Otto-von-Guericke-University Magdeburg

Faculty of Computer Science

Department of Simulation and Graphics

Master Thesis

Smoke-Like Visualization of Aortic Dissection Hemodynamics

Aaron Schroeder

June 6, 2024

Supervisor

Prof. Bernhard Preim
Department of Simulation and Graphics
Otto-von-Guericke University Magdeburg

Advisor

Dr. Gabriel Mistelbauer
Department of Radiology
Stanford University School of Medicine

Statement of Authorship

I herewith assure that I wrote the present thesis independently, that the thesis has not been partially or fully submitted as graded academic work and that I have used no other means than the ones indicated. I have indicated all parts of the work in which sources are used according to their wording or to their meaning. I am aware of the fact that violations of copyright can lead to injunctive relief and claims for damages of the author as well as a penalty by the law enforcement agency.

Magdeburg, June 6, 2024

.....

Abstract

Aortic dissection is a life-threatening cardiovascular disease characterized by blood entering the media layer of the aortic vessel wall, creating a second flow channel. This new flow channel, called false lumen, not only weakens the vessel wall but can lead to various complications such as branch vessel malperfusion or even fatal aortic rupture. The tissue separating true and false lumen is called dissection flap and comprises one or more tears called fenetrations. Current risk stratification of aortic dissections mostly relies on morphologic features such as location and size of fenestrations, number of fenestrations, and aortic diameter. Increasingly, hemodynamic features such as flow rate and wall shear stress (WSS) are used in risk stratification. However, effective visualization of the complex hemodynamics of aortic dissections remains challenging. Flow visualization methods commonly used to show flow in vascular structures such as streamlines, pathlines, and streaklines suffer from occlusion and clutter. We investigate the use of a smoke-like rendering technique for the visualization of flow to mitigate these issues. *Smoke Surfaces* developed by von Funck et al. [1] are used to create a real-time visualization of aortic dissection hemodynamics. Their technique uses streak surfaces augmented with opacity modulations to create a smoke-like appearance and hide possible artifacts arising from the streak surface structure. We use aortic dissection datasets created through 2-way fluid-structure interaction (FSI) computational fluid dynamics (CFD) simulations, that provide detailed, high quality hemodynamic and morphologic data. To improve the visual fidelity when applying *Smoke Surfaces* to the complex hemodynamics of aortic dissections, we adapt the computation of the opacity terms. Additionally, hemodynamic measures are dispayed on the streak surfaces through color mapping. Aside from the commonly color mapped velocity magnitude, measures specific to the flow in aortic dissections, namely lumen of origin and retrograde flow, are displayed. Furthermore, a circular seeding structure mimicking the shape of the surrounding vessel is employed and the structure of the streak surface adapted accordingly. Von Funck et al. [1] proposed a tubular topology for the streak surface mesh, whereas we use a toroidal topology. To provide valuable context to the visualization of flow, the aortic vessel wall is rendered transparent with a Fresnel effect and the dissection flap is rendered opaque. We demonstrate the visualization results by comparing *Smoke Surfaces* to streaklines. Both techniques are applied to two aortic dissection datasets.

Kurzfassung

Aortendissektion ist eine lebensbedrohliche Herz-Kreislauf-Erkrankung, die dadurch gekennzeichnet ist, dass Blut in die Mediaschicht der Aortenwand eindringt und einen zweiten Flusskanal bildet. Dieser neue Flusskanal, genannt falsches Lumen, schwächt nicht nur die Gefäßwand, sondern kann auch zu verschiedenen Komplikationen wie Malperfusion von Seitenäste oder sogar fatalem reißen der Aorta führen. Das Gewebe, welches das wahre und das falsche Lumen trennt, wird Dissektionsmembran genannt und umfasst einen oder mehrere Risse, die als Fenestrationen bezeichnet werden. Die aktuelle Risikostratifizierung von Aortendissektionen basiert hauptsächlich auf morphologischen Merkmalen wie Lage und Größe der Fenestrationen, Anzahl der Fenestrationen und Aortendurchmesser. Zunehmend werden hämodynamische Merkmale wie Flussrate und Wandschubspannung (WSS) in der Risikostratifizierung verwendet. Die effektive Visualisierung der komplexen Hämodynamik von Aortendissektionen bleibt jedoch eine Herausforderung. Die üblicherweise verwendeten Methoden zur Strömungsvisualisierung in Gefäßstrukturen wie Stromlinien, Pfadlinien und Streichlinien leiden unter Verdeckung und Überladung. Wir untersuchen die Verwendung einer rauchähnlichen Rendertechnik zur Visualisierung der Strömung, um diese Probleme zu mildern. *Smoke Surfaces*, entwickelt von von Funck et al. [1], werden verwendet, um eine Echtzeit-Visualisierung der Hämodynamik von Aortendissektionen zu erstellen. Ihre Technik verwendet Streichflächen, die durch Transparenzsmodulationen erweitert wurden, um ein rauchähnliches Erscheinungsbild zu erzeugen und mögliche Artefakte, die aus der Struktur der Streichfläche resultieren, zu verbergen. Wir verwenden Aortendissektionsdatensätze, die durch numerische Strömungssimulationen mit 2-Wege-Fluid-Struktur-Kopplung erstellt wurden und detaillierte, hochwertige hämodynamische und morphologische Daten liefern. Um die visuelle Genauigkeit bei der Anwendung von *Smoke Surfaces* auf die komplexe Hämodynamik von Aortendissektionen zu verbessern, passen wir die Berechnung der Transparenzparameter an. Zusätzlich werden hämodynamische Messwerte durch Farbkodierten auf den Streichflächen dargestellt. Neben der üblicherweise farbkodierten Geschwindigkeit werden zwei spezifische Messwerte des Flusses bei Aortendissektionen, Ursprungs-Lumen und retrograder Fluss, angezeigt. Darüber hinaus wird eine kreisförmige Saatstruktur verwendet, welche die Form des umgebenden Gefäßes nachahmt, und die Struktur der Streichfläche entsprechend angepasst. Von Funck et al. [1] zeigten eine röhrenförmige Topologie für das Streichflächen-Mesh, während wir eine toroidale Topologie verwenden. Um einen wertvollen Kontext zur Visualisierung des Flusses zu bieten, wird die Aortenwand transparent mit einem Fresnel-Effekt gerendert und die Dissektionsmembran undurchsichtig dargestellt. Wir demonstrieren die Visualisierungsergebnisse, indem wir *Smoke Surfaces* mit Streichlinien vergleichen. Beide Techniken werden auf zwei Aortendissektionsdatensätze angewendet.

Acknowledgements

We thank Kathrin Bäumler (3D and Quantitative Imaging Laboratory, Department of Radiology, Stanford University) for providing the aortic dissection datasets.

Contents

List of Figures

Acronyms

AABB axis-aligned bounding box. 27
AS acceleration structure. 27, 28, 31, 32

BLAS bottom-level acceleration structure. 27, 31
BVH bounding volume hierarchy. 27

CFD computational fluid dynamics. 3, 5, 6, 18, 40
CPU central processing unit. 9, 31
CT computed tomography. 5, 12
CTA computed tomography angiography. 3, 18

DVR direct volume rendering. 8

FPS frames per second. 43
FSI fluid-structure interaction. 18, 40

GB gigabyte. 21, 37
GPU graphics processing unit. 9, 11, 12, 21, 23, 26, 43

MIP maximum intensity projection. 3, 5
MPR multiplanar reformation. 5
MRI magnetic resonance imaging. 2, 3, 5–10, 12, 13, 18

TEVAR thoracic endovascular aortic repair. 6
TLAS top-level acceleration structure. 27

WSS wall shear stress. 6, 7, 18, 19

1 Introduction

Cardiovascular diseases like aneurysms and aortic dissection pose a significant health risk for patients. The formation of these diseases is closely related to local hemodynamics as they are accelerated by increased pressure inside vessels. After and during their development, aortic dissections impact the local blood flow, often leading to more complications. Detailed analysis of the anatomy and hemodynamics is challenging as the specific configuration of a dissection varies greatly between patients.

1.1 Aortic Dissection

Aortic dissection is a cardiovascular disease characterized by the formation of a second flow channel inside the aorta. The aortic vessel wall is composed of three layers. The innermost layer is called tunica intima and is in direct contact with the blood carried by the vessel. The outermost layer is called tunica adventitia and anchors the vessel to the surrounding tissue. In between these two layers there is a media layer, which provides support to the vessel.

An aortic dissection is caused by blood entering into the media layer of the aortic vessel wall and carving a second flow channel. This phenomenon is depicted in Figure 1.1. The new flow channel is called the false lumen and runs parallel to the original flow channel of the aorta, called the true lumen. The extent of the false lumen varies greatly between patients, with the dissection starting at a so-called entry tear, extending distally and ending at one or more exit tears. After the formation of an aortic dissection both true and false lumen carry blood to varying degrees. The two flow channels are separated by a thin layer of the aortic vessel wall, the tunica intima. The section of the intima separating true and false lumen is called the dissection flap and commonly comprises two or more tears also called intimal tears. These are the primary entry and exit tears, while additional intimal tears may be present, which are commonly referred to as fenestrations.

Aortic dissections are rare with an incidence of 15 in 100,000 patient years and in-hospital mortality of 39% [6], most often occurring in patients between the ages of 65 and 75 [7]. The disease is life-threatening [8] requiring patients to be under constant clinical monitoring to prevent complications including pericardial tamponade, branch vessel malperfusion or even aortic rupture [9]. Any of these complications may require urgent surgical or endovascular treatment.

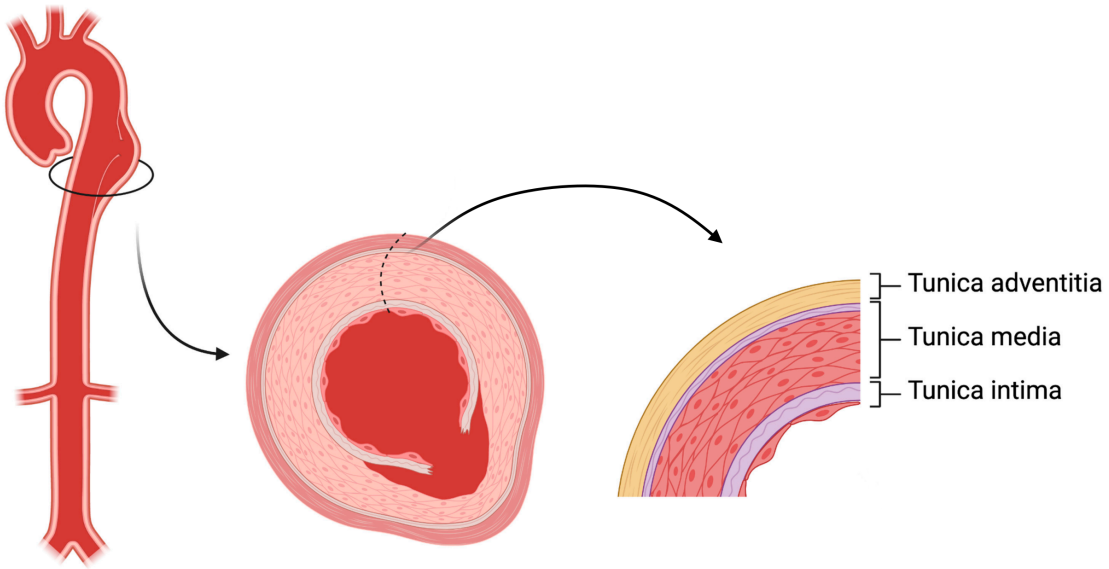


Figure 1.1: The structure of the aortic wall. In aortic dissection, a tear in the intima layer results in blood entry to the media layer, developing in an intimal flap and dividing the original vessel into true and false lumen. Illustration from Yuan et al. [5]

1.2 Diagnosis, Prognosis and Treatment

Currently, the classification, treatment and prognosis of aortic dissection is mostly informed by morphological features captured on imaging studies [10]. However, phantom studies suggest, that true lumen collapse and branch vessel malperfusion are related to number, size and location of intimal tears as well as the distribution of branch vessels draining the false and true lumen [11, 12]. Also, increased false lumen pressure may relate to false lumen dilation [13, 14], which can culminate in aortic rupture. At the same time, limited false lumen outflow has shown to contribute to overall disease progression [15, 16, 17, 18]. These findings show that hemodynamics of aortic dissection impact disease progression. As a result, detailed analysis of hemodynamic information may help inform treatment and prognosis of aortic dissection, ultimately improving patient quality of life.

1.3 Flow Simulation

Measurements of in vivo blood flow have become part of clinical routine since the introduction of 2D phase contrast-magnetic resonance imaging (MRI) in the late 1980s [19]. More recently, 4D flow MRI providing time-resolved three-dimensional velocity encoding has been developed and enabled comprehensive evaluation of complex blood flow patterns. The 4D flow MRI studies performed in order to analyze patient-specific blood flow yield large amounts of measured flow data. This data presents with a plethora of information about aortic dissection hemodynamics such as stroke volume, primary entry tears, helical flow, and velocity. These measurements are related to the rate of aortic expansion [20]. However, such data is associated with noise and artifacts while missing an important measurement: pressure.

Using patient-specific morphology captured using e.g. computed tomography angiography (CTA) and flow data captured using 4D flow MRI, flow simulations can be performed. The resulting data is not only of high quality, but can also contain a multitude of information about the simulated flow, which is not accessible through MRI imaging. Most importantly computational fluid dynamics (CFD) simulations allow for the accurate calculation of pressure inside a fluid. As mentioned before, pressure inside the aorta, especially the false lumen, can lead to false lumen dilation [13, 14] and also to flow obstruction of the true lumen by the dissection flap [21].

1.4 Flow Visualization

While the measurement of aortic blood flow has been possible for decades, clinical practice mostly relies on morphological features and only employs well-known flow visualization techniques such as maximum intensity projection (MIP), arrow glyphs, streamlines, and pathlines. However, these techniques are often limited in terms of information density, perceptual effectiveness, occlusion, and clutter.

Both MRI imaging and CFD simulations result in large datasets, which include critical information for both diagnosis and treatment planning. Due to the sheer size of the datasets and the complex phenomena they encode, efficient and meaningful visualization is challenging. Aortic dissection presents with complex pathologies also leading to complex flow patterns. Varying location, extent and curvature of the dissection flap as well as number, location and size of intimal tears can create vastly different hemodynamics.

Traditional means of flow visualization often do not suffice for such complex data, creating the need for visualization techniques tailored to the analysis of hemodynamics in and around intricate pathologies. Visualizations enabling the assessment of blood flow in two parallel channels while highlighting hemodynamic features like flow direction and dissection flap movement could not only help researchers improve their understanding of the development of aortic dissections, but also impact treatment development and selection.

1.5 Goal of the Thesis

Virtual flow visualization is fundamentally inspired by real-life experimental flow visualization methods, which typically entail introducing a medium such as smoke or dye into a flow and observing its advection. This methodology also finds application in contrast-enhanced angiography, where a contrast agent is introduced into the blood flow to highlight vessels in non-invasive medical imaging procedures. The contrast agent is injected at a single location and spreads through the vascular structure as it is advected by the flow.

Our objective, is to replicate this process familiar to radiologists and physicians by simulating the injection of smoke or dye into the blood flow. This goal can be achieved through the utilization of various techniques.

Commonly, rendering of individual particles is utilized to simulate highly realistic smoke. However, this technique necessitates the advection and rendering of very large numbers of particles to be effective, which does not lend itself to real-time flow visualization.

An alternative approach tailored specifically for real-time flow visualization was introduced by von Funck et al. [1]. They employ streak surfaces to represent the flow and modulate the surface's opacity to create a smoke-like appearance. Their method, termed *Smoke Surfaces*, offers a balance between realism and computational efficiency, making it suitable for real-time visualization applications. The presented visualization technique was not tested on any blood flow datasets.

Given the intricate and distinct geometry inherent in aortic dissections, our objective is to employ *Smoke Surfaces* for visualizing dissection hemodynamics. Meanwhile, possible challenges caused by the parallel flow channels separated by a thin wall and multiple connections between the flow channels need to be addressed.

To enhance the information density and visual fidelity of the visualization, we plan to employ color mapping and additional opacity modulation. Our aim is to implement and test the encoding of multiple measures such as velocity through color mapping.

1.6 Structure of the Thesis

In the following chapter we begin by reviewing the state-of-the-art regarding the evaluation and risk stratification of aortic dissections as well as medical flow visualization. We also discuss different techniques for the smoke-like visualization of flow.

Subsequently, we give a comprehensive overview of the preparation of the medical flow data and important implementation details. Furthermore, the visualization technique used to produce an interactive smoke-like visualization of flow inside aortic dissections is discussed, including our augmentations and contributions to the technique.

Afterwards, the results are presented and compared to a widely used flow visualization technique. Several examples are shown, demonstrating the visualization's effectiveness. Lastly, we give a short conclusion and discuss potential future work.

2 Related Work

Medical visualization techniques have revolutionized the field of healthcare and education, providing a more meaningful and engaging way to visualize the life sciences. These techniques include surface and volume rendering, as well as interaction techniques to adjust these visualizations with appropriate transfer functions [22]. In addition, advanced visualization techniques such as multiplanar reformation (MPR), MIP, and 3D rendering are now a standard-of-care in the interpretation of medical images in many clinical scenarios [22].

Medical flow visualization techniques, on the other hand, are a subset of medical visualization techniques that focus on visualizing fluid flow, such as blood flow in the human body. These techniques are particularly useful in diagnosing and treating cardiovascular diseases [23] and mostly based on either MRI measuremtens or CFD simulations [24]. Specialized techniques for flow visualization of specific vascular structures such as the aorta [25] or carotid arteries [26] and specific diseases such as aneurysms [27, 28] were presented in the past. Despite these advancements, there remain areas, such as the visualization of aortic dissections, where specialized techniques are still scarce.

Little work has been done on techniques tailored to the visualization of the anatomy and blood flow within aortic dissections. Ostendorf et al. [29] made a first attempt to assess shading styles for the rendering of multiple vessel wall surfaces, specifically the outer vessel wall and dissection flap. We use their findings to render the aortic vessel wall and dissection flap. While multiple approaches for the visualization of anatomy and hemodynamics have been presented in the past, they were not necessarily designed for the visualization of aortic dissections, which presents unique challenges. Related work on this topic is covered in the following sections.

2.1 Morphological and Hemodynamic Evaluation of Aortic Dissection

Aortic dissection poses significant morbidity and mortality risks, underscoring the critical importance of selecting appropriate treatment strategies. This selection process relies on extracting a multitude of features from standard anatomical imaging modalities such computed tomography (CT) and MRI.

Recent research has increasingly focused on understanding the relationship between aortic hemodynamics, particularly those of the false lumen, and adverse aortic remodeling. However, despite these advancements, there remains a need for enhanced risk stratification [30]. Noninvasive techniques for measuring and analyzing false lumen

hemodynamics hold promise in improving risk stratification, offering potential avenues for further research and clinical application.

2.1.1 Morphological Risk Stratification

Noninvasive imaging is a key tool for diagnosis and risk stratification in aortic dissection. Multiple morphological features have been studied in an effort to improve risk stratification. Maximum aortic diameter is the largest diameter of the dissected aorta measured in a cross-section perpendicular to the aortic centerline. An increased maximum aortic diameter has been shown to predict aortic complication in several studies [10]. Both the size of the false lumen and its size relative to the true lumen have been studied, showing increased false lumen diameter to predict late aneurysmal degeneration [31]. Similarly, a later study found increased false lumen area to be associated with higher incidence of complications such as limb ischemia, progression of dissection and aortic rupture [32].

The ratio between true lumen and false lumen volume was investigated by Lavingia et al. [33]. They found that lower relative volume of the true lumen was highly predictive of patients requiring aortic intervention, while higher true lumen volume compared to the false lumen was highly predictive of freedom from delayed operation.

The location of the primary entry tear can be identified in most patients [30], and some authors have suggested that location and size of the primary entry tear correlate with adverse events. A study found entry tear size to be predictive of dissection-related adverse events and mortality [34]. They also found proximal location of the primary entry tear to be a significant predictor of complications. The number of intimal tears was shown to be related to aortic growth, with a single tear showing more aortic growth than zero or multiple tears [35].

2.1.2 Hemodynamic Risk Stratification

False lumen pressurization seems to be associated with adverse false lumen remodeling and complications [36] as the main treatment methods of aortic dissection, antihypertensives, reduce mean arterial pressure [37] and thoracic endovascular aortic repair (TEVAR) limits false lumen inflow [38]. Primarily two methods, namely CFD and 4D flow MRI are used to study the complex interplay of multiple hemodynamic features in aortic dissection.

CFD has been used to study factors that contribute to false lumen outflow resistance and the aortic consequences. Cheng et al. [39] found a larger primary entry tear to cause increased false lumen flow rate. Elevated percentage of total aortic flow passing through the false lumen was found to increase risk of aneurysm formation [40]. Furthermore, low wall shear stress (WSS) showed to be associated with increased false lumen thrombus formation [41], while elevated WSS showed to be associated with aortic growth [40, 42]. Finally, the pressure gradient between true and false lumen was shown to be correlated to false lumen dilation [39, 43], tear propagation, and branch vessel malperfusion [44, 45].

