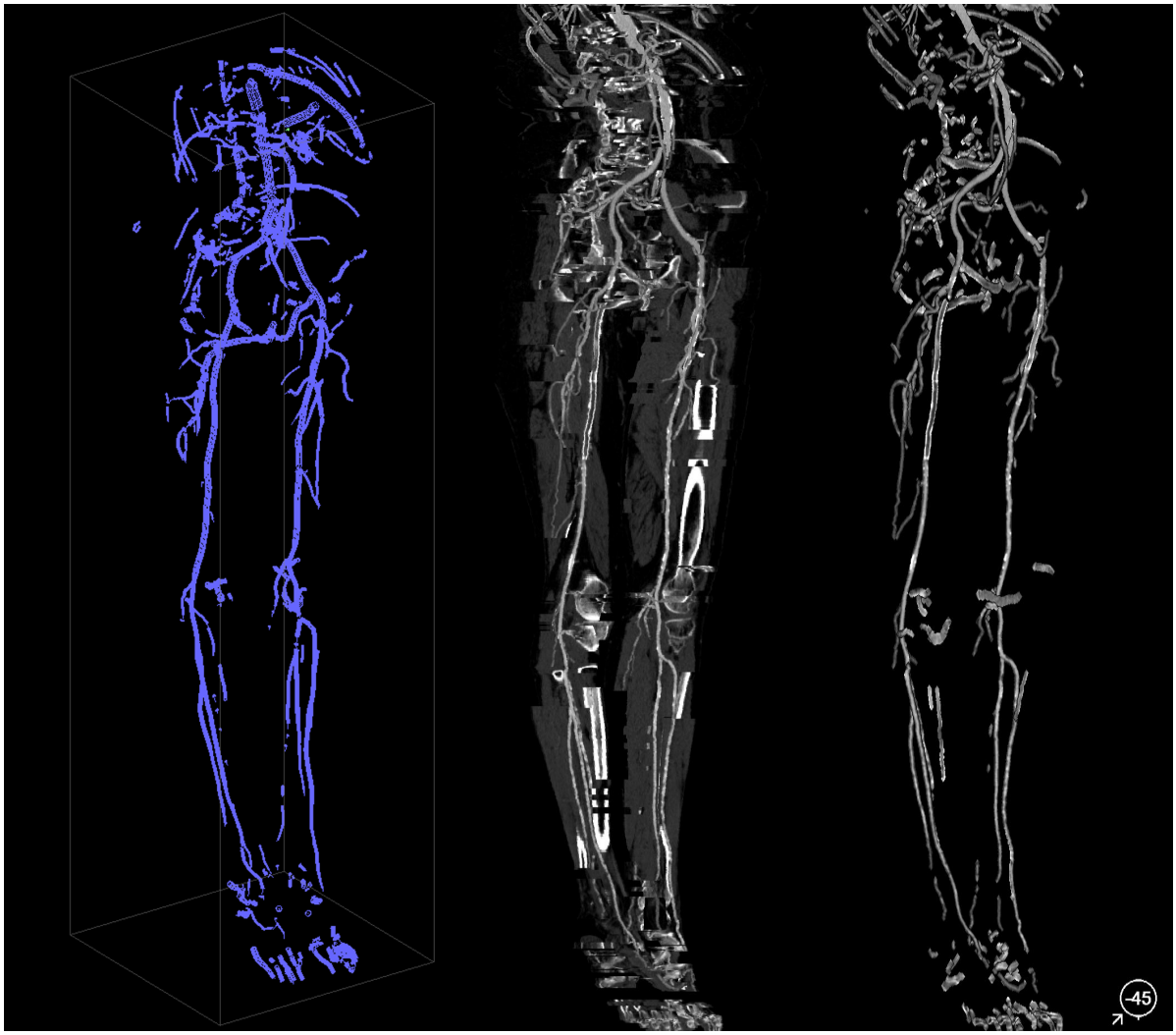


Figure 4.8: *Comparison of mpCPR and CR of the peripheral arteries of a human data-set, acquired by CTA. The left image shows a perspective projection of the vessel tree using a cylindrical model, supported by the AVT, for visualization. The thickness of the vessels corresponds to the size of the cylinders. The middle and right image use orthogonal projection. The middle image shows the mpCPR of the vessel tree. Since the tree is very branched, the image will be highly partitioned. The right image displays the vessel tree using CR. Due to the different sampling approach, no surroundings are created, leading to a clearer visualization of the structure of the vessel tree. (Images created using the AngioVis Toolbox)*

Figure 4.9: *Example of a calcified human aorta. In the left image the blood flow seems to be completely blocked. Whereas in the right image, rendered by CR, it can be noticed that the calcifications are located at the wall of the aorta, so that the blood stream is able to pass through. (Images created using the AngioVis Toolbox)*