4D flow MRI is used for both quantitative and qualitative measurements of aortic dissection hemodynamics. Using this technique further findings on hemodynamic risk factors were made. Allen et al. [46] showed a correlation between primary entry tear size and the amount of false lumen flow occurring both at its peak and averaged across the cardiac cycle. Two studies showed that retrograde flow is more prevalent in the false lumen than in the true lumen [21, 47]. Furthermore, increased false lumen stroke volume (the sum of the average velocity within a false lumen cross-section multiplied by the cross-sectional area for all time steps) showed association with more rapid aortic growth [20].

2.2 Medical Flow Visualization

Flow Visualization is a well established area of scientific visualization, which has become invaluable to many fields, such as automotive and aerospace design, meteorology and medical imaging. Dense 2D flow fields can be effectively visualized using texture-based techniques. These techniques are well-known and can be extended by color mapping and glyphs to encode the direction of the flow and additional features [48]. Hemodynamic flow data typically consists of 3D vector fields, with simulated datasets providing additional measures like pressure or WSS. Visualizing 3D flow data poses significant challenges beyond simply handling the large volume of data. Occlusion and clutter are common issues in 3D flow visualizations, primarily due to the dense and overlapping nature of structures like streamlines. Moreover, encoding various measures such as velocity or pressure, along with the dynamic behavior of the flow, introduces additional complexity to the visualization process. Although a multitude of techniques have been presented to cope with the increased complexity and size of flow datasets enabled by modern hardware, extracting detailed information from flow data of highly complex nature like of aortic dissections requires specially tailored visualizations.

In an attempt to create more expressive flow visualizations, a branch of visualization approaches inspired by handcrafted illustrations emerged. These illustrative visualizations aim to maximize the amount of information communicated in a comprehensive way through the use of visual abstraction techniques. Brambilla et al. [49] provide an overview of illustrative flow visualization techniques. Born et al. [50] presented an approach to visualizing aortic and cardiac blood flow by combining multiple techniques. They use streamlines rendered as bundles of tapes to reduce visual clutter. A set of line representatives is selected to reflect the most important flow aspects, further decreasing the number of lines. In addition, vortices are highlighted using hatched tube-like structures.

2.3 Focus and Context Visualization

In addition to showing hemodynamics, we display the anatomy of the surrounding vessel. As a result, even more structures need to be visualized simultaneously while minimizing visual overload for the user. This problem is typically addressed by using a visualization technique called Focus and Context visualization, where main structures are highlighted for better visibility, while maintaining a view of the surroundings to provide essential

context. This technique plays a crucial role in enhancing the comprehensibility of complex visualizations.

Focus visualization involves emphasizing specific structures or regions of interest within the visualization to draw the viewer's attention and facilitate a more detailed examination. This can be achieved through techniques such as color-coding, outlining, or magnifying the relevant structures. On the other hand, context visualization aims to provide a broader overview of relevant local structures. This is crucial for ensuring that users can understand the relationships between different structures and appreciate the overall complexity of the system. Different approaches to focus and context visualizations of anatomical structures have been presented in the past, as discussed in the subsequent paragraphs.

Blood flow inside aneurysms in conjunction with the aneurysm surface was visualized by Gasteiger et al. [27]. They used color-coded streamlines to create focused visualizations of flow and applied a ghosted view approach to the aneurysm surface as context. Front facing surfaces are rendered transparently with a Fresnel effect, while backfaces are opaque. Atmospheric attenuation is used to fade out objects farther from the viewer and therefore improve depth perception.

Lawonn et al. [51] proposed an approach using a similar ghosted view and animated pathlines. Color coding on pathlines is used to display information, such as curvature or vorticity. Vessel surfaces are also color-coded, but use a 2D texture lookup table to display both surface curvature and proximity to pathlines. Again, the technique focuses on visualizing flow, but supplies important context in the form of the vessel surface. This technique was also applied to aneurysm datasets.

Köhler et al. [25] developed a tool for visual exploration and analysis of 4D phase contrast MRI data. With a focus on detecting and characterizing vortices, they display flow using pathlines, augmented by different color mappings and rendering styles. To reduce clutter, they filter pathlines according to different criteria. Together with direct volume rendering (DVR) of the MRI data as context, they create a combined visualization of both flow and anatomy. Additionally, different attributes, such as velocity, axial velocity and rotation direction, are displayed through minimum/maximum intensity projection.

Behrendt et al. [28] presented an approach similar to Lawonn et al. [51] for the visual exploration of intracranial aneurysms, which employs a combination of the Fresnel effect and traditional Phong lighting. In addition, the pathlines used to show flow are seeded using evolutionary algorithms based on user-specified regions of interest on the vessel wall. This reduces clutter caused by unwanted pathlines, while also reducing undersampling in important areas.

All of this previous work relies on integral lines for the display of flow. Approaches that aim to create a more realistic visualization of flow through the emulation of smoke are discussed in the next section.

2.4 Smoke-Like Flow Visualization

Instead of taking inspiration from illustrated depictions of flow, one can also take inspiration from flow appearing in the real world. By taking inspiration from experimental flow visualization, which uses injection of particles into the flow, realistic, detailed visualizations can be created. Typically, these visualizations mimic the injection and advection of smoke or dye into the flow, but simulations of experimental techniques like wool tufts are also possible. The realistic depictions of flow created using these techniques can aid in the intuitive understanding of complex flow dynamics and the interpretability of those visualizations.

The line-based techniques for the visualization of 3D flow data mentioned in the previous sections do not lend themselves well to creating a smoke-like appearance. Instead, there are different techniques, which can be used to simulate smoke being advected by a flow.

2.4.1 Particle-based techniques

Probably the most natural approach to visualizing smoke is to simulate it, as it appears in the real world using individual particles. To achieve a convincing approximation of real-world smoke, large amounts of particles need to be created, advected and subsequently rendered. Each particle provides a negligible contribution to the entire smoke visualization, but the sum of all particles is able to create a complex visualization showing smoke motion as well as density. Because of the small contribution of each particle, particle counts of multiple millions are not uncommon for a visualization to appear convincing. Such high numbers of particles require lots of processing power and specialized algorithms to be advected and rendered efficiently. The first particle systems could not be rendered interactively even for smaller numbers of particles.

One of the first interactive particle systems was proposed by Krüger et al. [52], making use of modern graphics processing units (GPUs). Interactive frame rates are achieved by performing particle advection on the GPU and rendering them using techniques by Kipfer et al. [53]. All the while expensive transfer operation between central processing unit (CPU) and GPU are avoided. Using Krügers approach, Van Pelt et al. [2] present an early application of particle-based flow visualization to measured aortic blood flow. They also demonstrate the use of pathlines and streak surfaces to show flow inside the aorta captured using 4D flow MRI (see Figure 2.1).

Multiple different approaches for the rendering of particles were developed, aiming to improve the appearance of smoke as well as reduce the number of particles required to achieve a convincing effect. Those techniques include surface-particles [54], which are rendered as points with normals, semi-transparent billboards [55] and point sprites [56].

Particle systems, while providing an accurate simulation of real-world smoke, suffer from possible over-/undersampling of specific areas. As with other flow visualization techniques such as streamlines or pathlines, the number and location of seeding positions is crucial for the effectiveness and efficiency of a visualization. There is always a trade-off between sparse and dense seeding. Sparse seeding, while producing less cluttering, can

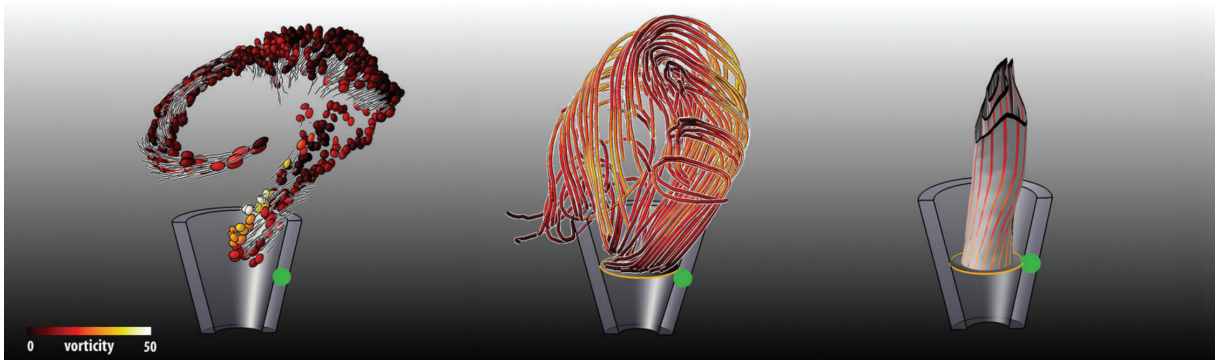


Figure 2.1: Blood flow visualization by van Pelt et al. [2] using illustrative particles (left), pathlines (center) and streak surfaces (right). © 2011 IEEE

lead to important features being missed. Dense seeding may capture more features, but necessarily increases clutter. However, dense seeding does not guarantee that no features are missed, as particles tend to accumulate in regions of slow or converging flow.

Engelke et al. [57] proposed a new type of particle system consisting of autonomous particles. Particles are terminated or split based on user defined criteria, controlling the density of particles in specific areas. This reduces cluttering caused by large amounts of particles in less important areas and ensures sufficient sampling of regions of interest.

Aiming to simulate the injection of particles into the blood flow, De Hoon et al. [58] created flow visualizations of aortic flow based on phase-contrast MRI data. A smoke-like appearance was achieved by advecting numerous particles and rendering them semi-transparently. While particle counts as high as 2 million where tested, interactive frame rates could be achieved using up to 400,000 individual particles. Color mapping was applied to the particles, which encoded seeding position or particle age but no hemodynamic properties.

Although high numbers of particles simulate the appearance of smoke quite well, advecting millions of particles is computationally expensive. Lowering the number of particles increases performance but also destroys the illusion of smoke. To mitigate this issue, particles can be connected to form streak surfaces.

2.4.2 Streak Surface Techniques

Streak surfaces are part of the group of integral surface structures. Stream surfaces, being the equivalent surface structures to streamlines, have been used extensively in flow visualization [59, 60, 61, 62]. Starting at a polygonal seeding structure, new vertices are introduced while integrating over a vector field. The front of the surface, which results from connecting these vertices to a mesh, constantly advances. This front may need to be adaptively modified however, as converging or diverging flow can lead to too large or too small distances between vertices. To mitigate these over-/undersampling artifacts, either new vertices need to be inserted into the mesh or existing vertices need to be collapsed. In specific cases, such as the flow being parted by an obstacle, the surface may even need

to split into multiple parts, necessitating splitting the frontline and tracing multiple parts independently. In unsteady flows, both stream and path surfaces can be created similarly to the steady case.

Streak surfaces on the other hand are created by periodically emitting vertices at the seeding polygon and continuously advecting them. While stream and path surfaces are built by only updating the frontline, all vertices of the streak surface need to be advected simultaneous. Therefore, structural changes such as collapse or splitting, which could previously only occur at the frontline, may occur at any place on the surface, requiring constant checking of all vertices. These expensive operations limited the use of streak surfaces in interactive application for a long time.

The first approach to making streak surfaces interactive was made by Von Funck et al. [1]. They presented a technique based on streak surfaces, which simulates smoke through clever opacity modulation of the surface and omitting any restructuring. Multiple factors including the quality and size of the triangles in the mesh are combined to create a smoke-like appearance and hide parts of the mesh, which are deformed too much as a result of advection. Although, it could increase the information density of the visualization, the authors did not test any color mapping on the streak surfaces.

Shortly after, Bürger et al. [63] were the first to produce interactive streak surface visualizations including proper restructuring of the mesh. They achieved this by performing all computations on the GPU and two different approaches to generating a streak surface. The first techniques improved parallelization by using a patch-representation that avoids interdependence between patches. The second approach used a particle-based surface representation including particle refinement and coarsening. Both techniques produced interactive streak surfaces, while the second approach resulted in better performance and reduced artifacts. Unfortunately, the second approach does not allow for a dynamically moving seed polygon. Therefore, Ferstl et al. [64] adopted the patch-based approach for their generation of interactive separating streak surfaces. They place the seeding structure at separation locations, so that the generated streak surface shows separating profiles and allows the user to detect unstable manifolds.

Focusing on the rendering of integral surface structures, Hummel et al. [3] created an illustrative rendering approach. They employ multiple techniques to improve the rendering of stream, path and streak surfaces. Transparency is used to allow the user to see multiple overlapping layers of the surface. As this introduces more clutter, careful modulation of surface opacity is important. They provide two different transparency modulation schemes. Additionally, silhouettes are used to reveal the shape and highlight creases in the surface. Lastly, different stripe patterns are used to texture integral surfaces, further improving the display of shape and curvature of the surface. The entire collection of rendering techniques is compatible with interactive visualization of integral surfaces and does not require expensive preprocessing. An example of the possible visualization results is shown in Figure 2.2.

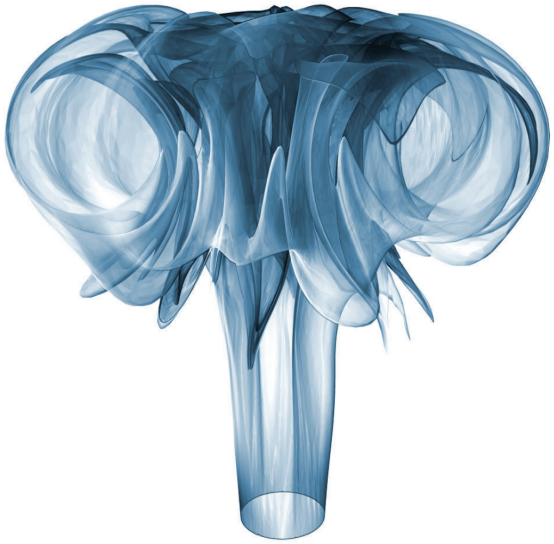


Figure 2.2: A rising plume streak surface rendered using techniques presented by Hummel et al. [3]. © 2010 IEEE

Recently, Schindler et al. [65] applied the transparency modulation presented by von Funck et al. to animated streak surfaces. They use *Smoke Surfaces* for the comparison of multiple trajectories emerging from different initial conditions of a 4D biological dynamical system.

2.4.3 Volumetric Techniques

With the advances in computer hardware, real-time visualizations of gaseous phenomena became possible through the use of volumetric approaches. Instead of rendering geometric primitives, the rendering is performed using ray casting, where rays passing through volumetric data are sampled at regular or adaptive intervals to produce an image. This rendering technique is also widespread among medical imaging, as the slice-based volumetric data, volume rendering relies on, is directly supplied by imaging techniques such as MRI and CT.

One of the first volume renderings of gaseous phenomena was produced by Schpok et al. [66], who rendered realistic animated clouds. They performed the necessary complex computations in parallel on the GPU. Using features of modern graphics processors such as 3D textures, geometry shaders and compute shaders, real-time realistic renderings of volumetric smoke, fire and water were made possible [67]. Later, interactive smoke rendering incorporating dynamic environmental lighting using compensated ray marching was presented by Zhou et al. [68].

In the field of flow visualization, flow volumes [4] were introduced as the volumetric counterpart to streamlines. This technique allows for interactive exploration of vector fields through the use of volumetric tetrahedral meshes, which are rendered transparently. Instead of seeding at individual locations and advecting to form integral lines, like with streamlines, flow volumes are formed by seeding at a mesh and advecting the entire mesh. Similar to the creation of stream surfaces, new vertices of the flow volume are periodically created as the mesh front is advected. The resulting segments are subdivided

Figure 2.3: Flow Volume visualization presented by Max et al. [4]. © 1993 IEEE

into tetrahedra before being rendered using volume rendering, as seen in Figure 2.3. During the ray casting, no large volume, which may only contain important information in small areas needs to be traversed. Instead, only the flow volume itself needs to be considered, while the remaining space can be skipped.

2.5 Advection Performance

Yenpure et al. [69] provide an overview of state-of-the-art techniques to increase the performance of particle advection for flow visualizations. Cell localization and access performance are important when trying to achieve interactive frame rates while performing live advection. The efficiency with which this process can be performed is governed by the structure of the flow data. While uniform, structured data typically enables quick cell localization, unstructured data often requires specialized techniques to be accessed efficiently. Medical flow data often originates from MRI scans which provide a vector field in a structured grid format. When using this data directly like in InkVis developed by De Hoon et al. [58], it can be easily used on a GPU. The uniform structure of the voxel data also allows for cell localization in constant time, improving overall advection performance. When flow visualizations need to be performed in unstructured data, different techniques can be used to speed up the cell localization process.

One approach is to organize cells of the flow data in a hierarchical data structure. Commonly octrees are used for this purpose [70, 71]. Similar to octrees, kd-trees allow for hierarchical subdivision of space, but do so while facilitating non-uniform subdivisions. This is useful when cell density varies greatly in the original data, but kd-trees introduce more storage overhead compared to octrees. Andrysco and Tricoche [72] presented an efficient storage scheme for kd-trees and octrees to address this issue. Their *Matrix *Trees* store tree levels in compressed sparse rows removing most of the memory overhead introduced by kd-trees. Vectorization of non-uniform hierarchical subdivision is challenging, but possible on modern hardware with good performance as shown by Garth and Joy [73].

Another approach to speeding up the localization of cells in unstructured data is successive neighbor search. Particle advection relies on repeated interpolation in cells of the flow data grid, which are typically close. As a result, particle advection can be sped up by searching the cell which contains the current location through a neighborhood search starting with the cell used during the last integration step. This technique works for all integration steps apart from the first especially with small step lengths. Bußler et al. [74] even presented a GPU-based successive neighbor search on tetrahedral cells, which could perform both the initial cell localization and subsequent searches without interaction of the CPU. Their algorithm was heavily based on the prior work of Schirski et al. [75].

3 Methods

Our previous work, covering the simultaneous display of flow and anatomy is now being extended through the use of *Smoke Surfaces* in the context of this thesis. The previously used pathlines lead to a lot of clutter and occlusion when many lines are seeded. *Smoke Surfaces* on the other hand consist of continuous streak surfaces augmented by opacity modulation to create a smoke-like appearance. Therefore, this work mainly focuses on the utilization of *Smoke Surfaces* developed by von Funck et al. [1] to visualize blood flow in aortic dissection. Simultaneously the outer vessel wall and the dissection flap need to be shown, while they should not obstruct the view of the flow.

The outer vessel wall is rendered semi-transparently with a Fresnel Effect [27]. This allows the user to look inside the vessel but still shows the extent and important features of the vessel. The dissection flap is rendered opaque so as to not create too many transparent layers and simultaneously give a detailed look at the shape and curvature of the flap. Multiple hemodynamic and derived measures can be mapped onto the vessel wall and flap surfaces through color. The surfaces are also animated, showing the movement of the vessel.

All aspects of our visualization including display of aortic dissection morphology, flow visualization, and color mappings were developed in close collaboration with cardiovascular imaging scientists, radiologists, and clinicians with experience in the treatment of aortic dissections.

The desired visualization was implemented in the *Visician* framework for medical visualizations. It provides tools and techniques for the display of flow data of aortic dissections. A renderer, which enables the effective visualization of both hemodynamics and vessel anatomy was already implemented.

3.1 Smoke Surfaces

Streamlines and pathlines are most commonly used when visualizing hemodynamic flow. Both line structures represent the movement path of a massless particle advected by the flow. Additionally, there are streak lines, which are created by continually seeding particles at a single location and connecting the advected points. Streaklines most accurately simulate experimental flow visualization carried out using the advection of smoke or dye. Similar to these integral line structures, integral surfaces can be created. Analogous to streak lines, streak surfaces result from continually seeding multiple particles at once. All particles are connected to form the streak surface mesh.

Because every vertex of the mesh is individually advected, they can move arbitrarily. This can lead to triangle cells of the mesh being deformed and becoming too large (undersampling of an area), too small (oversampling of an area), or of poor quality, i.e.

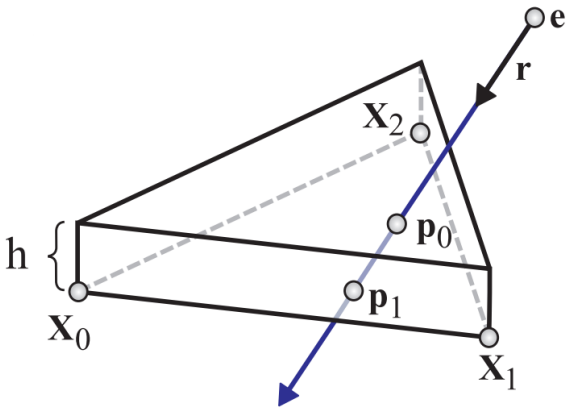
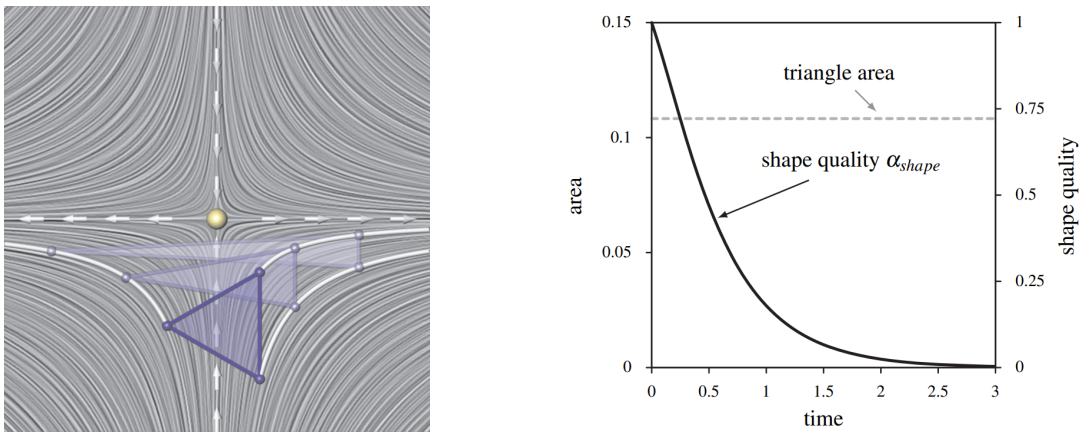


Figure 3.1: Conceptual prism used to compute $\boldsymbol{\alpha}_{density}$. Smoke density is simulated by conceptually intersecting the view ray along $\mathbf{r}$ with a prism of height h over a triangle cell of the streak surface mesh. With shallower angles between triangle cell and view ray, the distance between the intersection points p_0 and p_1 increases and more of the simulated smoke is traversed by the view ray. © 2008 IEEE

having very acute internal angles. To mitigate this, the distance between mesh vertices of a streak surface need to be checked to detect, where vertices may need to be inserted or joined in every advection step. However, accurately inserting a vertex requires the vertex to be inserted at the seeding location and advecting it until it matches the advection time of the rest of the mesh. This can occur at multiple locations within the mesh at once. Afterward, the mesh needs to be triangulated again to include potentially newly inserted vertices and creating high quality triangles. This process is very costly and is not beneficial to the real-time integration of streak surfaces.