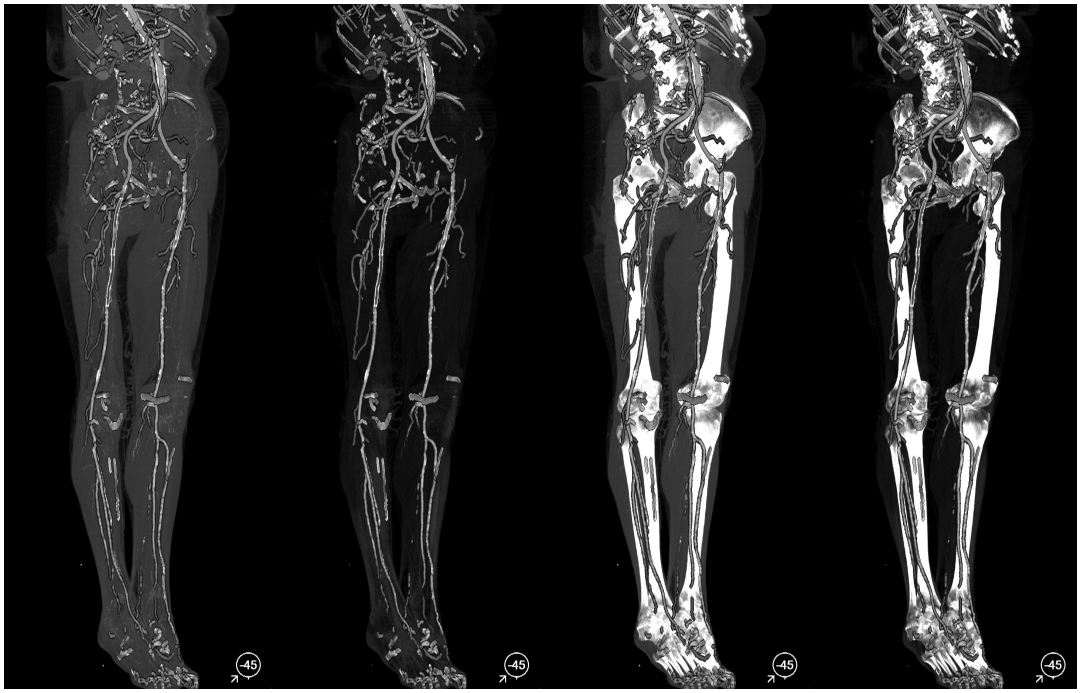


Figure 4.10: *Result of focus and context rendering of a human CTA data-set. In all images the whole detected vessel tree is rendered. In the first two images bone is removed. Furthermore, the first and the third image use MIP whereas the second and the fourth image use MIDA. It is noteworthy that those contexts were rendered utilizing the described polygon rasterization, because of orthogonal projection. (Images created using the AngioVis Toolbox)*

4.3 Pulmonary Embolism

A pulmonary embolism is presented using *CTA* data-set of a human lung as shown in Fig. 4.11. Since the number of the detected vessels is large, an appropriate visualization technique would be of great benefit, since all vessels need to be inspected, as shown Fig. 4.12. The results of the vessel scale detection are presented in Fig. 4.13. Note how the vessels of the lung are well emphasized and, with increasing scale size, even bones are extracted. The embolism is displayed in Fig. 4.14. Note that it is hard to estimate an appropriate contrast stretching in order to make it visible, because on the one hand the density variation of the embolism and its surrounding tissue is very low and thus a significant contrast enhancement is required, but on the other hand this can lead to many artifacts introduced due to contrast over-stretching.

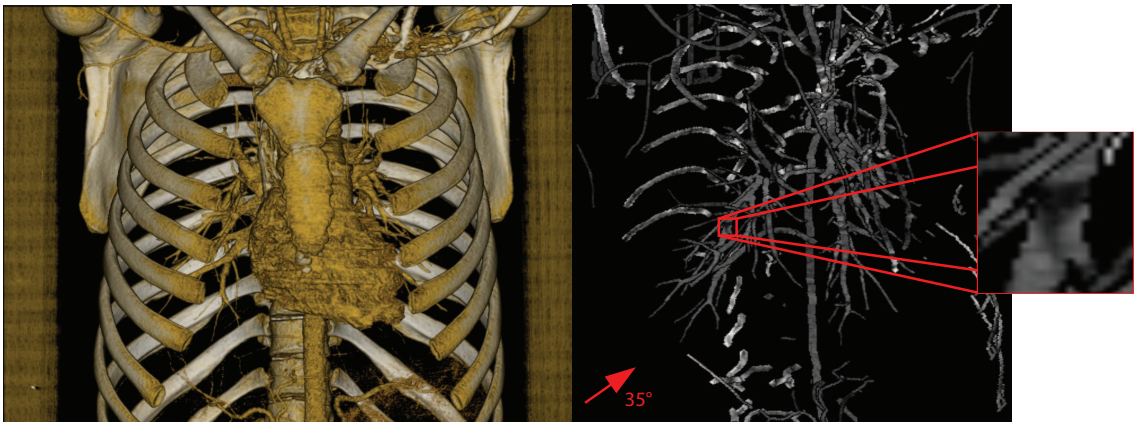


Figure 4.11: *Visualization of a human pulmonary data-set. The left image shows a 3D view, the central image displays the location of the embolism rendered with CR viewed 35 degrees from the left. The right image shows a close-up of the embolism. (Images created using the AngioVis Toolbox)*

Figure 4.12: *Comparison of mpCPR and CR of the vessels of a human lung. Since the vessel tree is highly branched and the vessels lie in arbitrary directions, partitioning the image without leading to a quality impact is difficult to establish using mpCPR, as outlined in the middle image. In contrast, the right image presents the whole vessel tree using CR. (Images created using the AngioVis Toolbox)*