Von Funck et al. [1] circumvent the process of inserting or joining vertices, by retaining constant connectivity between vertices of the mesh and modulating the opacity of severely deformed triangle cells. Using this method, no vertices need to be inserted or joined. Also, there is no need for retriangulation, as the mesh connectivity is set up only once in the beginning. Multiple opacity factors contribute to the smoke-like appearance and avoiding artifacts. The opacity of each mesh vertex is calculated as follows.

$$
\begin{aligned}
\boldsymbol{\alpha}_{density} &= \frac{k}{area\,(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2)\ cos\boldsymbol{\gamma}}, \\
\boldsymbol{\alpha}_{shape} &= \left(\frac{4\ area\,(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2)}{\sqrt{3}\ max\{d_0 d_1, d_1 d_2, d_2 d_0\}} \right)^s, \\
\boldsymbol{\alpha}_{curvature} &= 1 - b \cdot max\{|\boldsymbol{n}_0 \boldsymbol{e}_i| : i = 1..valence\,(\boldsymbol{x}_0)\}, \\
\boldsymbol{e}_i &= \frac{\boldsymbol{x}_i - \boldsymbol{x}_0}{\|\boldsymbol{x}_i - \boldsymbol{x}_0\|}, \\
\boldsymbol{\alpha}_{fade} &= 1 - \frac{t}{t_{max}}, \\
\boldsymbol{\alpha} &= \boldsymbol{\alpha}_{density}\ \boldsymbol{\alpha}_{shape}\ \boldsymbol{\alpha}_{curvature}\ \boldsymbol{\alpha}_{fade}.
\end{aligned}
\tag{3.1}
$$



(a) **(b)**

Figure 3.2: Distortion of an equilateral triangle integrated towards the saddle point in the linear vector field $v = (x, -y)^T$. The left figure shows three instances of an integrated triangle. The right figure plots the triangle area and shape quality over integration time. © 2008 IEEE

The opacity value $\boldsymbol{\alpha}$ assigned to each vertex is the product of $\boldsymbol{\alpha}_{density}$, $\boldsymbol{\alpha}_{shape}$, $\boldsymbol{\alpha}_{curvature}$ and $\boldsymbol{\alpha}_{fade}$.

Depending on the area of a mesh cell (triangle $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2$), $\boldsymbol{\alpha}_{density}$ is calculated, simulating the density of smoke. A larger cell and therefore lower density of smoke leads to a more transparent surface. The term k controls the initial density and is linked to the height of the conceptual prism seen in Figure 3.1. This figure also shows the view ray intersecting the prism. $\boldsymbol{\gamma}$ denotes the angle between a mesh cell and the view ray.

$\boldsymbol{\alpha}_{shape}$ is also calculated per mesh cell and indicates the quality of each triangle cell. d_0, d_1, d_2 denote the lengths of the edges of a mesh cell while s controls the influence of $\boldsymbol{\alpha}_{shape}$ on $\boldsymbol{\alpha}$. $\boldsymbol{\alpha}_{shape} = 1$ for equilateral triangles and decreases as the triangle is deformed. The effect of triangle deformation on its shape quality can be seen in Figure 3.2

$\boldsymbol{\alpha}_{curvature}$ computes the curvature of the surface at a vertex $\boldsymbol{x}_0$ using the neighboring vertices x_i in the 1-ring of $\boldsymbol{x}_0$ ($i = 1..valence(x_0)$). If all vertices are on a plane, $\boldsymbol{\alpha}_{curvature}$ is equal to 1. $\boldsymbol{n}_0$ denotes the estimated surface normal at $\boldsymbol{x}_0$ and b controls how strong a large surface curvature influences $\boldsymbol{\alpha}_{curvature}$.

Finally, $\boldsymbol{\alpha}_{fade}$ simulates the dispersion of smoke, by steadily decreasing opacity over time t. The value starts at 1 and approaches 0 as t reaches t_{max}, which is set to the maximum integration time after which a particle is reset to its seeding position.

$\boldsymbol{\alpha}_{density}$ and $\boldsymbol{\alpha}_{shape}$ of a vertex are set to the minimum value of its adjacent triangles, as these values are initially defined per triangle. $\boldsymbol{\alpha}_{curvature}$ and $\boldsymbol{\alpha}_{fade}$ are already defined per vertex and all values have to be clamped to the interval [0,1] individually. As a result, $\boldsymbol{\alpha}$ determines the per-vertex opacity with a value between 0 and 1 so that it can be linearly interpolated across the entire streak surface.

3.2 Simulation Data

The use of CFD simulations to obtain detailed flow data of patient-specific hemodynamics, while not being clinical routine, is becoming more common. Bäumler et al. [9] created high resolution flow simulations of aortic dissections using CTA and 4D Flow MRI data of multiple patients and phantom models, which we are using as input data for our visualizations. The CTA data was used to segment the aorta and extract not only the outer aortic vessel wall and blood volume, but also the dissection flap. The flow field provided by 4D Flow MRI was used to inform the simulation about the flow velocity and its change over the cardiac cycle at multiple locations (aortic root, branching vessels, etc.). Subsequently, blood flow was simulated alongside the deformation of the outer vessel wall and dissection flap using 2-way fluid-structure interaction (FSI) CFD simulations. These were carried out over the course of multiple cardiac cycles to allow the flow to equalize and used 4000 simulation steps per cycle. The last cardiac cycle was recorded at every 50th simulation step, resulting in 80 captured timesteps. We choose to only use 40 out of the 80 exported timesteps, as this provides sufficient temporal resolution for visualization while limiting computational cost and memory requirements.

Simulation results comprise two separate, unstructured tetrahedral meshes. The first mesh contains the volume occupied by the blood inside the aorta, while the second mesh contains the vessel tissue including outer vessel wall and dissection flap. Every vertex on either of the meshes contains multiple hemodynamic and mechanical measurements such as pressure, WSS or displacement in addition to position and velocity.

Real time advection of hundreds of points inside a fluid requires fast access to the flow data. As discussed in Section 2.5 the structure of the data impacts performance heavily. These CFD results are organized in an unstructured grid, meaning vertices are arbitrarily spread throughout 3D space and not aligned to a regular grid. As a result of the irregular structure of the data, integration is very slow and cumbersome. To reduce access times we convert the simulation data to a regular grid data structure. The following section describes the data structure chosen to facilitate quick access to the vector field and the necessary steps to convert the simulation results.

3.3 Data Structure

Vector fields can be organized using different techniques to support specific requirements for both acquisition/simulation and visualization [69]. A common way of organizing vector fields [69] is using a regular grid with uniform spacing along each axis. Aside from simplicity this method offers consistent sampling of space while allowing for fast access times. Given a regular 3D grid, the cell containing an arbitrary location can easily be calculated using the data's bounding box and the voxel size/spacing (see Equation 3.2) without the need of costly searches or complicated data structuring like octrees or kd-trees. Flow data acquired using 4D flow MRI commonly consists of voxels, organizing the data on a regular grid. We resample the tetrahedral mesh data provided by the CFD simulations to a regular rectilinear grid to improve access efficiency. Simultaneously, compatibility with MRI datasets is improved, as their voxel data structure is very similar to the structure

we convert the simulation data to. As a result, both simulated and measured flow data can be accessed and visualized in the same manner.

3.3.1 Data Conversion

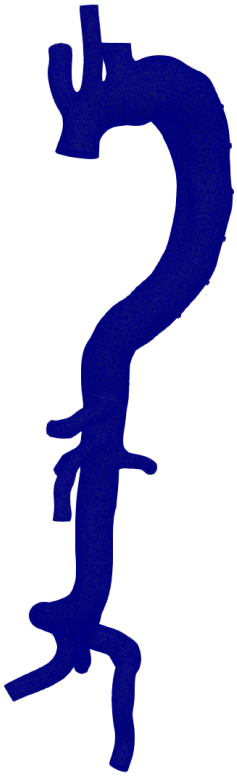
Augmenting an unstructured grid to improve access performance by using an octree for example, does not necessarily require resampling the data. In order to convert an unstructured grid to a structured grid, on the other hand, the data needs to be resampled, creating a new set of vertices located at regular intervals and connected to form a voxel structure instead of a tetrahedral mesh.

Before starting the conversion, the bounding box of the unstructured grid needs to be found and a voxel size needs to be chosen. The size of the voxel determines the distance between vertices and consequently the number of vertices. Smaller voxel size translates to higher resolution through more vertices but also increased memory consumption. Ideally, the voxel size is chosen to be half of the average size of a tetrahedral cell of the unstructured mesh. This allows sufficient sampling of the original data to not lose large amounts of detail. However, a trade-off between quality and memory usage needs to be made, as the size of the voxelized data increases rapidly with reduced voxel size.

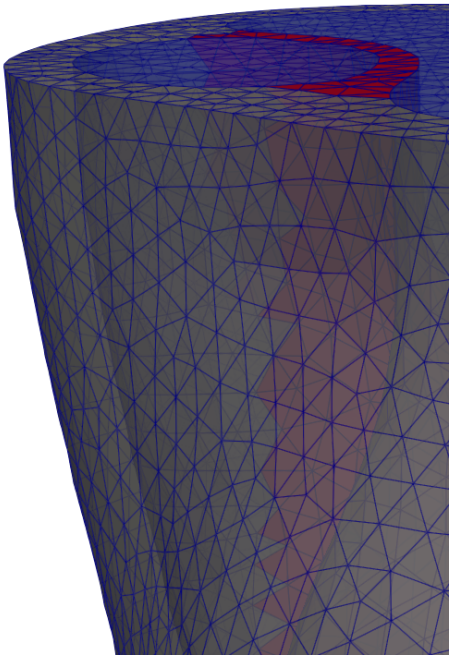
After choosing an appropriate voxel size and determining the bounding box of the data, a regular grid is created by computing its dimensions to fill the bounding box with evenly sized voxels and allocating memory. Subsequently, the mesh data is resampled to fill the regular grid. All values such as velocity vectors, pressure, WSS, and displacement are linearly interpolated, transferring the simulation results into a structured grid and enabling fast access. Importantly, the individual positions of the vertices do not need to be determined and saved but are intrinsically defined by one vertex of the bounding box and the voxel size (see Equation 3.2). The conversion of an unstructured grid into a structured grid including interpolation is carried out using the *Visualization Toolkit (VTK)*. Example of a tetrahedral mesh as well as the resampled voxel structure are shown in Figure 3.3.

$$\begin{aligned} \mathbf{cell} &= \lfloor(\mathbf{loc} - \mathbf{minbounds})/\mathbf{voxelsize}\rfloor \\ \mathbf{dimension} &= \lceil(\mathbf{maxbounds} - \mathbf{minbounds})/\mathbf{voxelsize}\rceil \\ cell_{index} &= cell_x + cell_y * dimension_x + cell_z * dimension_x * dimension_y \end{aligned} \tag{3.2}$$

The tetrahedral mesh provided by the simulation is not static due to deformation of the vessel wall, meaning all vertices of the mesh move over the course of a cardiac cycle. When converting to a structured grid, the same tetrahedron cell might end up in a different voxel of the structured grid depending on its position, which changes over time. As a result, each timestep needs to be converted to a structured grid individually, while maintaining the same bounding box and voxel size. Finally, all converted timesteps are combined by concatenation.

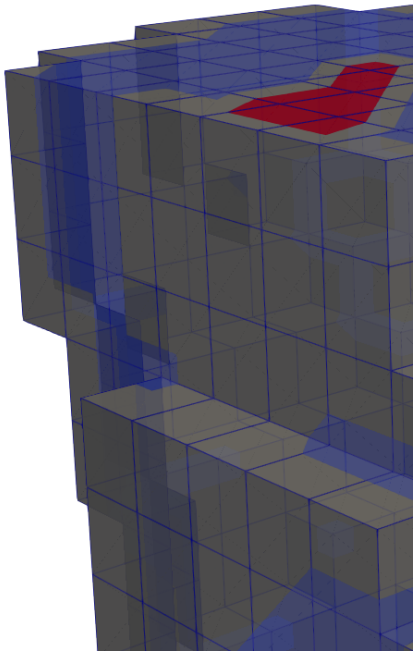


(a) Tetrahedral mesh.



(b) Closeup of tetrahedral mesh.

(c) Uniform grid.



(d) Closeup of uniform grid.

Figure 3.3: Data converted from volumetric tetrahedral mesh, with non-uniform cell, on the top, to a uniform rectilinear grid with voxel size of 1.5 mm on the bottom. Images generated using Paraview [76].

3.3.2 Sparse Grid Representation

To optimize the fidelity of our simulated blood flow datasets, it is imperative to select a small voxel size and a high number of time steps. This ensures the preservation of crucial information and facilitates the generation of intricate flow visualizations. However, when resampling all 40 timesteps over the course of one cardiac cycle with a voxel size of one millimeter, the resulting regular grids often exceed the size of 10 gigabyte (GB), which is commonly manageable by main memory but exceeds the capacity of most consumer-grade GPUs. Without altering the data structure this problem can only be mitigated by reducing either spatial or temporal resolution, or both.

Given our focus on vascular flow visualization, the volume of the bounding box is seldom entirely occupied by important data. Particularly in the context of the aorta's geometry, a significant proportion of voxels in the generated regular grid remain empty. To reduce the memory overhead of our regular grid, we adopt a sparse representation, eliminating the majority of empty voxels. The primary objective of this optimization is to ensure efficient utilization of computational resources while preserving the integrity of our vascular flow visualizations. Maintaining constant access time for non-empty voxels is paramount to ensure real-time performance and seamless interaction with the visualization system.

Our sparse voxel representation is based on a two level hierarchical data structure. At the higher level, the voxel grid is divided into larger cubic regions called chunks. Each chunk represents a section of the voxel grid and can either be filled (containing data) or empty (containing no data). This top-level structure serves as a coarse representation of the overall voxel grid. Each filled chunk in the top-level structure contains a finer subdivision of smaller cubic regions. These smaller regions are traditional voxels, representing the finest level of detail in the grid. Unlike the top-level nodes, which are only either filled or empty, these voxels can individually contain data or be empty.

In the initial structure, data for each timestep is stored as concatenated blocks, meaning that all the data for each timestep is contiguous. Each block represents a specific timestep, containing voxel data for that moment in time. To convert to a sparse voxel representation, each block corresponding to a timestep is further divided into smaller blocks, each sized to fit within a single chunk of the hierarchical structure. These smaller blocks are cubic and contain voxel data for a portion of the overall voxel grid. Only blocks of filled chunks are kept, while empty chunks are omitted. The location in memory of each chunk, filled or empty, is kept in a separate field to enable constant access. When accessing a filled chunk, the necessary location can simply be read, while access to empty chunks can be denied. An illustration of the sparse grid data structure can be seen in Figure 3.4

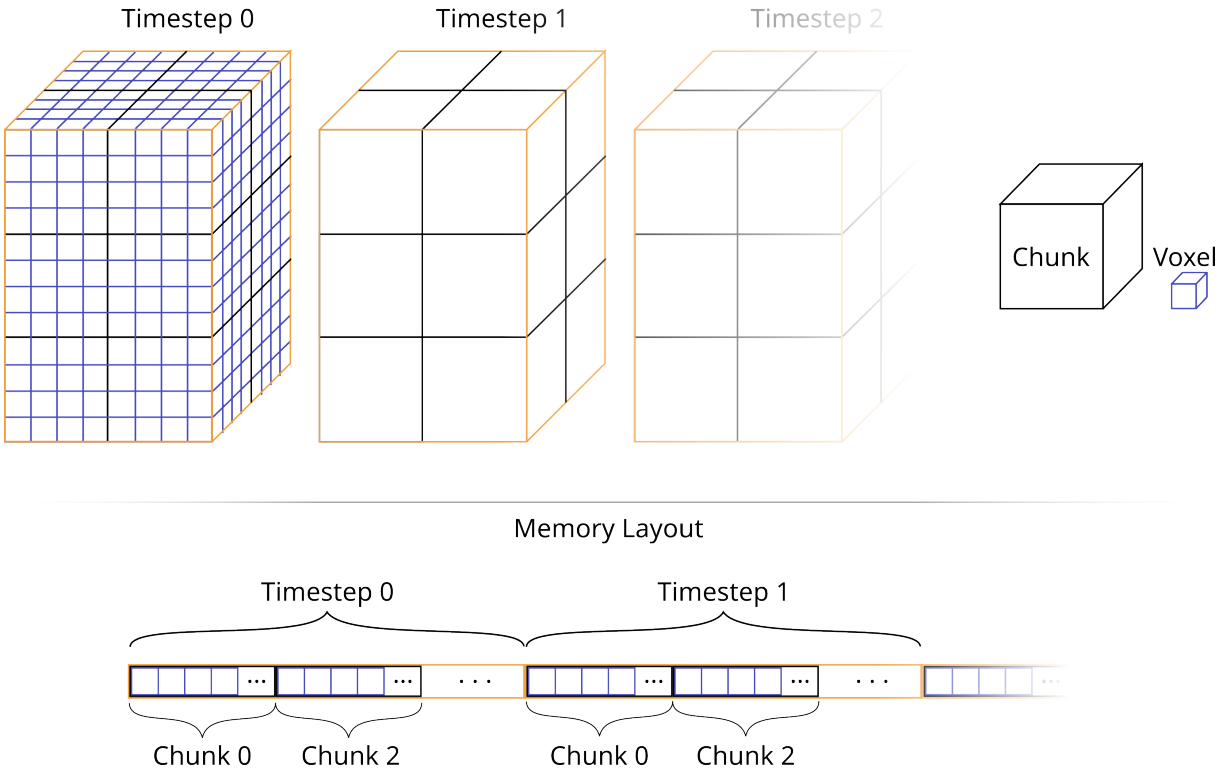
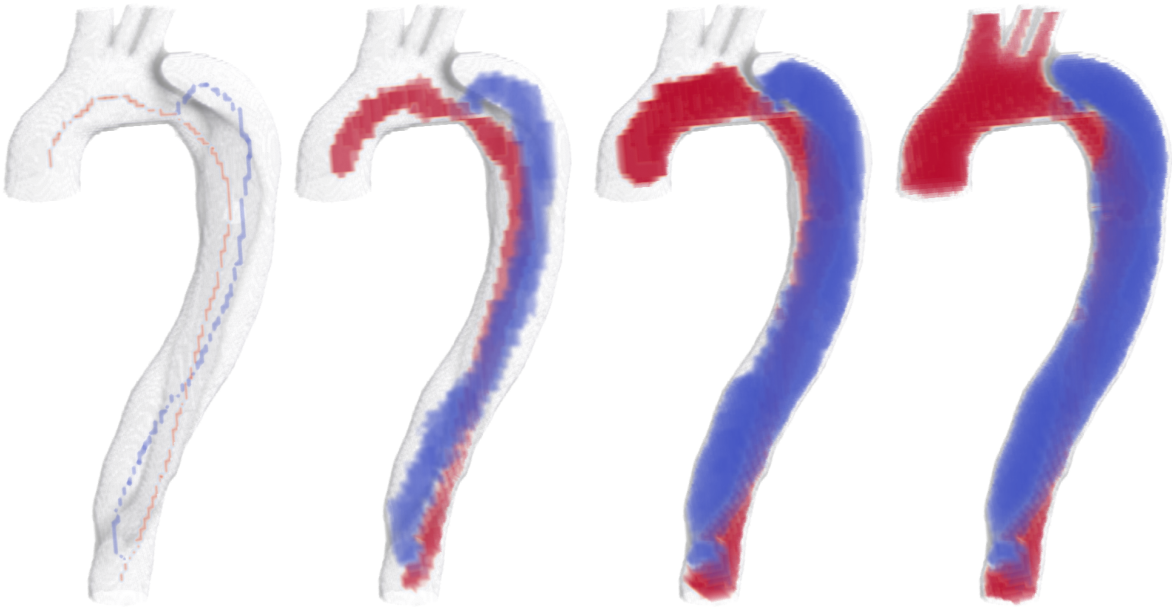


Figure 3.4: Illustration of our sparse grid data structure. On the top the spatial layout of multiple timesteps, chunks and voxels are shown. On the bottom is the corresponding memory layout. Orange represents the extent of a timestep, which is the bounding box surrounding the data. Black represents the extent of a chunk and blue the extent of a voxel. The hierarchical structure can be seen at the top with each timestep encoding the same space at a different point in time. Memory is laid out to enable dense packing of filled chunks. The above example shows chunk 1 being omitted in memory across time steps, demonstrating the dense packing of filled chunks.