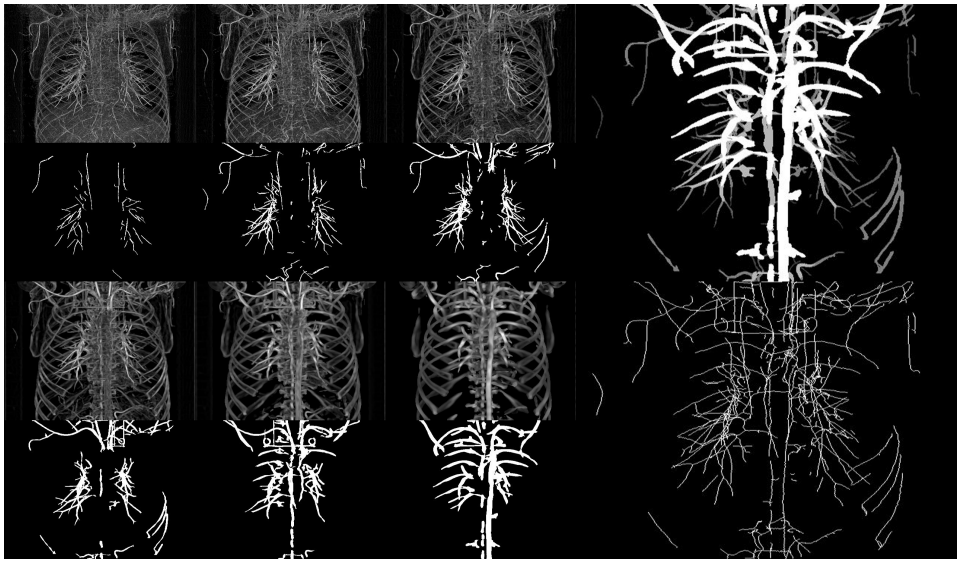


Figure 4.13: *Scale space detection results of a pulmonary embolism of a human lung. The results of the Hessian filter (with the sigma values of 1, 1.4, 2, 2.8, 4, 5.6) and the corresponding Hysteresis Threshold (with 0.6 as low value and 1.2 as high value) are shown in the left images, where the first and the third row correspond to the second and fourth row. The top right image shows the combined sigma volume, whereas the image below is the final skeleton of the detected vessels. (Images created using the AngioVis Toolbox)*

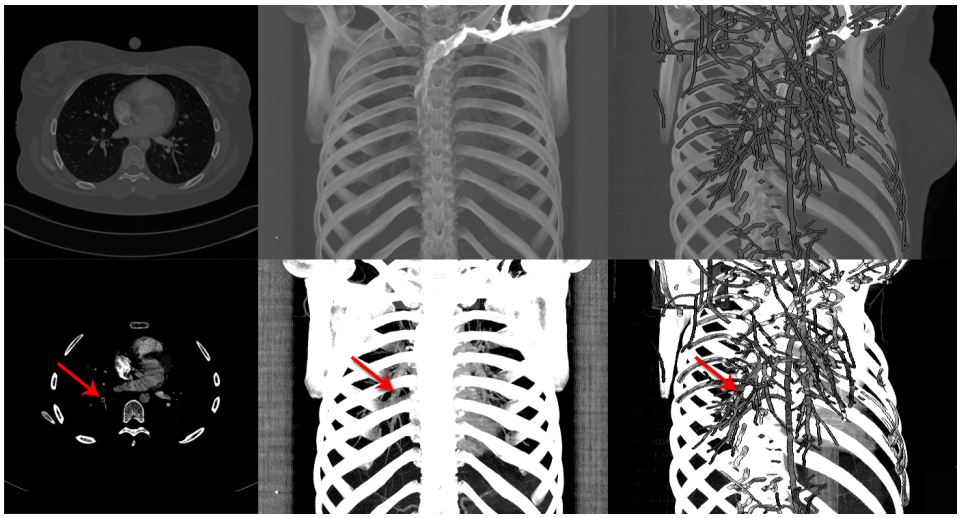


Figure 4.14: *Resulting images of a pulmonary embolism detection of a human lung CTA dataset. The red arrows indicate the position of the embolism. The top row shows the data with no contrast enhancement, hence it is hardly possible to see the embolism, whereas in the bottom row the contrast is significantly enhanced in order to make the embolism visible. The first column shows the transversal slice, the second presents the frontal view and the third column renders the vessels using CR with MIP as context, viewed from a specified angle along the z-axis. (Images created using the AngioVis Toolbox)*

4.4 Aneurysm

As an example of an aneurysm, a saccular one is presented in Fig. 4.15. A selection of specific vessels according to domain semantics is given in Fig. 4.16. Furthermore, the results of the vessel detection pipeline are presented in Fig. 4.17. Note how the different sizes of the vessels become emphasized with increasing scale. Additionally, the combination of all scales and the resulting skeleton of the vessel tree are shown.