(a) 0 cycles **(b)** 2 cycles **(c)** 4 cycles **(d)** 17 cycles

Figure 3.5: Labeling of grid cells according to the local lumen. The initially set labeling at the luminal centerlines (a) is iteratively dilated until all cells are labeled (e). The true lumen is colored red, while the false lumen is colored blue. Images generated using Paraview [76].

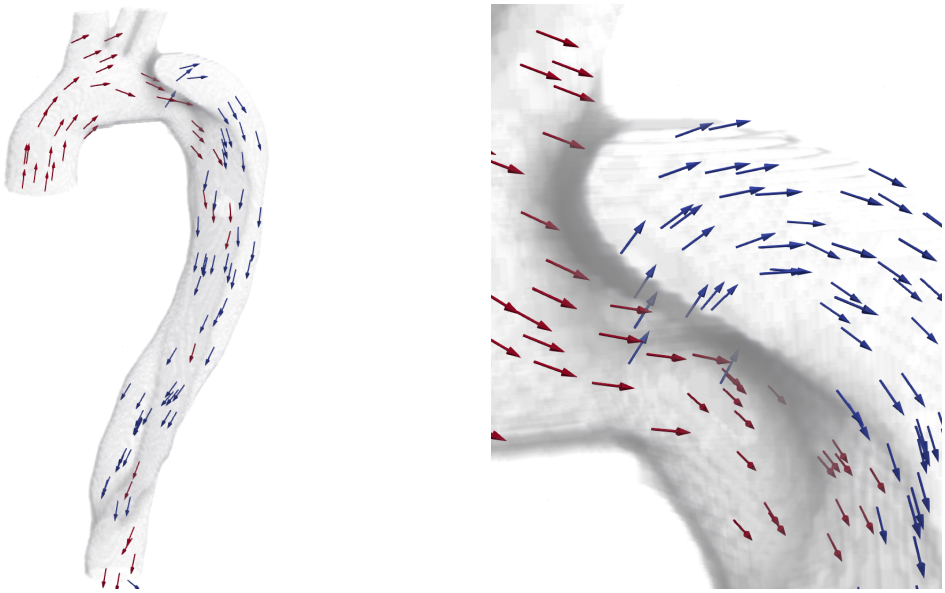
3.3.3 Additional Features

Expanding on the measures provided in the simulation results, we calculate additional hemodynamic measures, which have been selected in collaboration with cardiovascular imaging scientists and cardiovascular radiologists. These additional measures enable a more comprehensive analysis of aortic dissections, and in some cases, facilitate visualization of phenomena that are conventionally imperceptible.

Determining Lumen Origin

Visualizing the origins of individual flow streams in complex flow scenarios can pose significant challenges. In case of aortic dissections, distinguishing between the true lumen and the false lumen can be particularly challenging. To address this, we introduce a color mapping technique applied to *Smoke Surfaces*, aiding in the identification of flow origins. To enable this visualization the dataset is segmented, wherein each grid cell within the fluid domain is assigned a label indicating its association with either the true lumen or the false lumen.

Utilizing the regular grid structure of the converted data, we leverage parallel processing capabilities of the GPU for efficient labeling. Initially, a set of voxel cells is labeled based on the luminal centerline for both lumina. Subsequently, this initial labeling is dilated until all cells are accurately categorized as belonging to either the true or false lumen. This iterative process is illustrated in Figure 3.5.



(a) Full dataset **(b)** closeup

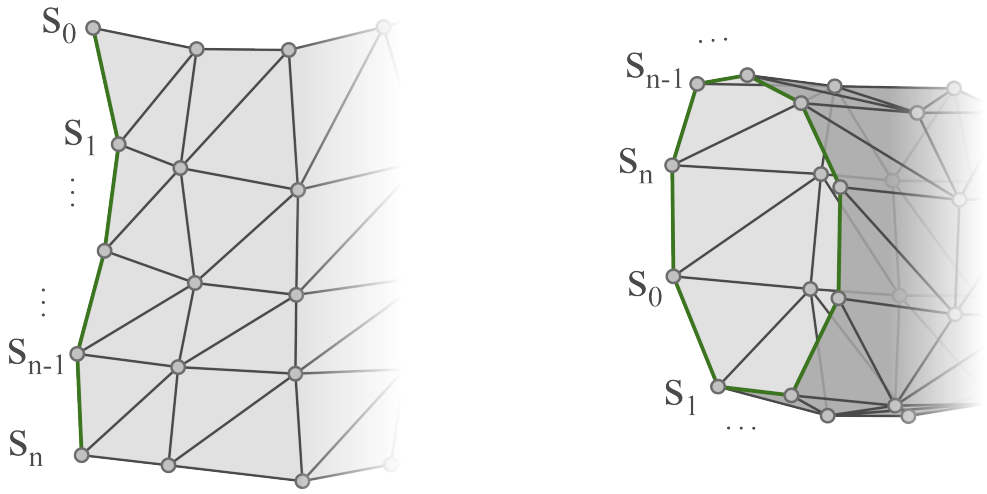
Figure 3.6: Arrow glyphs illustrate the direction of the local antegrade vector applied to grid cells. A subset of all data points, color-coded based on the local lumen is shown. Arrows representing voxel cells of the true lumen are colored red, while those representing voxel cells of the false lumen are displayed in blue. Images generated using Paraview [76].

Identification of Flow Direction

While retrograde flow near the aortic root was shown to indicate aortic valve disease [77], aortic dissections also appear to lead to retrograde flow in specific areas. Retrograde flow occurring at the entry tear inside the false lumen might be related to false lumen pressurization, while retrograde flow inside the whole false lumen might indicate recirculation of blood through the exit tear [78]. Such complex hemodynamic phenomena, which only occur in dissections are typically hard to identify and visualize.

To address this, we introduce a visual mapping technique that emphasizes retrograde flow in any lumen. The relative flow direction (retrograde or antegrade) of a triangle cell of the *Smoke Surface* is determined by comparing the local velocity vector and the overall trajectory of the vessel. To facilitate efficient computation during visualization the vessel's trajectory is established for each voxel cell of the uniform data structure in a preprocessing step.

After labeling each voxel cell based on the local lumen, the local trajectory of the lumen is determined using its lumen-specific centerline. Initially, the closest point on the relevant centerline is identified. Subsequently, a vector tangent to the centerline at this location is computed. Employing a highly parallelized process, each voxel is assigned such a vector representing the local trajectory of the lumen. An example of the resulting vectors is shown in Figure 3.6.



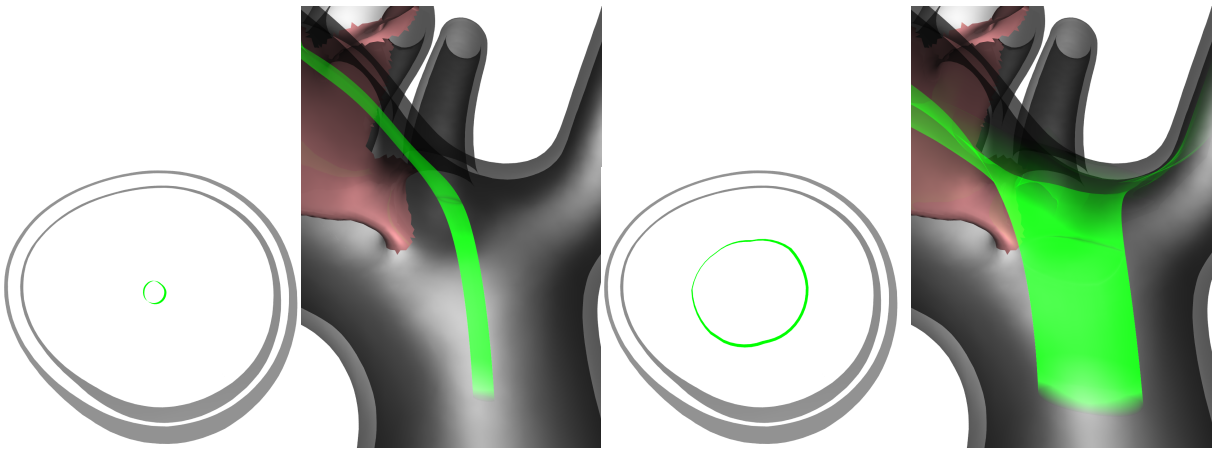
(a) Line structure **(b)** Ring structure

Figure 3.7: Streak surface seeding structures. Figure (a) shows the open, polygonal seeding structure used by von Funck et al., while Figure (b) shows the ring structure we use for seeding inside of vessels.

3.4 Seeding

As shown in the original work of von Funck et al. [1], the location and design of the seeding structure has large impact on the visualization results and can be used to create *Smoke Surfaces*, that approximate different experimental flow visualization techniques. Von Funck et al. used a polygonal seeding structure $(s_0, ..., s_n)$, most commonly chosen to represent a continuous line or curve. As a result, the location and extent of the generated *Smoke Surface* can be chosen arbitrarily, with the surface starting out smooth near the seeding structure and being deformed by the flow, as it is advected. To facilitate the constant connectivity of the *Smoke Surface* and produce a closed surface even when continuously advecting vertices and resetting them to the seed location after reaching the maximum advection time, it is defined as a closed surface of cylindrical topology. This means in the vertex array $(x_{i,j};\ i = 0, ..., m;\ j = 0, ..., n)$ the first column of vertices $(x_{0,0}, ..., x_{0,n})$ is connected to the last column of vertices $(x_{m,0}, ..., x_{m,n})$.

Continuous curve seeding structures were shown to be suitable for the visualization of flow around aerofoils, car bodies or well-known phenomena like the von Kármán vortex street. Vascular structures, particularly aortic dissections present with distinct geometry, which requires a tailored seeding structure. We adapt the curve seeding structures, presented by von Funck et al. [1] by closing it to form a ring. This structure mimics the tubular morphology of vascular structures, creating a closed *Smoke Surface*, which in itself appears tubular. Following the adaption of the seeding structure, the *Smoke Surface* mesh is now defined as a closed surface of toroidal topology. In addition to the connection of first and last column of the mesh, we also connect the first $(x_{0,0}, ..., x_{m,0})$ and last $(x_{0,n}, ..., x_{m,n})$ rows of the mesh. Figure 3.7 depicts the structure of the line as well as ring seeding structures.



(a) Position = 0.1 **(b)** Position = 0.1 **(c)** Position = 0.5 **(d)** Position = 0.5

Figure 3.8: Demonstration of seeding structure generation in the cross-section of an aortic dissection. Each example includes a streak surface extending from the seeding structure, displayed in green, along with the outer vessel wall in gray and the dissection flap in red. In Figures (a) and (b), the relative position of all polygon points between the center and the vessel wall is 0.1. In Figures (c) and (d), the relative position is 0.5.

Due to the unique geometry of aortic dissections and our use of a closed ring the generation of appropriate seeding structures requires special treatment. We generate a seeding structure based on the cross-section of the surrounding lumen, resulting in a roughly circular polygon of evenly spaced points, which are located inside the lumen. The distance between seed points and vessel wall can be adjusted by the user, with all points being placed on the same relative position between vessel wall and cross-section center, as can be seen in Figure 3.8.

3.5 Vulkan Implementation

Vulkan is a modern Graphics API developed by the Khronos Group [79]. It provides a cross-platform standard for high-performance 3D graphics, and a number of built-in extensions. Most importantly for our implementation Vulkan provides an extension, which allows for rendering using GPU accelerated ray tracing. Additionally, non-graphics workloads can be handled by Vulkan through the use of compute shaders, allowing for highly parallelized computations.

3.5.1 Ray Tracing

Ray tracing has emerged as a powerful technique in computer graphics for simulating the behavior of light in virtual environments, enabling highly realistic rendering effects. Because of our goal to create realistic visualizations of smoke we chose ray tracing as a rendering technique. Advanced rendering effects were not yet implemented, as this thesis focuses on the flow visualization technique.

Vulkan has introduced ray tracing support, offering developers unprecedented control and flexibility in crafting immersive visual experiences. In this section, we discuss the rendering pipeline of Vulkan ray tracing and the associated shaders that play a crucial role in the process.

Acceleration Structures

Acceleration structures (ASs) are fundamental data structures employed in ray tracing algorithms to efficiently identify intersections between rays and scene geometry. These structures play a crucial role in optimizing ray traversal, enabling real-time rendering of complex scenes with high visual fidelity. In this section, we explore the ASs commonly used in Vulkan ray tracing.

ASs organize scene geometry in a hierarchical manner, facilitating rapid ray-object intersection tests. These structures serve as spatial indices, partitioning the scene into manageable regions and enabling efficient culling of geometry that lies outside the view frustum or is occluded by other objects. By leveraging ASs, ray tracing algorithms can dramatically reduce the number of geometric primitives that need to be tested for intersection, leading to significant performance improvements.

Bounding volume hierarchies (BVHs) represent one of the most commonly used ASs in Vulkan ray tracing. BVH organizes scene geometry into a binary tree structure, where each node corresponds to a bounding volume enclosing a subset of primitives. The hierarchy is constructed recursively by partitioning bounding volumes along spatial splits, such as axis-aligned bounding boxes (AABBs) or spatial median planes. BVH traversal involves descending the tree from the root node to the leaf nodes, testing rays against bounding volumes and efficiently pruning subtrees that do not intersect with the ray.

During the setup of a scene for ray tracing, Vulkan distinguishes between two types of ASs. These two types, namely top-level acceleration structure (TLAS) and bottom-level acceleration structure (BLAS), represent different levels in the hierarchy of the scene and need to be build in a specific order.

The BLAS represents the lowest level of the acceleration hierarchy and is responsible for encapsulating individual or grouped geometric primitives. In Vulkan ray tracing, a BLAS is constructed from vertex and index buffers containing geometry data, such as vertices, normals, and texture coordinates. The construction of a BLAS involves partitioning the geometry into primitive groups, such as triangles or AABBs, and encoding them into a compact representation optimized for ray traversal.

At the opposite end of the acceleration hierarchy lies the TLAS, which provides a high-level representation of scene geometry for ray traversal. The TLAS encapsulates instances of BLAS, allowing multiple instances of geometry to be efficiently organized and traversed during ray tracing. Each instance in the TLAS specifies a transformation matrix, enabling dynamic positioning, scaling, and orientation of geometry within the scene. TLAS serves as the entry point for ray traversal, initiating intersection tests against scene geometry and facilitating efficient occlusion culling and visibility determination.

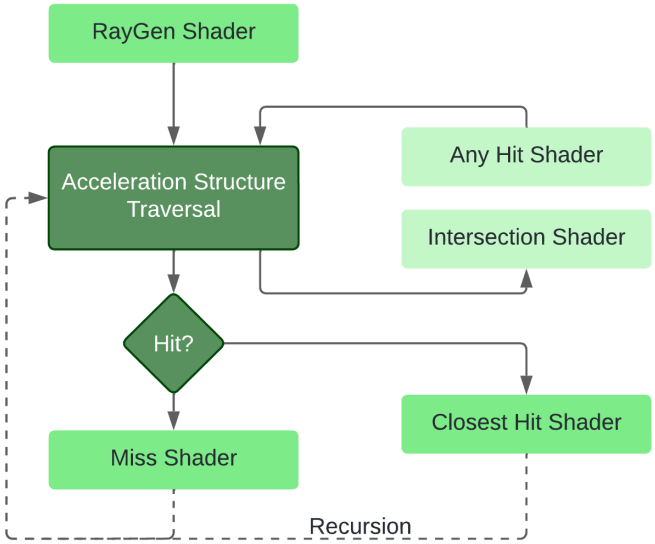


Figure 3.9: Schematic of the Vulkan ray tracing pipeline including ray generation, hit, and miss shaders as well as the optional any hit and intersection shaders.

Ray Tracing Pipeline and Shaders

The Vulkan ray tracing pipeline encompasses a series of stages designed to efficiently trace rays through a scene and compute the resulting color values. At its core, the pipeline revolves around the concept of ray generation, intersection testing, and shading. Each stage contributes to the overall rendering process, from initializing rays to evaluating material properties and computing final pixel colors.

The ray generation stage serves as the entry point for the ray tracing pipeline. Here, developers define a shader responsible for generating primary rays that originate from the virtual camera and traverse the scene. This shader typically computes the initial ray directions based on the pixel coordinates and camera parameters.

Following ray generation, the intersection stage comes into play, where rays are tested for intersections with scene geometry. Utilizing ASs, Vulkan efficiently traverses the scene to determine ray-object intersections. Upon detecting an intersection, the any-hit shader is invoked, which allows culling geometry or prematurely terminating the ray. After determining all intersections along a single ray, the closest-hit shader is invoked to compute the color contribution based on surface properties such as color, reflectance and transparency. If no intersection is detected for a ray, the miss shader is invoked. This shader is essentially used to render the scenes background by providing a constant color, sampling a texture, etc. A schematic of the ray tracing pipeline can be seen in Figure 3.9

One of the key advantages of Vulkan ray tracing is its flexibility and performance. By providing low-level access to hardware resources and execution control, developers can optimize shaders and pipeline configurations to achieve desired rendering effects while leveraging the full capabilities of modern GPUs.

3.5.2 Smoke Surface Computation

Vulkan, as a low-level graphics API, offers more than just rendering capabilities. It provides a versatile framework for general-purpose parallel computing through compute shaders. Compute shaders are specialized programs executed on the GPU that perform parallel computations without the traditional graphics pipeline. Unlike graphics shaders, which are primarily concerned with rendering geometry and pixels, compute shaders are tailored for general-purpose computations, offering a flexible and efficient means of harnessing the computational power of modern GPUs. We use compute shaders to perform highly parallelized computations such as advection and mesh updates without the restrictive structure of a rendering pipeline.

Vulkan compute shaders operate in the compute pipeline stage, which is distinct from the graphics pipeline. They are dispatched by the CPU and executed in parallel by multiple shader invocations, or workgroups, on the GPU. Each workgroup comprises multiple threads, with the exact number determined by the developer during shader dispatch. The size of these workgroups impacts performance and needs to be specified at compile time. We optimized the workgroup size for streak surface sizes commonly used during testing. Compute shaders have access to specialized memory regions, such as shared memory and global memory, enabling efficient data sharing and communication among threads. This is crucial for efficient computations in our real-time application, especially because of a high number of memory accesses.

One of the notable features of Vulkan compute shaders is their integration with the graphics rendering pipeline. Compute shaders can be used to perform pre-processing tasks, post-processing effects, and off-screen rendering operations, augmenting the capabilities of traditional graphics shaders. We make use of this integration as live advection of the streak surfaces needs to be synchronized with their rendering.

The workload of streak surface advection lends itself well to parallelization, as each vertex of the surface mesh can be processed individually. The generation and animation of *Smoke Surfaces* not only relies on advection of the surfaces vertices, but also on additional parameters, which need to be reevaluated for every frame. In order to perform the necessary calculations quick enough to enable interactive frame rates, they are also carried out in a compute shader. This is a multistep process, which uses 3 distinct shaders, to advect the vertices, compute per-face parameters as well as normals and compute per-vertex parameters and normals. Finally, the individually computed parameters are combined upon ray intersection using the closest-hit shader.

Advection Shader

The advection shader is responsible for the sampling of the vector field, integration and interpolation to update the positions of the *Smoke Surface* vertices. Along with the common inputs, namely seeding positions and vector field, the advection shader receives additional parameters, such as voxel grid dimension and origin, streak surface size, and elapsed time. To be able to access the correct cell of the vector data, the dimension of the grid, the origin / minimum bounds and the size of a cell need to be known. Using these information and the calculations from Equation 3.2 the cell containing a given 3D

location can be identified. Important for the proper advection of streak surface vertices is the size of the surface mesh, meaning the number of rows and columns, as well as the time difference between columns (t_{col}). This difference is used to delay the advection of vertices at the beginning of the visualization and determine, when they need to be reset to their seeding position. Lastly, since the shader only keeps track of how far a vertex is in its cycle, before it needs to be reset and not of the overall duration of the visualization, the elapsed time is provided, which is also used to delay advection and reset vertices to their seeding position, as can be seen in Equation 3.3.