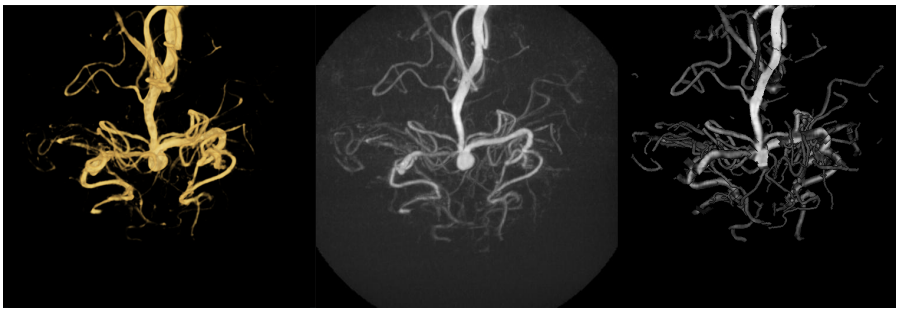


Figure 4.15: *Renderings of a saccular aneurysm data-set. The left image shows a DVR of the aneurysm, the middle image presents the MIP and the right image shows CR of the complete detected vessel tree. (Middle and right image created using the AngioVis Toolbox)*

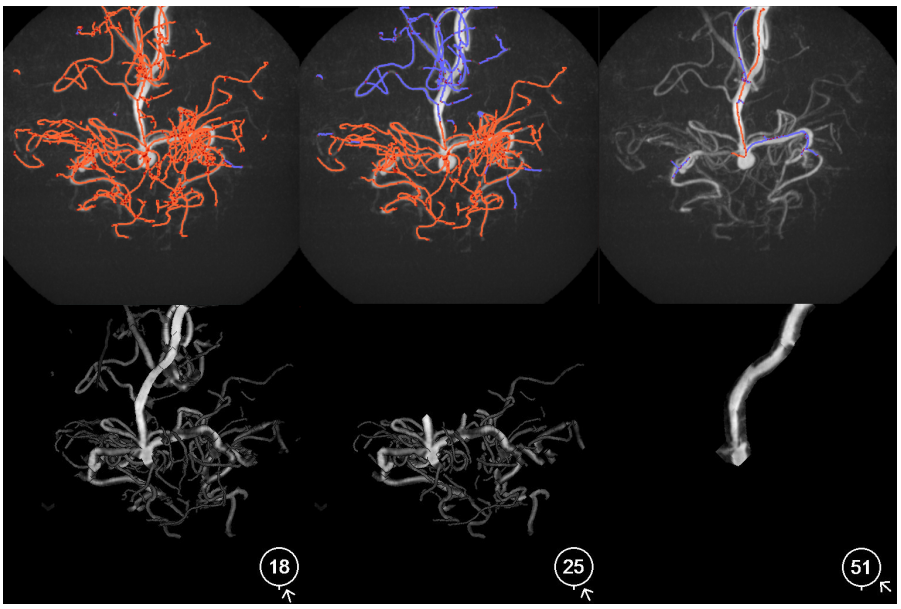


Figure 4.16: *Example usage of visual queries to focus on a specific selection of vessels of the aneurysm data-set. Selected vessel segments are highlighted in orange and unselected in blue. In the upper row the selection is shown and in the lower row the selection is visualized using CR. The right column uses the average vessel thickness to reduce the number of thin vessels in order to display the lumen of the thick vessel. (Images created using the AngioVis Toolbox)*