As described in Section 3.4 a vertex ($x_{i,j}$) starts at its seeding position (s_j) and is then advected. To space out the vertices of the mesh, vertices with matching i are released simultaneously, only after the time since start (t_{start}) exceeds their release time (t_{rel}). Subsequently, they are advected until the time since release ($t_{start} - t_{rel}$) exceeds t_{max}. This is determined using the $fade$, which indicates, how much longer a vertex is advected before being reset. Each vertex starts with $fade = 1$ when it is released and is reset as soon as $fade = 0$. As t_{start} continuously increases, even over multiple cardiac cycles, a modulo operation of 1 is applied to restrict $fade$ to the interval (0,1].

$$\begin{aligned} t_{max} &= m * t_{col} \\ t_{rel} &= i * t_{col} \\ fade &= 1 - \left(\frac{t_{start} - t_{rel}}{t_{max}} \ mod \ 1 \right) \end{aligned} \tag{3.3}$$

Mesh Face Shader

The face shader operates on a thread per triangle mesh cell (face) of the *Smoke Surface*. Multiple values are then computed, which need to be determined per face before being assigned to vertices by the subsequent shader. First, the face normal, essential for shading and the calculation of $\alpha_{density}$, is determined. The area of the face, also needed for $\alpha_{density}$, is calculated and passed on. Finally, the opacity value α_{shape} is determined within the face shader (see Equation 3.1). This concludes the necessary computations for the original *Smoke Surface* visualization. Additionally, we compute the longest edge length of the face for an augmented computation of α_{shape}.

Mesh Vertex Shader

Analogous to the face shader, the mesh vertex shader operates on a thread per mesh vertex. Here, the per-face values are converted to per-vertex values using the following methods. As described in Section 3.1, α_{shape} and the triangle area values of a vertex are set to the minimum value of its adjacent mesh cells. The vertex normal is computed by normalizing the sum of the normals of all adjacent mesh cells. Lastly, $\alpha_{curvature}$ is computed according to Equation 3.1.

Closest Hit Shader

The closest hit shader is called once the closest ray intersection is determined. The indices of the intersected object and primitive, provided through shader variables, are used to select the applied shading style and surface properties. If the ray hits a part of the *Smoke Surface*, the final value of $\alpha_{density}$ is calculated using the triangle's area, computed in the Mesh Face Shader and the incident angle of the ray (see Equation 3.1). Subsequently, all parameters are combined. Lastly, the final opacity value and a color, which can be determined through different color mappings, are passed back to the ray generation shader.

3.5.3 Synchronization

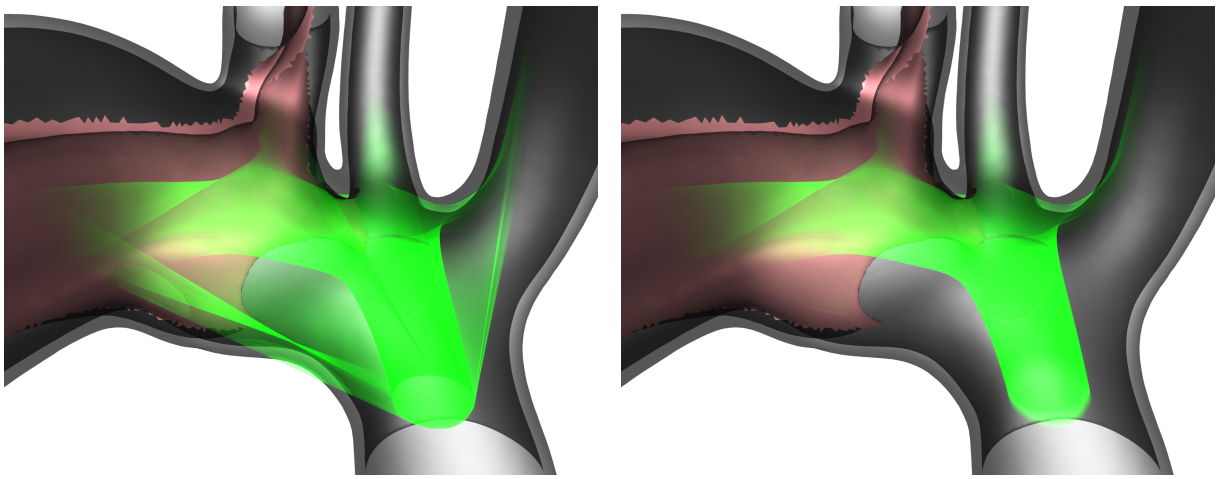
Synchronization is a crucial part of developing with the Vulkan API. Proper synchronization of shaders becomes especially important when working with multiple consecutive shaders to ensure correct results.

In our case the Advection, Mesh Face and Mesh Vertex shaders are executed sequentially and cannot be easily combined. As mentioned before, the advection shader computes the new positions of all mesh vertices. Subsequently, the Mesh Face shader computes per-face parameters. These shaders cannot be combined, because all vertices of a face need to be updated by the advection shader before the face parameters can be computed. Similarly, the Mesh Face and Mesh Vertex shaders follow the same principle, where the parameters of all faces adjacent to a vertex need to be set before the per-vertex parameters can be computed. Consequently, each shader execution must be completed entirely before the next one can commence. Unfortunately, this sequential processing hampers performance, as the computations cannot be executed in parallel.

Furthermore, the entire streak surface advection process is intertwined with rendering. Since Vulkan ray tracing is accelerated using hierarchical ASs any mesh alterations necessitate updating the BLAS to ensure both optimal performance and accurate rendering. Given that during the animation of a streak surface, its vertices regularly undergo significant movement, updating the positions of the mesh's vertices mandates a subsequent update of the corresponding AS. Only after updating the AS can the rendering of a new frame commence.

While operations on ASs can not be performed during rendering, streak surface advection and mesh updates could possibly be decoupled, potentially increasing performance. However, implementing such decoupling requires careful consideration of data transfer and synchronization, which may introduce significant overhead. Ultimately, this optimization may not yield the desired performance benefits, as the overhead incurred could potentially offset any gains achieved through decoupling. Further testing is required to ascertain the effectiveness of this approach.

We implement the synchronization between compute shaders using Vulkan's Memory Barriers, which defer the execution of processes until a specific type of memory access has been completed. To ensure the proper timing of AS updates, so-called Fences are utilized. These Fences are accessible by the CPU and serve as signals indicating when the execution of a command has finished. AS rebuild operations issued by the CPU must



(a) (b)

Figure 3.10: Demonstration of our adapted fade opacity term. Both figures show a streak surfaces seeded in the ascending aorta extending into multiple branching vessels and towards the descending aorta. Figure (a) uses the original fade opacity term without a lead-in fade. As a result, artifacts connecting the start and end of the streak surface can be seen. Figure (b) uses our adapted computation of $\boldsymbol{\alpha}_{fade}$ introducing a gradient at the start of the streak surface, effectively hiding the artifacts.

wait for all compute shaders to finish. Additionally, Pipeline Barriers are employed to defer rendering until all AS rebuilds have been completed.

3.6 Addressing Artifacts

The techniques involved in creating the *Smoke Surfaces* of von Funck et al. [1] were developed for and tested on datasets, that are focused on the aerodynamics around different objects such as cars or aerofoils. The distinct geometry of vascular structures, presents new circumstances and challenges for the use of this technique. As a result of the unique flow inside of vessels, aortic dissections in particular, we adapt multiple aspects of the original *Smoke Surfaces*, including the mesh structure and opacity calculations. Our adaption of the opacity calculations will be described in the following sections.

3.6.1 Fade Opacity Term

During testing of the original opacity mapping, a high number of artifacts could be seen. Some of these appeared as faint surfaces extending from the seed location to the end of the streak surface, as seen in Figure 3.10a. The reason for these artifacts is the application of the fade opacity term. The value of $\boldsymbol{\alpha}_{fade}$ is determined by the time a particle has spent in the flow since it was released at its seed point. As a particle reaches the maximum time, it is reset to the seed point, waiting to be released again. A subset of cells of the streak surface mesh therefore span between the very end and the very beginning of the streak surface, while at least one of the vertices receives an $\boldsymbol{\alpha}_{fade}$ of 1. Two solutions

to this issue were considered. First, to only reset a particle's time spent in flow as it is released. Second, to apply a small fade-in on top of the original fade-out effect of $\boldsymbol{\alpha}_{fade}$. We decided to implemend the second solution, as it does not change the behavior of a particles time spent in flow t, but rather the opacity calculated from it. This solution effectively removes the artifacts, as seen in Figure 3.10b. Adjusting the length of the fade-in, also allows smoothing of the otherwise harsh starting edge of the streak surface. Additionally, we add a factor f controlling the influence of fade on the overall opacity similar to the factors k, s and b. Our calculation of $\boldsymbol{\alpha}_{fade}$ can be seen in Equation 3.4, with l representing the length of the fade-in relative to the length of the entire streak surface. The value of l was set to 0.05 for all renderings of *Smoke Surfaces* in this work.

$$\boldsymbol{\alpha}_{fade} = \begin{cases} \left(t \, \dfrac{t_{max} - l}{t_{max}} \right)^f, & \text{if } \dfrac{t}{t_{max}} < l. \\ \left(1 - \dfrac{t}{t_{max}} \right)^f, & \text{otherwise.} \end{cases} \tag{3.4}$$

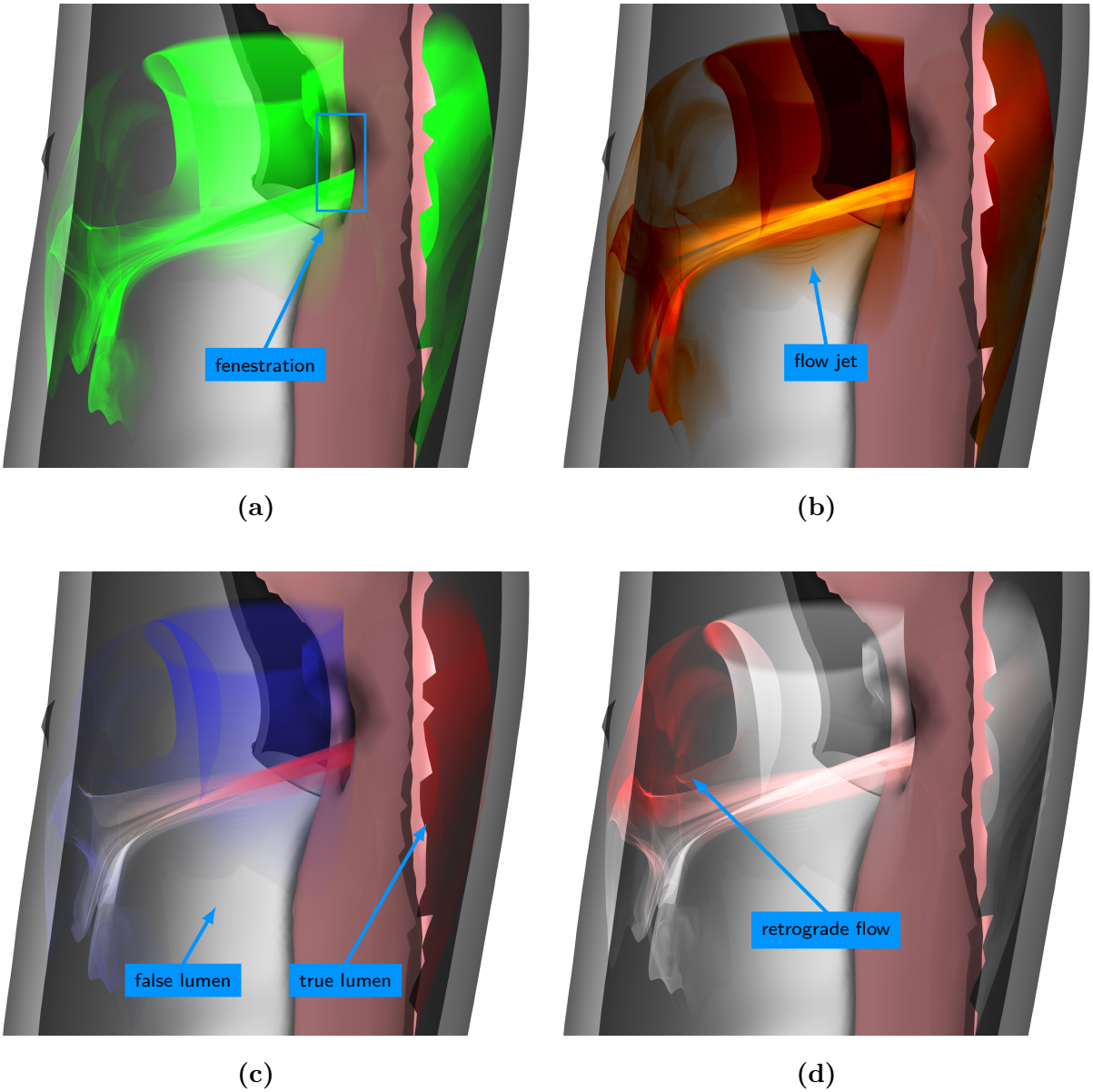
3.6.2 Shape Opacity Term

After removing the artifacts produced by the application of the fade term, some artifacts still remained. Those mostly occured due to diverging flow, in particular, when the flow splits into true and false lumina. Due to the constant connectivity of the streak surface mesh, vertices advected into the false lumen are still connected to other vertices, which stayed in the true lumen inspite of the dissection flap physically separating both flow channels. The original opacity terms do not succeed in completely hiding the mesh cells connecting vertices in separate lumina (see Figure 4.4a and Figure 4.4c). Most of those connecting cells are very thin and long, because they get stretched across the dissection flap. Therefore, as mentioned in Subsection 3.5.2, we introduce a per cell measurement of the longest edge length denoted e. The original shape opacity term is raised to the power of e, to drastically reduce the opacity of long mesh cells. A lower limit of 1 must be applied to e however, to not also increase the opacity of cells with $e < 1$. Our computation of $\boldsymbol{\alpha}_{shape}$ can be seen in Equation 3.5.

$$\boldsymbol{\alpha}_{shape} = \left(\left(\frac{4 \; area\left(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2\right)}{\sqrt{3} \; max\{d_0 d_1, d_1 d_2, d_2 d_0\}} \right)^{max(1,e)} \right)^s \tag{3.5}$$

3.7 Color-Mapping

Color mapping plays a pivotal role in enhancing the interpretability and information density of flow visualizations. By effectively assigning colors to represent various data attributes or parameters, complex information can be conveyed in a visually intuitive manner. In this section, we discuss the application of color mappings to *Smoke Surfaces*, aiming to augment the information conveyed and improve the overall comprehension of the presented data.



(a) **(b)**

(c) **(d)**

Figure 3.11: Demonstration of various color-mappings applied to the same streak surface. (a) presents the surface in a solid high-contrast color, facilitating the easy differentiation between vessel and streak surface geometry. (b) shows a heated-body color scale encoding flow velocity magnitude applied to the streak surface. The luminal origin of flow is encoded through a diverging red-blue color scale in Figure (c). Lastly, (d) showcases our visual mapping of flow direction using a monochrome color scale, highlighting retrograde flow in red.

3.7.1 Flow Velocity

One of the most common, but also most important measures encoded through color in flow visualizations is the velocity magnitude [80]. When assessing flow in aortic dissections a clear distinction between fast and slow flow is critical for identifying different phenomena such as flow jets or areas of stagnant flow. For this reason, we map the magnitude of the flow velocity vector to a heated-body color scale, ranging from black (minimum) to bright yellow (maximum). An example of this color-mapping is shown in Figure 3.11b.

3.7.2 Lumen-of-Origin

The accurate depiction of the originating lumen of branching vessels is of high importance in clinical evaluations of aortic dissection hemodynamics and in the strategic planning of stent graft interventions, as indicated by our colaborating clinicians. To this end, we identify the originating lumen of streak surfaces based on the lumen classification assigned to each voxel (recall Subsection 3.3.3). Each vertex of the streak surface mesh starts at a seed point whose lumen label is associated with its enclosing lumen. During advection, the lumen label $\mathcal{L}_t(v)$ of vertex v at time t adapts to the visited voxel element $\mathcal{L}_t(e)$ as follows:

$$\mathcal{L}_t(v) = (1 - \delta)\mathcal{L}_{t-1}(v) + \delta\mathcal{L}_t(e), \tag{3.6}$$

with $\delta = 0.05$ controlling the rate of adaptation. This value was determined through testing and can be adjusted by the user. This allows the streak surface to slowly change flow channel association when transitioning between channels. Figure 3.11c shows how this information is displayed using a diverging color scale, from red (true lumen) over white (transition) to blue (false lumen). This reveals the immediate interaction of the flow at a fenestration, while the contrast between sections of a single surface or between multiple streak surfaces is reduced when the flow unifies downstream.

3.7.3 Flow Direction

Our colaborating clinicians emphasize the importance of observing the development of the already weakened false lumen. To visualize flow direction, we determine the difference between a streak surfaces direction of movement and the designated luminal flow. For a given position $\mathbf{p}$ on a streak surface with velocity vector $\mathbf{v}$ we calculated the corresponding luminal flow direction as the tangent vector $\mathbf{t}$ at the luminal centerline point closest to $\mathbf{p}$ (recall Subsection 3.3.3). The angle $\gamma = \angle(\mathbf{v}, \mathbf{t})$ then allows us to distinguish between antegrade ($\gamma \leq 90°$) and retrograde ($\gamma > 90°$) flow. To represent a continuous change of direction, we modulate the colors of antegrade and retrograde streak surfaces depending on γ. When using a monochrome color scale from white to red, as show in Figure 3.11d, the predominant, normal, antegrade flow is shown in white, while the retrograde flow is highlighted.

4 Results and Discussion

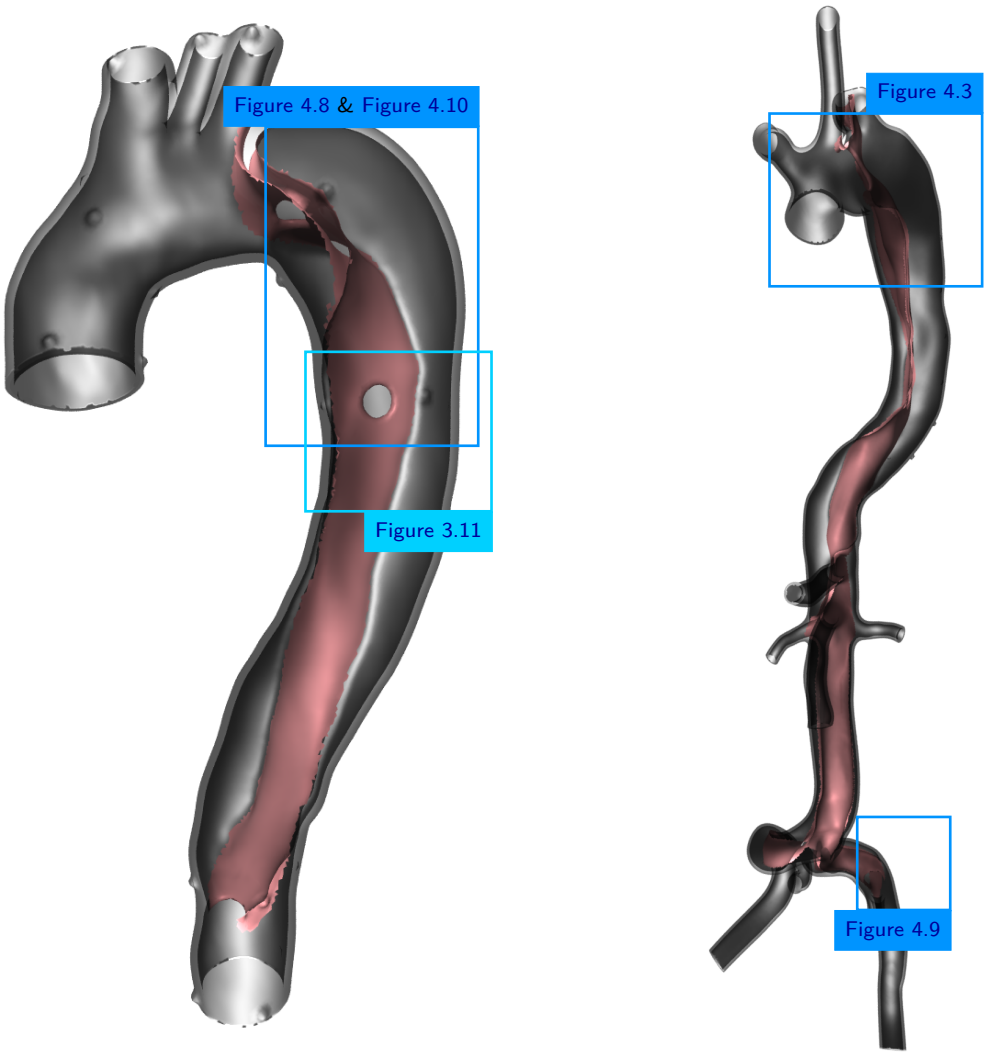
This chapter presents the results of our flow visualization for aortic dissections. The primary objective was to create an effective approximation of smoke being advected by the flow, enabling a detailed examination of the complex flow dynamics inherent in such a pathological condition. Adaptations were made to the original opacity terms to mitigate artifacts and enhance the visual fidelity of the *Smoke Surfaces* within the vascular environment.

To enhance the interpretability of the visualizations, color mappings were employed to encode flow attributes, such as velocity magnitude, flow direction, and the lumen-of-origin. This encoding scheme not only facilitated a comprehensive understanding of the flow behavior but also enabled the differentiation between flow patterns originating from different lumina.

The results are structured to provide a detailed demonstration of individual techniques involved in creating the final visualization, as well as exploration of the flow characteristics observed within the aortic dissections. We demonstrate our visualization's ability to show intricate flow phenomena, including flow separations and flow interactions between both lumina. Our results are showcased through two distinct datasets. For an overview of both datasets, refer to Figure 4.1.

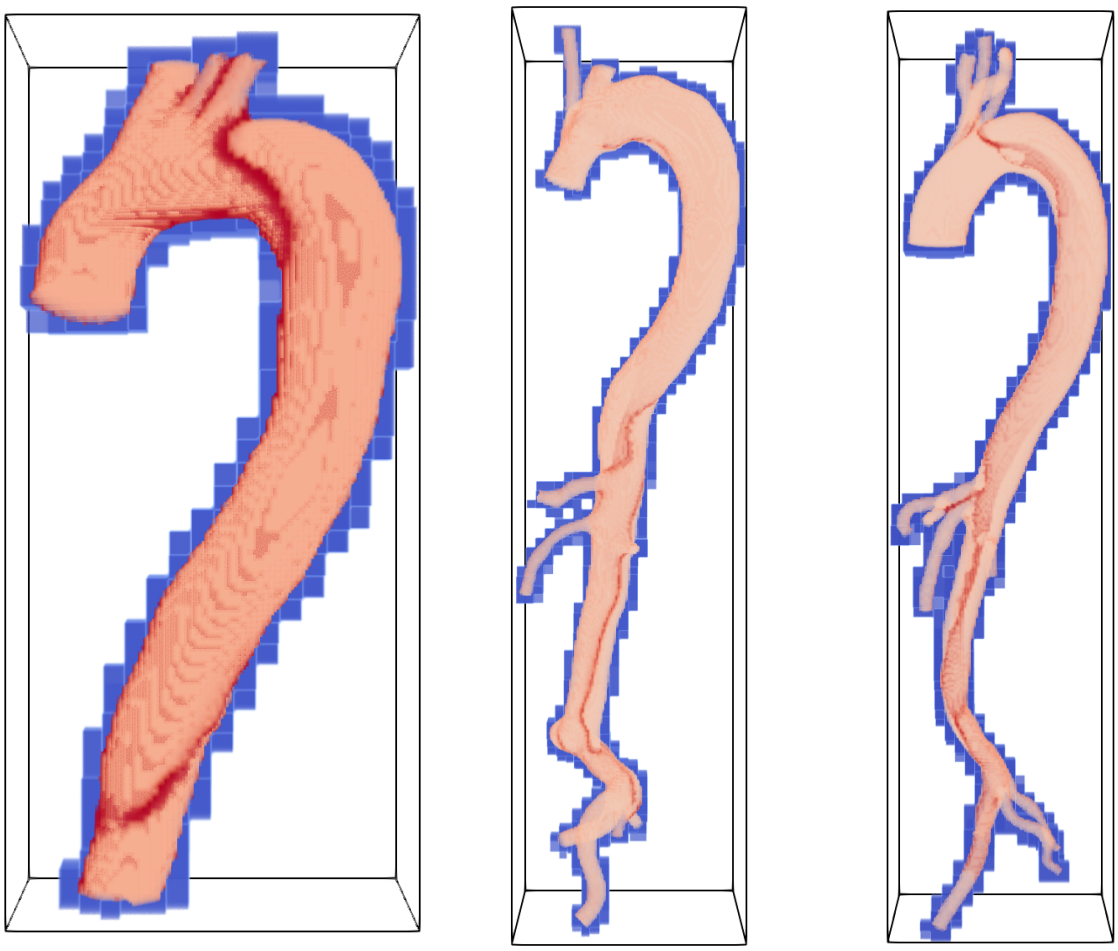
4.1 Sparse Voxel Grid

The sparse voxel grid data structure was tested on the two datasets shown in Figure 4.1 as well as a third dataset, which is not used to demonstrate our visualization. All three datasets were initially read from the simulation results in tetrahedral mesh format. Subsequently, each of the 40 timesteps underwent individuall conversion to a structured voxel grid, with a voxel size of 1.5 mm. The resulting dataset sizes are as follows: 1.89 GB for dataset 1, 13.45 GB for dataset 2, and 4.04 GB for dataset 3. Following this, the convertion to the sparse voxel representation was applied, using a chunk size of 4, wherein each chunk contains 4^3 voxels. Consequently, this operation reduced the size of dataset 1 to 0.43 GB (22.8%), dataset 2 to 1.18 GB (8.8%), and dataset 3 to 0.82 GB (20.4%). As illustrated in Figure 4.2, the original size of all three datasets is depicted by the bounding box, while the reduced size is visually represented by the volume rendered in blue.



(a) **(b)**

Figure 4.1: Overview of datasets 1 (a) and 2 (b). (a) shows an aortic dissection starting in the aortic arch and ending in the lower thoracic aorta. A single entry tear and renetry tear as well as a fenetration in the upper thoracic aorta can be seen. (b) shows an aortic dissection starting in the aortic arch and extending into both femoral arteries. The dissection comprises a single entry tear and two reentry tears, one in each femoral artery.

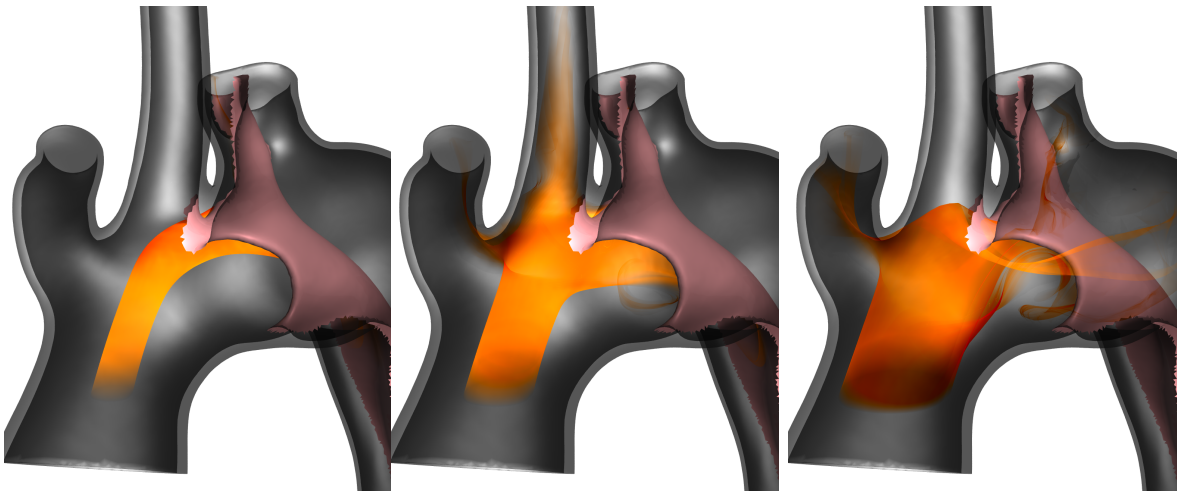


(a) (b) (c)

Figure 4.2: Visualization of sparse voxel grid representation. The black outline shows the bounding box of a dataset. Originally all voxels in the bounding box were included in the voxel grid structure. Rendered in blue is the sparse representation only comprising chunks, that contain data. Finally, rendered in red is the vessel itself. Images generated using Paraview [76].

4.2 Morphology

Section 2.3 explains the essentials of focus and context visualizations, which we applied to our visualization techniques. The surfaces of the aortic vessel wall rendered using the techniques of Ostendorf et al. [29] serve as context to enhance the interpretability of the flow visualization. An isolated depiction of the outer vessel wall and the dissection flap can be seen in Figure 4.1. Through transparent rendering of the outer vessel wall, the underlying dissection flap is revealed, while simultaneously maintaining visibility of the vessel's course, shape, and extent. By rendering the dissection flap opaque, we mitigate ambiguity arising from an excessive number of transparent layers, thereby supporting clear assessment of vessel morphology, as well as spatial perception and navigation.



(a) (b) (c)

Figure 4.3: Renderings of streak surfaces resulting from seeding structures of different sizes. All three figures show a streak surface seeded in the ascending aorta of dataset 2. Figure (a) uses a seeding structure size of 0.3. The resulting streak surface shows the flow in the center of the vessel does not supply any branching vessels or the false lumen. Figure (b) shows the streak surface generated from a seeding size of 0.5. It shows flow supplying mainly the left carotid artery and the true lumen of the aorta. Figure (c) places the seeding structure closer to the vessel wall with a size of 0.8. The streak surface shows flow supplying the brachiocephalic artery as well as true and false lumen of the aorta.

4.3 Hemodynamics

We utilize streak surfaces as a visualization technique to represent the hemodynamics involved in aortic dissection, employing CFD simulations conducted with a 2-way FSI approach. Moreover, to enhance the visual representation, we implement opacity modulations, heavily influenced by the methods proposed by von Funck et al. [1]. These modulations are crucial in creating a smoke-like appearance, which aids in the understanding and interpretation of the complex flow dynamics of aortic dissections.

4.3.1 Seeding

The streak surfaces used for flow visualization are seeded using a closed polygonal seeding structure. During the exploration of an aortic dissection dataset, the user can choose size and location of the seeding structure. A set of example seedings is demonstrated in Figure 4.3.

Using this technique, the seeding of streak surfaces can be adjusted to the needs of the user. Seeding at the aortic root may reveal supply of branching vessels and possible vortices. Figure 4.3 demonstrates the influence of the seeding structure size on the distribution of the streak surface. Conversely, seeding close before a fenestration can reveal the magnitude of cross-flow between lumina and possible flow jets, as seen in Figure 3.11.

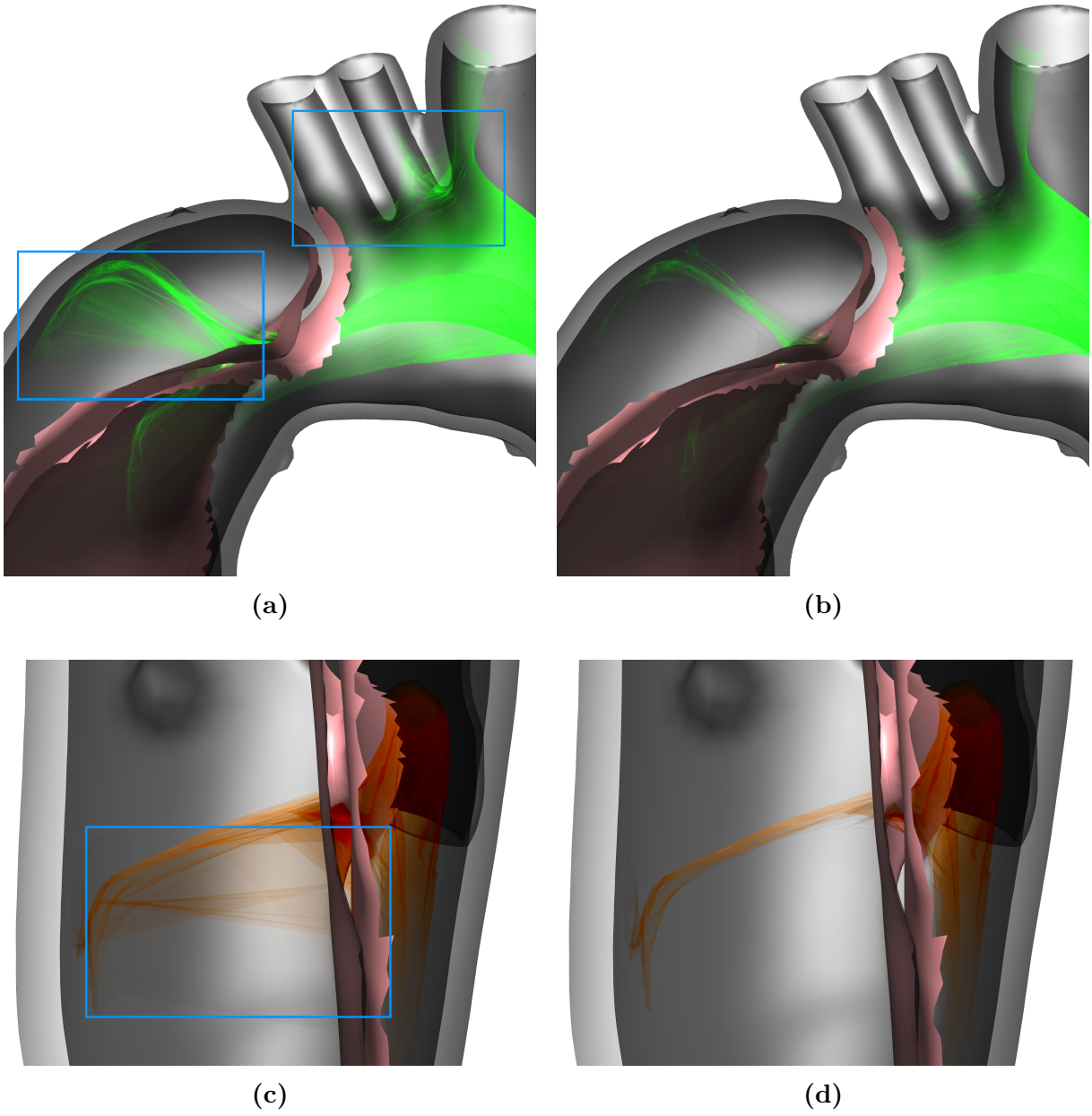


Figure 4.4: Demonstration of our improved shape opacity term. Figures (a) and (b) show streak surfaces seeded in the ascending aorta extending into the branching vessel at the aortic arch as well as the true and false lumina in the descending aorta. Figure (a) uses the original computation of the $\boldsymbol{\alpha}_{shape}$, while figure (b) uses the updated equation. Figures (c) and (d) show streak surfaces seeded in the descending aorta close before a fenestration with the streak surfaces passing through the fenestration. Analogous to the first example, (c) uses the original and (d) the updated equation for $\boldsymbol{\alpha}_{shape}$.

4.3.2 Artifact Reduction

The modifications of the *Smoke Surface* opacity terms described in Section 3.6 were developed to improve the use of *Smoke Surfaces* for the visualization of hemodynamics in aortic dissections. Using the original opacity terms, artifacts can become visible when the flow is separated by thin walls. While this may occur in non-medical applications, this phenomenon is especially common in aortic dissections as the flow splits between two flow channels only being separated by the thin dissection flap. Two examples of these artifacts are shown in Figure 4.4. Thin, stretched triangle cells of the streak surface mesh show up as straight lines, that connect parts of the streak surface, which should not be connected in both Figure 4.4a and Figure 4.4c. The first figure shows, that this does not only occur at the entry tear, but also, where multiple vessels branch off the aorta (see the highlighted areas). Figure 4.4c shows artifacts occuring where a streak surface enters the false lumen through a fenestration. After applying our changes to the computation of the shape opacity term, the majority of those artifacts were removed, while not interfering with the remaining triangles of the streak surface mesh. This is demonstrated in Figure 4.4b and Figure 4.4d by showing the same streak surfaces as in (a) and (b) with the only change being the computation of $\boldsymbol{\alpha}_{shape}$.

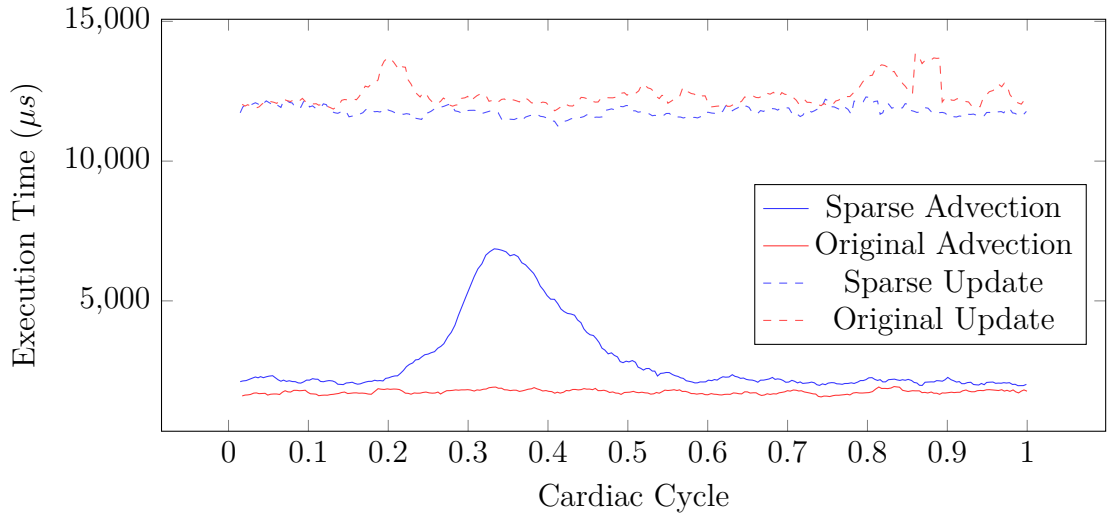


Figure 4.5: Execution times in μs of the advection process (solid line) and the mesh update process (dashed line) ploted over the course of one cardiac cycle. The performance of the original voxel grid data structure (red) is compared to the sparse representation (blue).

4.3.3 Performance

The execution time of the preprocessing steps does not impact the visualization, but the performance of computations during interactive real-time visualization is critical. Inadequate performance can result in an unresponsive application or even failure to deliver real-time visualization altogether. The major processes involved in creating an interactive flow visualization of *Smoke Surfaces* are (1) advecting the streak surface mesh's vertices, (2) updating any surface parameters, and (3) rendering the streak surface as well as any additional geometry.

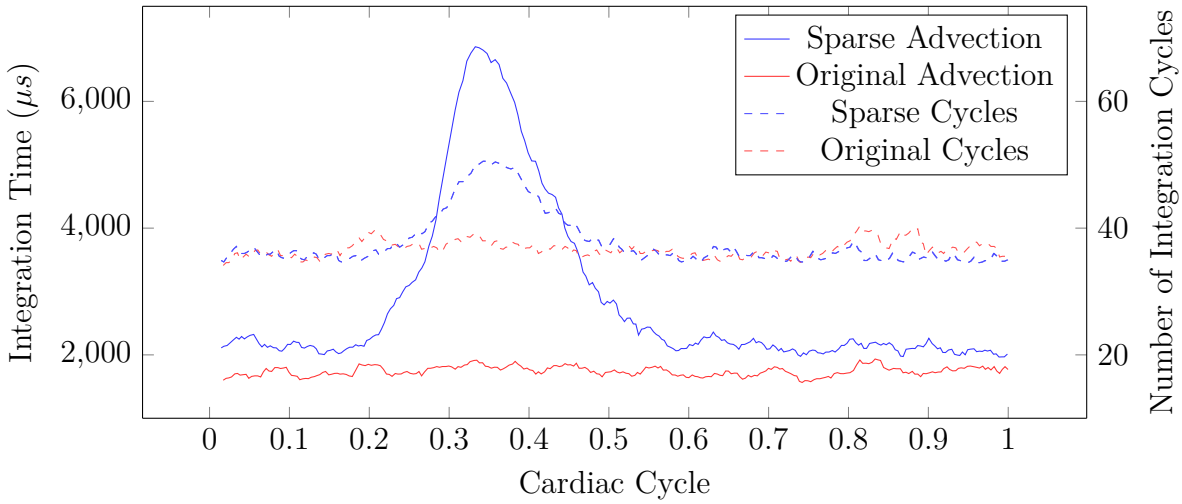


Figure 4.6: Execution time of the advection process (solid line) and integration steps (cycles) computed per frame (dashed line) are plotted over the course of one cardiac cycle. The performance of the original voxel grid data structure (red) is compared to the sparse representation (blue).

Achieving interactive frame rates (>30 frames per second (FPS)) necessitates ensuring that the combined execution time of these processes remains below 33 ms. Consequently, optimizing computation efficiency across all processing steps is paramount. The advection process frequently accesses the simulated flow data, which is optimized through the use of a regular voxel grid, ensuring constant access times. Furthermore, all three processes mentioned above are accelerated using parallel processing on the GPU. This enables simultaneous processing of numerous mesh vertices. Similarly, ray intersection tests required for rendering are executed in parallel.

The performance of processing step (1) and (2) was evaluated using a streak surface comprising 25,000 vertices. The sparse representation of the voxel grid introduces additional computational steps to ensure access to the correct data segments. Application of temporal as well as spatial interpolation and the 4th order Runge-Kutta integration scheme result in 64 accesses to the flow data for each integration step. Given that the computations introduced by the sparse representation must be performed for every access, they potentially have a significant impact on performance. Hence, we conducted performance testing of the advection shader using both the original and sparse voxel grid data structures.

During testing, we maintained consistency in several parameters, including the size and location of the seeding structure, the size of the streak surface mesh, animation speed, rendered geometry, and temporal as well as spatial resolution of the input data. This ensured that any observed differences in performance could be attributed specifically to the variations in the data structures being tested, rather than fluctuations in other parameters.