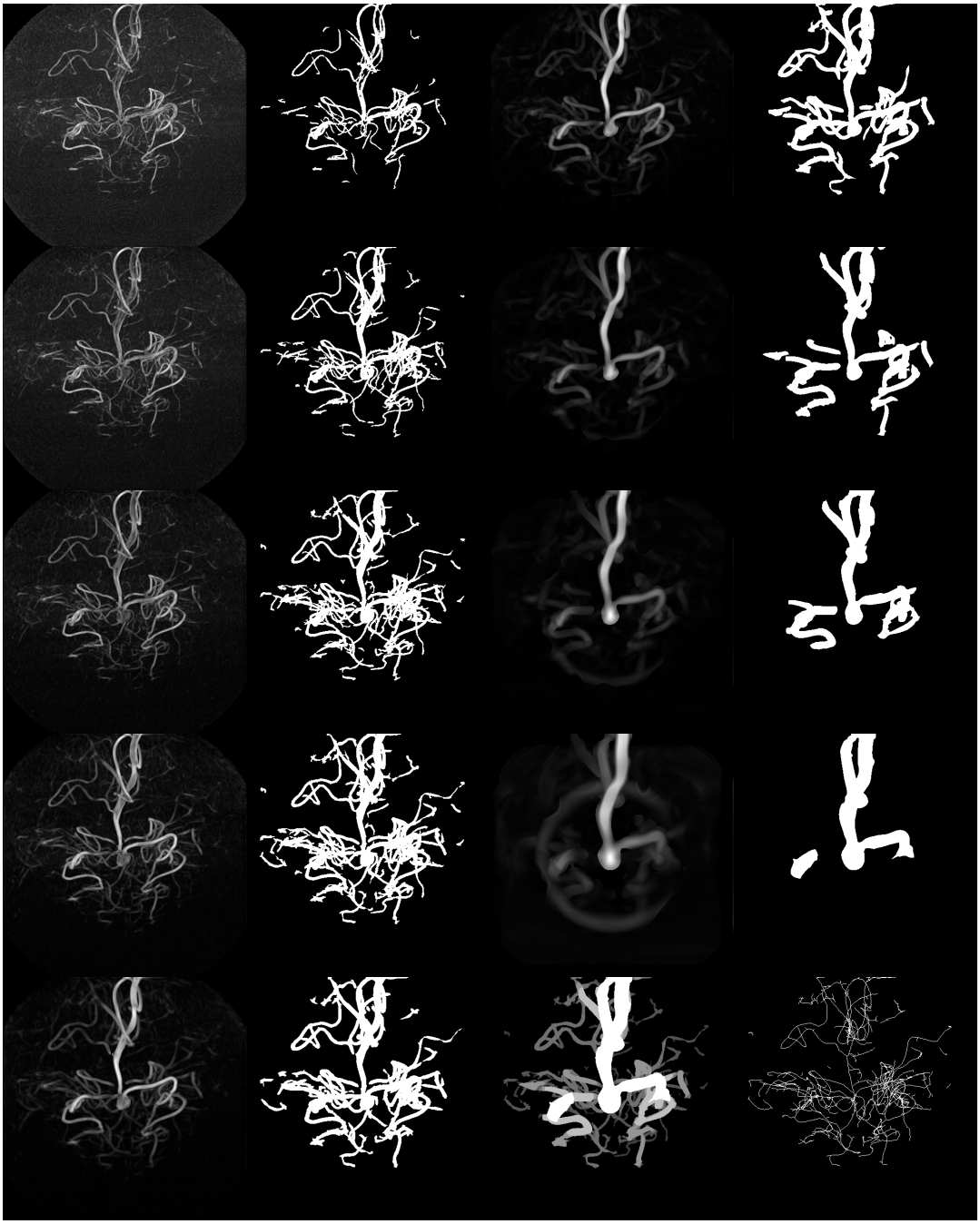


Figure 4.17: *Scale space detection results of the saccular aneurysm data-set. The first and third columns outline the results of the Hessian filter with increasing scales (1, 1.4, 2, 2.8, 4, 5.6, 8, 11, 16) whereas the images in the second and the forth column are the outcome of Hysteresis Threshold (with a low value of 3000 and a high value of 6000) of the corresponding scale to the left. The bottom image in the third and fourth column show the combined sigma volume and the resulting skeleton respectively. (Images created using the AngioVis Toolbox)*

5. Conclusion and Future Work

5.1 Conclusion

In the present thesis, first an overview of diseases has been given, with the intention to introduce the application field of visualization techniques such as Centerline Reformation. Furthermore, underlined in the results part of the thesis, for each presented disease several images have been presented to show the advantages of Centerline Reformation.

In addition to that a vessel detection pipeline has been presented requiring a minimum of user input. As described, the pipeline starts with preparing the data to have the correct and appropriate format. Then multi scale vessel detection is performed according to a series of user-specified scale values. This enhanced data is thresholded subsequently. Here we presented an adapted algorithm of Hysteresis Threshold and, furthermore, presented a possible streaming version for this global operation. Next, the threshold volumes are combined and skeletonization is performed in order to retrieve the centerlines of the vessels. Finally, the skeleton is converted into a graph representation with the target to be loaded in the AngioVis Toolbox for further use such as visualization. A user has to specify only a small amount of parameters in order to run the whole pipeline.

The second part of the thesis concerns visualization of vessels. The focus is on the visualization of the lumen of vessels in order to investigate possible suspicions, such as calcifications or embolisms. A novel method is proposed in order to address arbitrarily branched and spatially orientated vessels in contrast to the state-of-the-art techniques such as Curved Planar Reformation and Multipath Curved Planar Reformation. Several examples have been outlined, using additional specific domain semantics in order to specify regions of interest.

5.2 Future Work

As some future aspects concerning the vessel detection pipeline, the Pock filter mentioned in [37], can be investigated in order to improve the results of the Hessian filter. A broader range of morphological operations to be applied prior to skeletonization, should be investigated in order to reduce the noise in the combined volume. The goal of this operation is to remove possible clumps in the skeleton, since a smoother skeleton would result in better centerlines and vessel graph.

Streaming approaches for skeletonization and graph conversion should be investigated, since those operations still need a noticeable amount of memory, which can possibly become

a bottleneck when using huge data-sets. Furthermore, the given implementation of Hysteresis Threshold can be improved to gain a better performance. For all parts of the vessel detection pipeline further investigations according to performance should be done, for example when utilizing the *GPU*, by using *CUDA*, whereby an increased overall performance can possibly be obtained.

Future aspects related to *CR*, for example, the fast marching method for spreading distances, should be taken into account and, furthermore, the projected centerlines can possibly be reduced to a graph problem. Topological aspects of this graph can be investigated in order to improve the visibility and connection problem. Furthermore, other techniques for focus and context rendering should be explored, as for example non-photorealistic rendering in order to create illustrative context renderings.

Bibliography

[1] AMENTA, N. ; CHOI, S. ; KOLLURI, R. K.: The power crust, unions of balls, and the medial axis transform. *Computational Geometry* 19 (2001), No. 2–3, pages 127–153

[2] BAMAN, T. S. ; COLE, J. H. ; DEVIREDDY, C. M. ; SPERLING, L. S.: Risk Factors and Outcomes in Patients With Coronary Artery Aneurysms. *The American Journal of Cardiology* 93 (2004), No. 12, pages 1549–1551

[3] BLASI, C.: The autoimmune origin of atherosclerosis. *Atherosclerosis* 201 (2008), No. 1, pages 17–32

[4] BROSNAN, M. ; COLLINS, C. ; MONELEY, D. ; KELLY, C. ; LEAHY, A.: Making the Case for Cardiovascular Screening in Irish Males: Detection of Abdominal Aortic Aneurysms, and Assessment of Cardiovascular Risk Factors. *European Journal of Vascular and Endovascular Surgery* 37 (2009), No. 3, pages 300–304

[5] BRUCKNER, S. ; GRÖLLER, M. E.: Instant Volume Visualization using Maximum Intensity Difference Accumulation. *Computer Graphics Forum* 28 (2009), No. 3, pages 775–782