The performance results for (1) and (2) are plotted over one cardiac cycle in Figure 4.5. In the original data structure (depicted in red), both the advection and mesh update stages exhibit constant execution times, averaging 1.7 ms and 12.3 ms, respectively. Since the mesh update stage is unaffected by the change in data structure, the measured

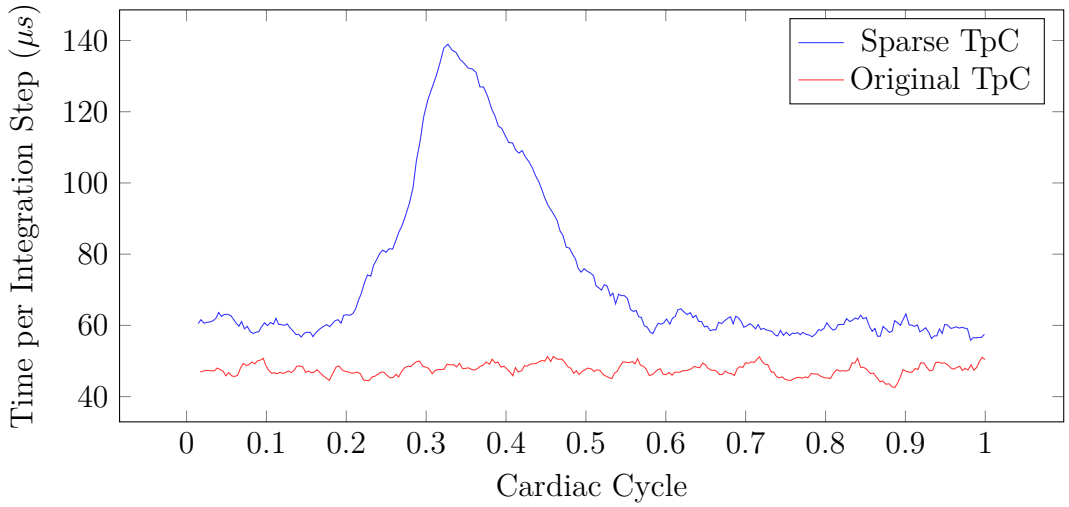


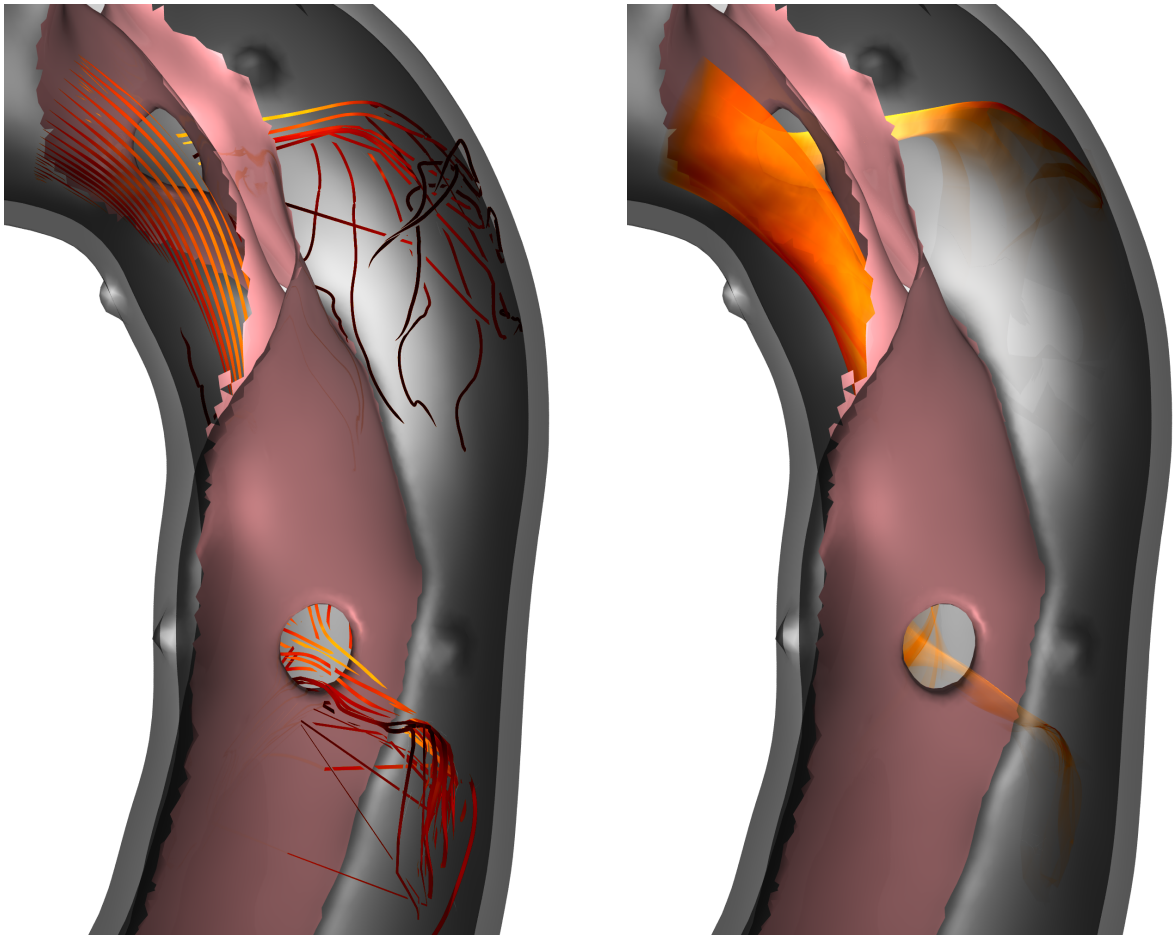
Figure 4.7: The execution time per integration step (TpC) is plotted over the course of one cardiac cycle. The performance of the original voxel grid data structure (red) is compared to the sparse representation (blue).

performance of mesh update for the sparse data structure remains comparable to the original, with an average execution time of 12.3 ms.

When examining the plot corresponding to the sparse data structure (illustrated in blue), the execution time of the advection process (solid line) appears slightly increased compared to the original structure for most of the cardiac cycle, as expected. However, at approximately 35% of the cardiac cycle, the execution time increases significantly. This phenomenon occured consistently with large streak surface meshes and persisted even after attempts to remedy the issue.

Figure 4.6 shows the overall execution time of the advection shader per frame alongside the number of integration steps (cycles) computed in that frame. The number of cycles is derived from the frame rate, to keep the speed of the simulation steady. A notable increase of the number of cycles, indicated by the blue dashed curve, occurs at approximately 35% of the cardiac cycle. As expected, with the increase in the number of cycles computed per frame, the execution time also rises. However, it is observed that the execution time increases disproportionately to the number of cycles. This observation is supported by the data presented in Figure 4.7, which displays the execution time per cycle. A significant increase in the per-cycle execution time is evident in the blue curve, corresponding to the sparse data structure. In contrast, this phenomenon is not apparent in the original data structure (red).

All processes involved in our real-time flow visualization are designed to maintain a constant execution time. For this reason, the cause of the sudden increase in execution time is unclear. Potential causes could include optimization or the lack thereof in the execution of Vulkan compute shaders, memory transfers, or other internal processes of the Vulkan API. Further investigation and debugging are warranted to identify and address this issue effectively.



(a) **(b)**

Figure 4.8: Rendering of flow in dataset 1 using streak lines (a) and a streak surface (b). The seeding structure is placed in the true lumen in the aortic arch, close before the entry trear. Depth-dependent halos [81] are applied to the streaklines to reduce clutter. Additionally, both techniques employ a color mapping of the velocity magnitude with dark red indicating slow flow and bright yellow indicating fast flow.

4.4 Appearance

In this section, we present the results of our smoke-like visualization of aortic dissection flow and discuss its application in analyzing various flow phenomena and hemodynamic aspects of the disease. Throughout our analysis, we frequently compare the *Smoke Surfaces* to the previously implemented streak lines computed using the same data and integration techniques but rendered differently.

Unlike the streak surfaces, no opcacity modulation is applied to the streak lines. Instead, depth-dependent halos [81] are added to reduce clutter. It is important to note that the streak line visualization is not part of this work and only serves as example for techniques commonly used in the visualization of hemodynamics.

Figure 4.8 shows dataset 1 with streak lines (a) alongside a streak surface (b) seeded in the true lumen in the aortic arch, close before the entry tear. Both techniques employ velocity magnitude color mapping, representing slow flow in dark red and fast flow in bright yellow. The seeding location, length of lines/surface, and color mapping parameters are consistent for figure (a) and (b). Both images were aquired at the same time in the cardiac cycle, during late systole. Notably, a fenestration in the dissection flap is visible at the center of both figures.

Upon comparison, both streak lines and the streak surface clearly depict blood flowing through the true lumen, entering the false lumen through the entry tear, and traversing the fenestration. The color mapping effectively portrays the velocity of the flow, highlighting its acceleration as it passes through the narrow fenestration in the dissection flap.

Upon closer examination, the *Smoke Surface* appears less cluttered, offering a more continuous and fine-grained visualization of the flow dynamics. Furthermore, the surface representation enables a more intuitive understanding of flow. For instance, as the blood flows through the entry tear, it forms a flow jet that impacts the opposing outer vessel wall of the false lumen. Consequently, the flow is dispersed, moving along the vessel wall and predominantly down the false lumen. While streak lines can depict the flow jet and its deflection at the vessel wall, the streak surface excels at illustrating the thin, cohesive flow jet and its dispersion at the vessel wall. The reduction in opacity intuitively conveys the dynamics of the phenomenon, whereas the same understandig needs to be actively deduced from the increasing distance between lines when observing streak lines.

Due to the continuous advection of each vertex of a streak line or surface, both techniques suffer from segments being stretched as the flow diverges (recall Subsection 2.4.2). However, when using *Smoke Surfaces*, the resulting artifacts are hidden, which is not the case with streak lines. The artifacts appearing where the flow diverges can be seen in Figure 4.8a, where the flow passes through the fenestration. Erroneous, long, straight sections of streak lines appear to be passing through the dissection flap. In contrast, the same segments are hidden in the streak surface, producing a smoke-like visualization of flow without artifacts.

Figure 4.9 presents another set of images diplaying both streak lines and streak surfaces. The generation parameters for all images only differ in the technique used for flow visualization and the point in time during the cardiac cycle. The left

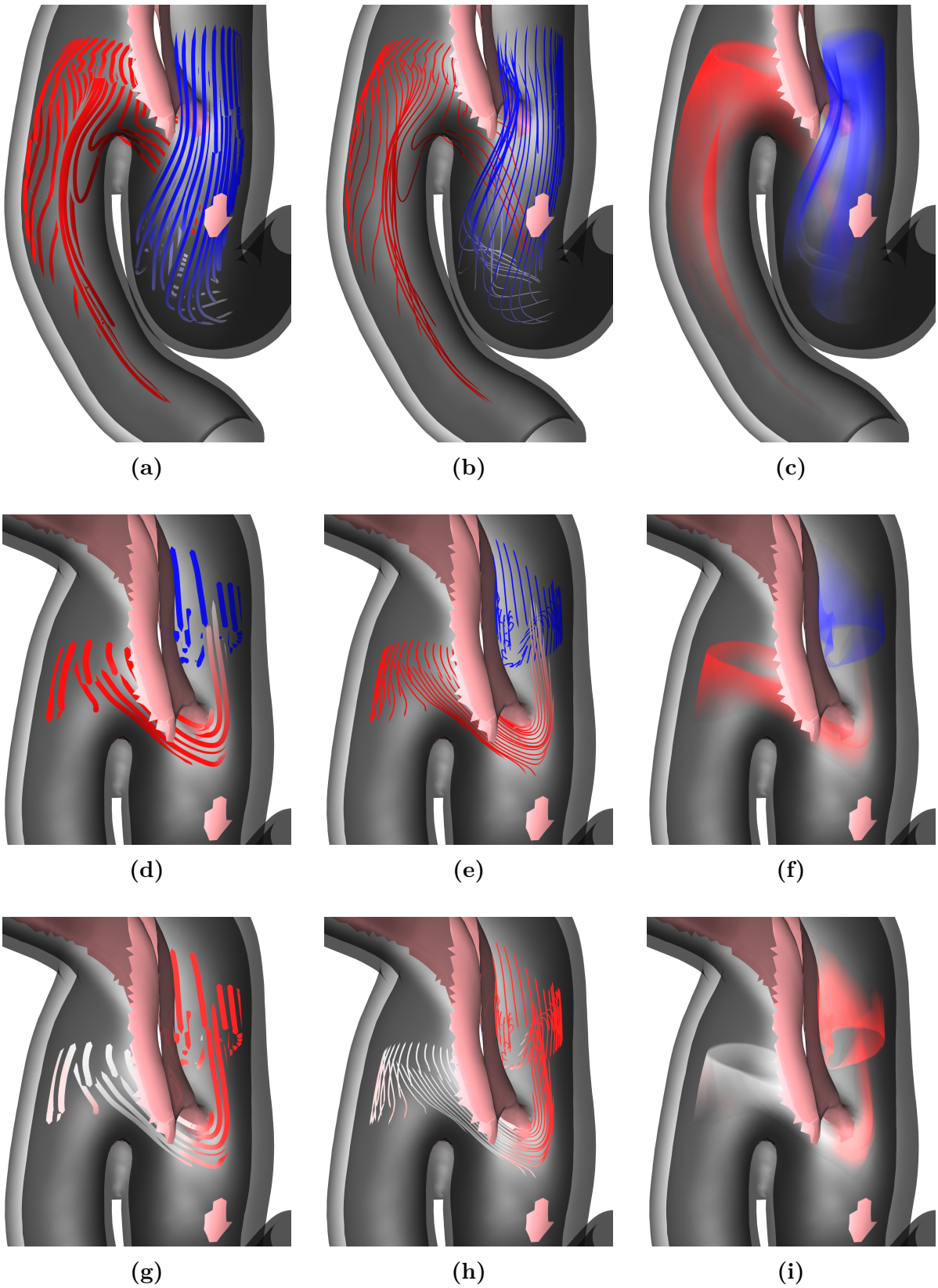


Figure 4.9: Various visualizations of flow at the reentry tear in the left femoral artery of dataset 2. Figures (a)-(c) are captured during systole, while figures (d)-(i) are captured during diastole. The left column displays a low number of streak lines with depth-dependent halos [81] to reduce clutter. The center column shows a larger number of streak lines without halos, enabling a more detailed visualization of flow. The right column shows two streak surfaces. In the top two rows, color mapping is employed to represent the flow's lumen-of-origin (red for true lumen, blue for false lumen). In the bottom row, color mapping represents the flow direction (white for antegrade, red for retrograde).

column of figures employs streak lines with depth-dependent halos [81], the center column uses streak lines without halos and the right column utilizes streak surfaces. Figures (a)-(c) are captured during systole, while figures (d)-(i) are captured during diastole.

Comparing the top row of images, all the techniques produce visualizations of flow, clearly showing the forward flow of true and false lumen, including the true lumen's contribution to the profunda femoris. The lumen-of-origin color mapping (red for true lumen, blue for false lumen) applied to all three techniques aids in this visualization. While the depth-dependent halos in figure (a) succeed in reducing clutter comparared to (b), figure (c) clearly appears the least cluttered, simultaneously exceeding the level of detail displayed through an increased number of streak lines in figure (b). The halos employed in (a) help distinguish lines in the foreground from lines in the background. However, this advantage diminishes as the spatial configuration becomes intuitively recognizable in all three techniques by rotating the view. Arguably, the transparent streak surfaces allow for the best interpretation of the entire structure during interactive observation as self-occlusion is reduced. Similar observations can be made, when comparing the three techniques in the images in the second and third rows in Figure 4.9. Both rows visualize the flow at the reentry tear during diastole. Streak surfaces produce the least cluttered visualizations, while displaying equal or greater amounts of detail.

All three techniques are able to clearly show blood flowing from the true lumen into the false lumen, as depicted in figures (d)-(f), thanks to the lumen-of-origin color mapping. As this flow occurs at the reentry tear of the dissection and moves up the false lumen, retrograde flow occurs. This is further highlighted in figures (g)-(i), which employ the flow direction color mapping, which displays antegrade flow in white and retrograde flow in red. The techniques based on streak lines are able to show retrograde flow. However, the smooth appearance of the *Smoke Surfaces* makes them easier to interpret, especially during animation.

To display the constantly changing flow inside an aortic dissection and demonstrate the ability of streak surfaces to visualize such flow, Figure 4.10 presents six images of a streak surface seeded in the true lumen in the aortic arch of dataset 1. The surface is advected throughout the cardiac cycle, conforming to the changing flow. Each image captures the surface at different point in time, with (a) representing the start of systole, (c) peak systole and (f) the end of diastole.

In figure (a) the streak surface appears as a short section at the top of the aortic arch, as the flow velocity is very small shortly before systole. Figure (b) shows the blood accelerating and flowing through the entry tear, into the false lumen in the form of a flow jet, as the systole begins. The velocity magnitude color mapping applied to the streak surface in all images highlights the increased velocity at the entry tear compared to the true lumen. Peak velocity is reached in the true lumen at the start of the descending aorta during peak systole, as shown in figure (c). Blood flows down the true lumen and into the false lumen simultaneously. Comparing the streams through the entry tear in (b) and (c), complex hemodynamics can be observed, as the flow jet changes over time. The streak surface also demonstrated how the blood flowing into the false lumen distributes after it is deflected by the false lumen outer wall.

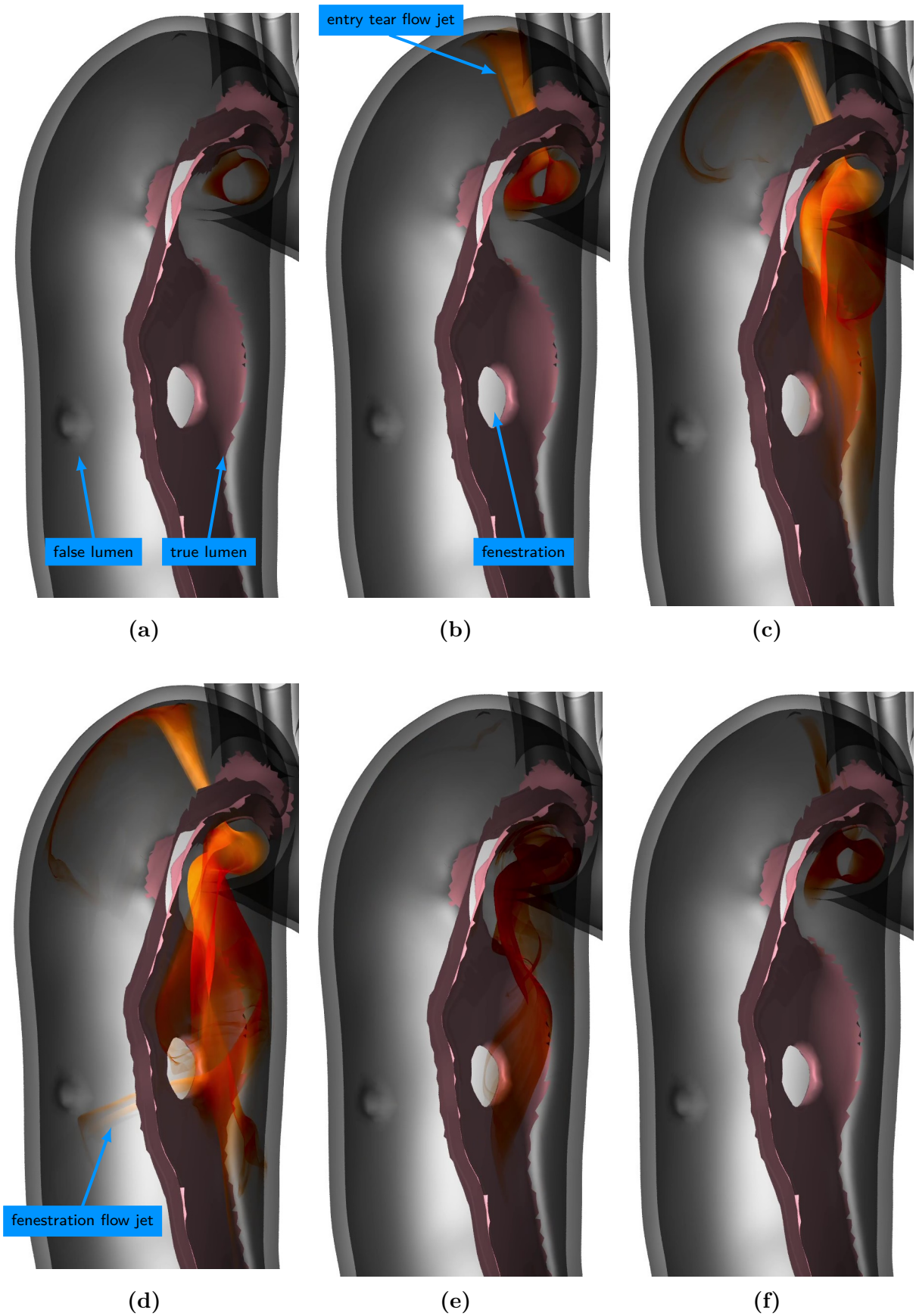


Figure 4.10: Rendering of a streak surface seeded in the true lumen in the aortic arch of dataset 1. Figures (a) through (f) depict the same streak surface at distinct, equally spaced times over the course of one cardiac cycle. A color mapping of the velocity magnitude is applied, showing slow flow in dark red and fast flow in bright yellow.

Figure (d) shows the streak surface extending into the false lumen at both the entry tear and the fenetration located further down the dissection. The false lumen experiencing inflow through both fenestrations is clearly illustrated by this. This can help improve the understanding of false lumen pressurization, as the small exit tear may not provide enough outflow. Appart from the flow through fenestrations, the streak surface also displays the constantly changing, partially helical flow inside the aorta.

Figure (e) represents the peak of diastole, with the flow slowing down significantly, as indicated by the color mapping. Finally, the blood flow almost stops completely towards the end of the diastole. Still, some flow through the true lumen as well as into the false lumen can be seen in figure (f).