[6] CONDURACHE, A. ; AACH, T.: Vessel Segmentation in Angiograms using Hysteresis Thresholding. *Proceedings of the IAPR Conference on Machine Vision Applications*, 2005, pages 269–272. – ISBN 4-901122-04-5

[7] CORNEA, N. D. ; SILVER, D. ; MIN, P.: Curve-Skeleton Properties, Applications and Algorithms. *IEEE Transactions on Visualization and Computer Graphics* 13 (2007), No. 3, pages 530–548

[8] CORREA, C. D. ; MA, K.-L.: Size–based Transfer Functions: A New Volume Exploration Technique. (2008)

[9] CSETVERIKOV, D.: Basic Algorithms for Digital Image Analysis. . – URL http://visual.ipan.sztaki.hu

[10] EVERTS, M. H. ; BEKKER, H. ; ROERDINK, J. B. ; ISENBERG, T.: Depth-Dependent Halos: Illustrative Rendering of Dense Line Data. *IEEE Transactions on Visualization and Computer Graphics* 15 (2009), No. 6, pages 1299–1306

[11] FRANGI, A. F.: *Three–Dimensional Model–Based Analysis of Vascular and Cardiac Images*, University Medical Center Utrecht, Netherlands, PhD Thesis, 2001

[12] FRANGI, A. F. ; NIESSEN, W. J. ; VINCKEN, K. L. ; VIERGEVER, M. A.: Multiscale Vessel Enhancement Filtering. *Lecture Notes in Computer Science* 1496 (1998), pages 130–137

[13] GOLDHABER, S. Z.: Pulmonary embolism. *The Lancet* 363 (2004), No. 9417, pages 1295–1305

[14] HATCHER, A.: *Algebraic Topology*. Cambridge University Press, 2002

[15] HUANG, Y.-K. ; HSIEH, H.-C. ; TSAI, F.-C. ; LU, S.-H. C. M.-S. ; KO, P.-J.: Visceral Artery Aneurysm: Risk Factor Analysis and Therapeutic Opinion. *European Journal of Vascular and Endovascular Surgery* 33 (2007), No. 3, pages 293–301

[16] ISENBURG, M. ; LINDSTROM, P.: Streaming Meshes. *Proceedings IEEE Visualization*, 2005, pages 231–238

[17] KANITSAR, A.: *Curved Planar Reformation for Vessel Visualization*, Institute of Computer Graphics and Algorithms, Vienna University of Technology, PhD Thesis, 2004

[18] KANITSAR, A. ; FLEISCHMANN, D. ; WEGENKITTL, R. ; FELKEL, P. ; GRÖLLER, E.: CPR – Curved planar reformation. *IEEE Computer Society Press* (2002), pages 37–44

[19] KANITSAR, A. ; FLEISCHMANN, D. ; WEGENKITTL, R. ; GRÖLLER, M. E.: Diagnostic Relevant Visualization of Vascular Structures / Institute of Computer Graphics and Algorithms, Vienna University of Technology. 2004 (TR-186-2-04-02). – Tech Report

[20] KANITSAR, A. ; WEGENKITTL, R. ; FELKEL, P. ; FLEISCHMANN, D. ; SANDNER, D. ; GRÖLLER, E.: Computed Tomography Angiography: A Case Study of Peripheral Vessel Investigation. *IEEE Visualization 2001*, ACM, 2001, pages 477–480

[21] KASS, M. ; WITKIN, A. ; TERZOPOULOS, D.: Snakes: Active contour models. *International Journal of Computer Vision* 1 (1988), pages 321–331

[22] KIRBAS, C. ; QUEK, F. K. H.: A Review of Vessel Extraction Techniques and Algorithms. *ACM Computing Surveys* 36 (2000), pages 81–121

[23] KLETTE, R. ; ŽUNIĆ, J.: *An Efficient Euclidean Distance Transform*. Chap. 3, pages 394–408. BAILEY, D. (Ed.): *Combinatorial image analysis*, Springer-Verlag, Berlin Heidelberg, 2004

[24] KYRLE, P. A. ; EICHINGER, S.: New diagnostic strategies for pulmonary embolism. *The Lancet* 371 (2008), No. 9621, pages 1312–1315

[25] LAW, C. C. ; SCHROEDER, W. J. ; MARTIN, K. M. ; TEMKIN, J.: A multi-threaded streaming pipeline architecture for large structured data sets. *Proceedings IEEE Visualization*, 1999, pages 225–232

[26] LEE, T.-C. ; KASHYAP, R. L. ; CHU, C.-N.: Building Skeleton Models via 3–D Medial Surface/Axis Thinning Algorithms. *Graphical Models and Image Processing* 56 (1994), No. 6, pages 462–478

[27] LEON, L. ; TAYLOR, Z. ; PSALMS, S. ; MILLS, J.: Degenerative Aneurysms of the Superficial Femoral Artery. *Clinical Neurology and Neurosurgery* 35 (2008), No. 3, pages 332–340

[28] LEVOY, M.: *Display of Surfaces from Volume Data*, University of North Carolina at Chapel Hill, PhD Thesis, 1989

[29] LIBBY, P.: Vascular biology of atherosclerosis: overview and state of the art. *The American Journal of Cardiology* 91 (2003), No. 3, pages 3 –6

[30] LORENSEN, W. E. ; CLINE, H. E.: Marching Cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics* 21 (1987), No. 4, pages 163–169

[31] MAHMUD, E. ; CAVENDISH, J. J. ; SALAMI, A.: Current Treatment of Peripheral Arterial Disease. *Journal of the American College of Cardiology* 50 (2007), No. 6, pages 473–490

[32] MORI, S. ; ZHANG, J.: Principles of Diffusion Tensor Imaging and Its Applications to Basic Neuroscience Research. *Neuron* 51 (2006), No. 5, pages 527–539

[33] NATTERER, F.: *The Mathematics of Computerized Tomography*. John Wiley & Sons Ltd, 1986. – ISBN 0-89871-493-1

[34] PAJAROLA, R.: Stream-Processing Points. *Proceedings IEEE Visualization*, 2003, pages 239–246

[35] PASCUCCI, V. ; SCORZELLI, G. ; BREMER, P.-T. ; MASCARENHAS, A.: Robust on-line computation of Reeb graphs: simplicity and speed. *ACM Transactions on Graphics* 26 (2007), No. 3

[36] POCK, T.: *Robust Segmentation of Tubular Structures in 3D Volume Data*, Institute for Computer Graphics and Vision, Graz University of Technology, Austria, Master's Thesis, 2004

[37] POCK, T. ; BEICHEL, R. ; BISCHOF, H.: A Novel Robust Tube Detection Filter for 3D Centerline Extraction. *Lecture Notes in Computer Science* Vol. 3540, 2005, pages 481–490

[38] PREIM, B. ; BARTZ, D.: *Visualization in Medicine*. Elsevier, Morgan Kaufmann Publishers, 2007. – ISBN 978-0-12-370596-9

[39] PUDNEY, C.: Distance–Ordered Homotopic Thinning: A Skeletonization Algorithm for 3D Digital Images. *Computer Vision and Image Understanding* 72 (1998), No. 3, pages 404–413

[40] PUIG, A. P.: Discrete Medial Axis Transform for Discrete Objects. 1998. – Tech Report

[41] RODGER, M. ; WELLS, P. S.: Diagnosis of Pulmonary Embolism. *Thrombosis Research* 103 (2001), No. 6, pages 225–238

[42] SATO, Y. ; NAKAJIMA, S. ; SHIRAGA, N. ; ATSUMI, H. ; YOSHIDA, S. ; KOLLER, T. ; GERIG, G. ; KIKINIS, R.: Three–dimensional multi–scale line filter for segmentation and visualization of curvilinear structures in medical images. *Medical Image Analysis* 2 (1998), No. 2, pages 143–168

[43] SETHIAN, J. A.: A Fast Marching Level Set Method for Monotonically Advancing Fronts. *Proc. Nat. Acad. Sci*, 1995, pages 1591–1595

[44] SETHIAN, J. A.: *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999. – ISBN 0521645573

[45] SHAHROKNI, A. ; SOLTANIAN-ZADEH, H. ; ZOROOFI, R. A.: A fast skeletonization algorithm for 3–D elongated objects. *SPIE Medical Imaging Conference*. San. Diego, CA, 2001, pages 323–330

[46] SHARMA, B. S. ; GUPTA, A. ; AHMAD, F. U. ; SURI, A. ; MEHTA, V. S.: Surgical management of giant intracranial aneurysms. *Clinical Neurology and Neurosurgery* 110 (2008), No. 7, pages 674–681

[47] SHARRETT, A. R. ; DING, J. ; CRIQUI, M. H. ; SAAD, M. F. ; LIU, K. ; POLAK, J. F. ; FOLSOM, A. R. ; TSAI, M. Y. ; BURKE, G. L. ; SZKLO, M.: Smoking, diabetes, and blood cholesterol differ in their associations with subclinical atherosclerosis: The Multiethnic Study of Atherosclerosis (MESA). *Atherosclerosis* 186 (2006), No. 2, pages 441–447

[48] STRAKA, M.: *Processing and Visualization of Peripheral CT–Angiography Datasets*, Vienna University of Technology, PhD Thesis, 2006

[49] TELEA, R. ; VILANOVA, A.: A robust level-set algorithm for centerline extraction. *In Proceedings of the symposium on Data visualisation 2003*, 2003, pages 185–194

[50] THORNTON, J. ; ALETICH, V. A. ; DEBRUN, G. M. ; ALAZZAZ, A. ; MISRA, M. ; CHARBEL, F. ; AUSMAN, J. I.: Endovascular Treatment of Paraclinoid Aneurysms. *Surgical Neurology* 54 (2000), No. 4, pages 288–299

[51] TODORAN, T. M. ; SOBIESZCZYK, P.: Catheter-Based Therapies for Massive Pulmonary Embolism. *Progress in Cardiovascular Diseases* 52 (2010), No. 5, pages 429–437

[52] VARCHOLA, A. ; VAŠKO, A. ; SOLČÁNY, V. ; DIMITROV, L. I. ; ŠRÁMEK, M.: Processing of Volumetric Data by Slice– and Process–Based Streaming. SLAY, H. (Ed.) ; BANGAY, S. (Ed.): *Afrigraph'07*. Grahamstown, South Africa : ACM Siggraph, 2007, pages 101–110

[53] WANG, H. H.: Analytical models of atherosclerosis. *Atherosclerosis* 159 (2001), No. 1, pages 1–7

[54] WELLS, P. S.: Pulmonary Embolism: A Clinician's Perspective. *Seminars in Nuclear Medicine* 38 (2008), No. 6, pages 404–411

Algorithms

Algorithm 1 hysteresisThreshold

```
1:  for all pixels p do
2:      if p ≤ low then
3:          p ← weak
4:      else if p > high then
5:          p ← strong
6:          FollowEdge(p)
7:      end if
8:  end for
9:  for all pixels p do
10:     if p is strong then
11:         p ← 1
12:     else
13:         p ← 0
14:     end if
15: end for
```

procedure `FollowEdge`(p)

```
1:  for all neighbors n of p do
2:      if n ≤ low then
3:          n ← weak
4:      else if n ≤ high then
5:          n ← strong
6:          FollowEdge(n)
7:      end if
8:  end for
```

Algorithm 2 hysteresisThresholdStreaming

```
1: classify all points as weak or strong, leave candidates
2: determine slice offsets
3: for all slices z do
4:    determine strong pixels in z, store them in LPL
5:    for all points p ∈ LPL do
6:       CPL ← p
7:       while CPL ≠ ∅ do
8:          pp ← first(CPL)
9:          GPPL ∪{pp}
10:         for all neighbors n of pp do
11:            if n is candidate and n ∉ GPPL and n ∉ CPL then
12:               CPL ∪{n}
13:            end if
14:         end for
15:      end while
16:   end for
17:   update pass // if necessary
18: end for
19: update pass // for sure
20: for all points p do
21:   if p is strong then
22:      p ← 1
23:   else
24:      p ← 0
25:   end if
26: end for
```

Algorithm 3 combine

```
1: for all slices s do
2:    for all scale volumes do
3:       read next slice
4:    end for
5:    for all pixels p ∈ s do
6:       maxSigma ← 0
7:       storeValue ← 0
8:       for all scale volumes v do
9:          if v(p) ≠ 0 then
10:            if maxSigma < sigma(v) then
11:               maxSigma ← sigma(v)
12:               storeValue ← v(p)
13:            end if
14:         end if
15:      end for
16:      out_p ← storeValue
17:      sigma_p ← maxSigma
18:   end for
19: end for
```

Algorithm 4 CenterlineReformation

```
1: for all selected vessel tree segments do
2:    // determine IBS from projected centerline
3:    projectCenterlineToImagePlane()
4:    BPQ ← IBS
5:    // determine projected centerline visibility in order to support segment self occlusions
6:    determineCenterlineVisibility()
7:    // propagate centerline pixels to off-centerline ones
8:    propagateCenterline()
9: end for
```

Procedure 5 projectCenterlineToImagePlane

1: **for all** control points p of segment s **do**
2: project p to image plane
3: center p on pixel
4: **if** $segment(p) = segment(\mathit{BPB}[p])$ **then**
5: **if** $depth(p) < depth(\mathit{BPB}[p])$ **then**
6: $\mathit{IBS} \leftarrow \mathit{IBS} \setminus \{p\}$
7: $\mathit{BPB}[p] \leftarrow p$
8: $\mathit{IBS} \cup \{p\}$
9: **end if**
10: **else**
11: $\mathit{BPB}[p] \leftarrow p$
12: $\mathit{IBS} \cup \{p\}$
13: **end if**
14: **end for**

Procedure 6 determineCenterlineVisibility

1: **for all** boundary points $p \in \mathit{IBS}$ **do**
2: **for all** neighbors $n \in \mathit{BPB}[N_8(p)]$ **do**
3: **if** $segment(p) = segment(n)$ **then**
4: **if** $distPI(p, n)$ > threshPI along the centerline **then**
5: determine if either n or p lies closer to the view plane
6: update occlusion candidates accordingly in the *BPB*
7: **end if**
8: **end if**
9: **end for**
10: **end for**

Procedure 7 propagateCenterline

```
1: while BPQ ≠ ∅ do
2:    remove first pixel p ∈ BPQ
3:    // update Front Propagation Buffer (FPB)
4:    if depth(p) < depth(FPB[p]) then
5:       if segment(p) = segment(FPB[p]) then
6:          if occCandidate(FPB[p]) and distPI(p, FPB[p]) > threshPI then
7:             FPB[p] ← p
8:          end if
9:       else
10:         FPB[p] ← p
11:      end if
12:   else if halo(p) = true then
13:      return
14:   end if
15:   // stop propagation, normal plus halo propagation done
16:   if radius(p) ≤ 0 and (halo(p) or haloWidth ≤ 0) then
17:      return
18:   end if
19:   // if normal propagation is done, start halo propagation if a halo is desired
20:   if radius(p) = 0 and halo(p) = false then
21:      radius(p) ← haloWidth
22:      halo(p) ← true
23:   end if
24:   // propagate to the neighbors of p now
25:   for all neighbors n ∈ BPB[N4(p)] do
26:      update data sampling position of p
27:      radius(p) ← radius(p) − 1
28:      if halo(p) = true then
29:         determine the halo position of p
30:         update depth of p
31:      end if
32:      if segment(p) = segment(n) then
33:         if depth(p) < depth(n) and occCandidate(n) and distPI(p, n) > threshPI then
34:            BPQ ← BPQ \ {n}
35:            BPB[n] ← p
36:            BPQ ∪ {n}
37:         end if
38:      else
39:         BPB[n] ← p
40:         BPQ ∪ {n}
41:      end if
42:   end for
43: end while
```