4.5 Limitations

Smoke Surfaces were successfully empoyed to simulate the injection and advection of smoke in the flow of aortic dissection. During testing a number of limitations became appearant.

Dispite our efforts to hide artifacts appearing where flow diverges, small artifacts inherent to streak surfaces without remeshing remain. They could only be observed for a very short period, when a streak surface passes through a fenestration, during our testing.

Due to the design of our ring seeding structure and its placement, investigation of the flow around intricate geometries such as fenestrations often requires the user to adjust size and location of the seeding structure. The reason for this is the fact, that blood flow near the vessel wall often differs significantly from the flow in the center of a lumen While the seeding structure can be adjusted in real time, which promotes exploratory investigation, simultaneous seeding at the wall and center of a lumen is currently not supported.

As presented in Subsection 4.3.3, advection performance fluctuates depending on the number of streak surface mesh vertices. Additionally, larger streak surfaces tend to lead to more overlapping transparent surfaces, which slows down rendering. This effectively limits the size and resolution of streak surfaces, as interactive framerates are otherwise unattainable.

5 Conclusion and Future Work

This thesis aimed to create smoke-like visualizations of the complex hemodynamics in and around aortic dissections to improve visualization clarity and information density while providing intuitive means of analyzing blood flow. Different techniques based on particles, surfaces, and volumes where discussed and a streak surface technique developed by von Funck et al. [1] selected for implementation. This technique is tailored to provide real-time visualizations of flow with a smoke-like appearance, by modulating the surfaces opacity based on multiple factors. The technique was used to display the blood flow inside aortic dissection and combined with a surface representation of the vessel morphology, creating a focus and context visualization. The simultaneous display of morphology and hemodynamics, can provide valuable insight into the complex interplay between hemodynamics and morphology. Additionally, real-time advection and rendering of streak surfaces allows for continuous exploration of the intricate dynamics of aortic dissections.

5.1 Contributions

The opacity modulation of the streak surface proposed by von Funck et al. [1] succeeded in hiding the majority of artifacts produced by streak surface advection without modification of the mesh, but significant artifacts appeared when applying the technique to the unique geometry of aortic dissections. We adapted the shape opacity term to mitigate this issue by introducing the maximum edge length of a triangle as an exponent. As a result, the majority of the surface appears as before, while hiding artifacts, appearing as long, straight lines, intersecting parts of the vessel.

Another adaption of the original work was performed in augmenting the seeding structure. *Smoke Surfaces* were developed to use a continuous curve as a seeding structure to simulate a coherend surface of smoke. Another option presented was seeding at multiple distinct locations to simulate wool tufts. We adapt their seeding structure to form a closed ring instead of a curve. This seeding structure leads to a tubular topology in the streak surface, inspired by the tubular structure of blood vessels.

Mapping color to integral structures is a technique commonly used in flow visualization. The effectiveness of color mapping on *Smoke Surfaces* was not tested in the original work. We apply a commonly used color mapping of flow velocity magnitude to the streak surface to encode more information. Additionally, we present two new color mappings, namely lumen-of-origin and flow direction, specifically developed for the use in aortic dissections. All three color mappings improve the interpretability of the visualization by providing additional information. The lumen-of-origin color mapping significantly simplifies the identification of blood flow originating from either true or false lumen and the interaction of both lumina.

5.2 Future Work

To the best of our knowledge, we have made the first attempt at enhancing flow visualizations of aortic dissections through a smoke-like technique. Therefore, there is potential for refining and extending the methods presented in our work or investigating the effectiveness of different techniques entirely.

5.2.1 Smoke Surfaces

Von Funck's [1] succeeds in creating real-time smoke-like visualizations of aortic hemodynamics. Nonetheless, several aspects such as seeding, color mapping and performance could be improved.

While our ring seeding structure samples an area of specified distance from the center of a cross-section, different seeding structures could be used to ensure more even sampling. For instance, placing a spiral or multiple rings in a cross-section to seed from improves the sampling across the cross-section but introduces overlapping streak surfaces, potentially compromising their comprehensibility. Moreover, augmenting color mapping on streak surfaces by integrating additional metrics such as vorticity, helicity, or residence time could enrich the visual representation.

Although the advection performance of our implementation proved adequate for real-time usage and may have the capacity to accommodate larger streak surfaces than those tested, the synchronization with the chosen rendering solution imposes limitations on performance. Implementing synchronization techniques that facilitate parallel execution of advection and rendering processes could yield significant performance enhancements. Furthermore, optimizations to the rendering pipeline itself hold promise for improving overall efficiency.

5.2.2 Flow Visualization Techniques

In addition to the streak surface-based technique we selected for our visualization, particle and volume based techniques were also discussed in Section 2.4. Particle-based techniques, widely employed across various applications, have been utilized for visualizing aortic hemodynamics by de Hoon et al. [58]. Given their versatility, it is conceivable that minimal adaptation would be necessary to apply these methods to aortic dissection datasets.

Volume rendering is widely used for 3D visualizations of medical volume data. Using similar rendering techniques to visualize hemodynamics, seems obvious. However, flow visualizations based on volumetric structures have seen little to no advancement since their initial development by Max et al. [4]. Nonetheless, volumetric flow visualization could be considered when investigating alternative techniques for smoke-like renderings of aortic dissection flow.

5.2.3 Ray Tracing

Highly realistic visualizations can be created using ray tracing. As mentioned in Subsection 3.5.1, only basic rendering effects were implemented in this work, as lighting effects are out of scope.

Lighting effects such as realistic shadows and ambient occlusion can enhance depth perception. However, casting shadows inside vascular structures is non-trivial and may require specially tailored techniques. Accurate reflections computed using ray tracing could be employed to improve the visualization through virtual mirrors, first introduced by Bichlmeier et al. [82]. Virtual mirrors were originally developed to improve depth perception interoperatively, but they can also provide a better overview of a structure by offering views from different directions. This functionality may prove particularly useful when investigating aortic dissections using a transparent outer wall and opaque dissection flap.

When viewing one lumen, the dissection flap may obstruct the view of the other lumen, necessitating the relocation of the camera to obtain a comprehensive view. However, with a virtual mirror, both the front of the dissection flap and the currently viewed lumen can be seen simultaneously with the back side of the dissection flap and the flow in the opposing lumen. This simultaneous visualization may significantly enhance the understanding of the spatial relationships and dynamic interactions of true and false lumen within the dissected aorta.

The addition of advanced lighting effects such as global illumination or subsurface scattering can improve the visual fidelity of the visualization but impacts performance, potentially compromising the interactivity of our visualization. While advanced lighting can produce highly realistic renderings, which may be used for patient communication, it may not necessarily aid in diagnosis or treatment planning. This necessitates further research to evaluate the clinical utility of such enhancements.

Bibliography

[1] W. von Funck, T. Weinkauf, H. Theisel, and H.-P. Seidel. Smoke surfaces: An interactive flow visualization technique inspired by real-world flow experiments. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1396–1403, November 2008.

[2] Roy van Pelt, Javier Olivan Bescos, Marcel Breeuwer, Rachel E. Clough, M. Eduard Groller, Bart ter Haar Romenij, and Anna Vilanova. Interactive virtual probing of 4d mri blood-flow. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2153–2162, December 2011.

[3] Mathias Hummel, Christoph Garth, Bernd Hamann, Hans Hagen, and Kenneth I. Joy. Iris: Illustrative rendering for integral surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1319–1328, 2010.

[4] Nelson Max, Becker G., and Roger Crawfis. Flow volumes for interactive vector field visualization. In *Proceedings Visualization*, VISUAL-93, pages 19–24. IEEE Comput. Soc. Press, November 1993.

[5] Xun Yuan, Andreas Mitsis, and Christoph A. Nienaber. Current understanding of aortic dissection. *Life*, 12(10):1606, October 2022.

[6] Maya Landenhed, Gunnar Engström, Anders Gottsäter, Michael P. Caulfield, Bo Hedblad, Christopher Newton-Cheh, Olle Melander, and J. Gustav Smith. Risk profiles for aortic dissection and ruptured or surgically treated aneurysms: A prospective cohort study. *Journal of the American Heart Association*, 4(1), January 2015.

[7] Dominic P.J. Howard, Amitava Banerjee, Jack F. Fairhead, Jeremy Perkins, Louise E. Silver, and Peter M. Rothwell. Population-based study of incidence and outcome of acute aortic dissection and premorbid risk factor control: 10-year results from the oxford vascular study. *Circulation*, 127(20):2031–2037, May 2013.

[8] Maximilian Wundram, Volkmar Falk, Jaime-Jürgen Eulert-Grehn, Hermann Herbst, Jana Thurau, Bernd A. Leidel, Eva Göncz, Wolfgang Bauer, Helmut Habazettl, and Stephan Kurz. Incidence of acute type A aortic dissection in emergency departments. *Scientific Reports*, 10(1):7434, May 2020.

[9] Kathrin Bäumler, Vijay Vedula, Anna Sailer, Jongmin Seo, Peter Chiu, Gabriel Mistelbauer, Frandics Chan, Michael Fischbein, Alison Marsden, and Dominik Fleischmann. Fluid–structure interaction simulations of patient-specific aortic dissection. *Biomechanics and Modeling in Mechanobiology*, 19(5):1607–1628, January 2020.

[10] Domenico Spinelli, Filippo Benedetto, Rocco Donato, Gabriele Piffaretti, Massimiliano Marrocco-Trischitta, Himanshu Patel, Kim Eagle, and Santi Trimarchi. Current evidence in predictors of aortic growth and events in acute type B aortic dissection. *Journal of Vascular Surgery*, 68(6):1925–35, August 2018.

[11] Jin Wook Chung, Christopher Elkins, Toyohiko Sakai, Noriyuki Kato, Thomas Vestring, Charles Semba, Suzanne Slonim, and Michael Dake. True-lumen collapse in aortic dissection: part I. Evaluation of causative factors in phantoms with pulsatile flow. *Radiology*, 214(1):87–98, February 2000.

[12] Jin Wook Chung, Christopher Elkins, Toyohiko Sakai, Noriyuki Kato, Thomas Vestring, Charles P. Semba, Suzanne M. Slonim, and Michael D. Dake. True-lumen collapse in aortic dissection: part II. Evaluation of treatment methods in phantoms with pulsatile flow. *Radiology*, 214(1):99–106, 2000.

[13] Ramon Berguer, Juan Parodi, Marty Schlicht, and Khalil Khanafer. Experimental and clinical evidence supporting septectomy in the primary treatment of acute type B thoracic aortic dissection. *Annals of Vascular Surgery*, 29(2):167–73, October 2015.

[14] Thomas T. Tsai, Marty S. Schlicht, Khalil Khanafer, Joseph L. Bull, Doug T. Valassis, David M. Williams, Ramon Berguer, and Kim A. Eagle. Tear size and location impacts false lumen pressure in an ex vivo model of chronic type B aortic dissection. *Journal of Vascular Surgery*, 47(4):844–51, April 2008.

[15] Zhuo Cheng, F. Tan, C. Riga, Colin Bicknell, Mohamad Hamady, Richard Gibbs, Nigel Wood, and Xiao Xu. Analysis of flow patterns in a patient-specific aortic dissection model. *Journal of Biomechanical Engineering*, 132(5):051007, May 2010.

[16] Barry Doyle and Paul Norman. Computational biomechanics in thoracic aortic dissection: Today's approaches and tomorrow's opportunities. *Annals of Biomedical Engineering*, 44(1):71–83, June 2015.

[17] Anna Sailer, Sander Kuijk, Patricia Nelemans, Anne Chin, Aya Kino, Mark Huininga, Johanna Schmidt, Gabriel Mistelbauer, Kathrin Bäumler, Peter Chiu, Michael Fischbein, Michael Dake, D. Miller, Geert Schurink, and Dominik Fleischmann. Computed tomography imaging features in acute uncomplicated stanford type-B aortic dissection predict late adverse events. *Circulation: Cardiovascular Imaging*, 10(4):e005709, April 2017.

[18] Anna Sailer, Patricia Nelemans, Trevor Hastie, Anne Chin, Mark Huininga, Peter Chiu, Michael Fischbein, Michael Dake, D. Miller, Geert Schurink, and Dominik Fleischmann. Prognostic significance of early aortic remodeling in acute uncomplicated type B aortic dissection and intramural hematoma. *The Journal of Thoracic and Cardiovascular Surgery*, 154(4):1192–200, October 2017.

[19] Zoran Stankovic, Bradley D. Allen, Julio Garcia, Kelly B. Jarvis, and Michael Markl. 4d flow imaging with mri. *Cardiovascular Diagnosis and Therapy*, 4(2), 2014.

[20] Rachel E. Clough, Matthew Waltham, Daniel Giese, Peter R. Taylor, and Tobias Schaeffter. A new imaging method for assessment of aortic dissection using four-dimensional phase contrast magnetic resonance imaging. *Journal of Vascular Surgery*, 55(4):914–23, 2012.

[21] Christopher J. François, Michael Markl, Mark L. Schiebler, Eric Niespodzany, Benjamin R. Landgraf, Christian Schlensak, and Alex Frydrychowicz. Four-dimensional, flow-sensitive magnetic resonance imaging of blood flow patterns in thoracic aortic dissections. *The Journal of Thoracic and Cardiovascular Surgery*, 145(5):1359–66, May 2013.

[22] Liang Zhou, Mengjie Fan, Charles Hansen, Chris R. Johnson, and Daniel Weiskopf. A review of three-dimensional medical image visualization. *Health Data Science*, 2022, January 2022.

[23] Yusman Azimi Yusoff, Farhan Mohamad, Mohd Shahrizal Sunar, and Ali Selamat. *Flow Visualization Techniques: A Review*, pages 527–538. Springer International Publishing, 2016.

[24] Steffen Oeltze-Jafra, Monique Meuschke, Mathias Neugebauer, Sylvia Saalfeld, Kai Lawonn, Gabor Janiga, Hans-Christian Hege, Stefan Zachow, and Bernhard Preim. Generation and visual exploration of medical flow data: Survey, research trends, and future challenges. *Computer Graphics Forum*, 38, March 2018.

[25] Benjamin Köhler, Matthias Grothoff, Matthias Gutberlet, and Bernhard Preim. Bloodline: A system for the guided analysis of cardiac 4D PC-MRI data. *Computers & Graphics*, 82:32–43, January 2019.

[26] Pepe Eulzer, Monique Meuschke, Carsten Klingner, and Kai Lawonn. Visualizing carotid blood flow simulations for stroke prevention, 2021.

[27] R. Gasteiger, M. Neugebauer, C. Kubisch, and B. Preim. Adapted surface visualization of cerebral aneurysms with embedded blood flow information. In *Eurographics Workshop on Visual Computing for Biology and Medicine*. The Eurographics Association, 2010.

[28] Benjamin Behrendt, Wito Engelke, Philipp Berg, Oliver Beuing, Ingrid Hotz, Bernhard Preim, and Sylvia Saalfeld. Visual exploration of intracranial aneurysm blood flow adapted to the clinical researcher. In *Eurographics – Dirk Bartz Prize*. The Eurographics Association, June 2021.

[29] Kai Ostendorf, Domenico Mastrodicasa, Kathrin Bäumler, Marina Codari, Valery Turner, Martin J. Willemink, Dominik Fleischmann, Bernhard Preim, and Gabriel Mistelbauer. Shading style assessment for vessel wall and lumen visualization. *Eurographics Workshop on Visual Computing for Biology and Medicine*, 2021.

[30] Zachary A. Zilber, Aayush Boddu, S. Chris Malaisrie, Andrew W. Hoel, Christopher K. Mehta, Patricia Vassallo, Nicholas S. Burris, Alejandro Roldán-Alzate, Jeremy D. Collins, Christopher J. François, and Bradley D. Allen. Noninvasive morphologic and hemodynamic evaluation of type b aortic dissection: State of the art and future perspectives. *Radiology: Cardiothoracic Imaging*, 3(3):e200456, June 2021.

[31] Jong-Min Song, Sung-Doo Kim, Jeong-Hoon Kim, Mi-Jeong Kim, Duk-Hyun Kang, Joon Beom Seo, Tae-Hwan Lim, Jae Won Lee, Meong-Gun Song, and Jae-Kwan Song. Long-term predictors of descending aorta aneurysmal change in patients with aortic dissection. *Journal of the American College of Cardiology*, 50(8):799–804, August 2007.

[32] Chih-Ping Chang, Juhn-Cherng Liu, Ying-Ming Liou, Shih-Sheng Chang, and Jan-Yow Chen. The role of false lumen size in prediction of in-hospital complications after acute type b aortic dissection. *Journal of the American College of Cardiology*, 52(14):1170–1176, September 2008.

[33] Kedar S. Lavingia, Sebastion Larion, Sadaf S. Ahanchi, Chad P. Ammar, Mohit Bhasin, Aleem K. Mirza, David J. Dexter, and Jean M. Panneton. Volumetric analysis of the initial index computed tomography scan can predict the natural history of acute uncomplicated type b dissections. *Journal of Vascular Surgery*, 62(4):893–899, October 2015.

[34] Arturo Evangelista, Antonio Salas, Aida Ribera, Ignacio Ferreira-González, Helena Cuéllar, Vicente Pineda, Teresa González-Alujas, Rosario Dominguez, Enric Batlle, and Gaietá Permanyer-Miralda. Long-term outcome of aortic dissection with patent false lumen: predictive role of entry tear size and location. *Circulation*, 125(25):3133–3141, 2012.

[35] Jip L. Tolenaar, Jasper W. van Keulen, Santi Trimarchi, Frederik H.W. Jonker, Joost A. van Herwaarden, Hence J.M. Verhagen, Frans L. Moll, and Bart E. Muhs. Number of entry tears is associated with aortic growth in type b dissections. *The Annals of Thoracic Surgery*, 96(1):39–42, July 2013.

[36] David Marlevi, Julio A. Sotelo, Ross Grogan-Kaylor, Yunus Ahmed, Sergio Uribe, Himanshu J. Patel, Elazer R. Edelman, David A. Nordsletten, and Nicholas S. Burris. False lumen pressure estimation in type B aortic dissection using 4D flow cardiovascular magnetic resonance: comparisons with aortic growth. *Journal of Cardiovascular Magnetic Resonance*, 23(1), May 2021.

[37] Prateek Gupta, Himani Gupta, and Ali Khoynezhad. Hypertensive emergency in aortic dissection and thoracic aortic aneurysm—a review of management. *Pharmaceuticals*, 2(3):66–76, September 2009.

[38] Foeke JH Nauta, Santi Trimarchi, Arnoud V Kamman, Frans L Moll, Joost A van Herwaarden, Himanshu J Patel, C Alberto Figueroa, Kim A Eagle, and James B Froehlich. Update in the management of type b aortic dissection. *Vascular Medicine*, 21(3):251–263, April 2016.

[39] Zhuo Cheng, Nigel Wood, Richard Gibbs, and Xiao Xu. Geometric and flow features of type B aortic dissection: Initial findings and comparison of medically treated and stented cases. *Annals of Biomedical Engineering*, 43(1):177–89, May 2014.

[40] Eric K. Shang, Derek P. Nathan, Ronald M. Fairman, Joseph E. Bavaria, Robert C. Gorman, Joseph H. Gorman, and Benjamin M. Jackson. Use of computational fluid dynamics studies in predicting aneurysmal degeneration of acute type B aortic dissections. *Journal of Vascular Surgery*, 62(2):279–284, August 2015.

[41] Claudia Menichini and Xiao Yun Xu. Mathematical modeling of thrombus formation in idealized models of aortic dissection: initial findings and potential applications. *Journal of Mathematical Biology*, 73(5):1205–1226, March 2016.

[42] A. Osswald, C. Karmonik, J.R. Anderson, F. Rengier, M. Karck, J. Engelke, K. Kallenbach, D. Kotelis, S. Partovi, D. Böckler, and A. Ruhparwar. Elevated wall shear stress in aortic type B dissection may relate to retrograde aortic type A

dissection: A computational fluid dynamics pilot study. *European Journal of Vascular and Endovascular Surgery*, 54(3):324–330, September 2017.

[43] Bijit Munshi, Louis P. Parker, Paul E. Norman, and Barry J. Doyle. The application of computational modeling for risk prediction in type b aortic dissection. *Journal of Vascular Surgery*, 71(5):1789–1801.e3, May 2020.

[44] Kwong Ming Tse, Peixuan Chiu, Heow Pueh Lee, and Pei Ho. Investigation of hemodynamics in the development of dissecting aneurysm within patient-specific dissecting aneurismal aortas using computational fluid dynamics (cfd) simulations. *Journal of Biomechanics*, 44(5):827–836, March 2011.

[45] Yongxue Zhang, Qingsheng Lu, Jiaxuan Feng, Peng Yu, Suming Zhang, Zhongzhao Teng, Jonathan H. Gillard, Runze Song, and Zaiping Jing. A pilot study exploring the mechanisms involved in the longitudinal propagation of acute aortic dissection through computational fluid dynamic analysis. *Cardiology*, 128(2):220–225, 2014.

[46] Bradley D. Allen, Pascale J. Aouad, Nicholas S. Burris, Amir Ali Rahsepar, Kelly B. Jarvis, Christopher J. François, Alex J. Barker, S. Chris Malaisrie, James C. Carr, Jeremy D. Collins, and Michael Markl. Detection and hemodynamic evaluation of flap fenestrations in type b aortic dissection with 4d flow mri: Comparison with conventional mri and ct angiography. *Radiology: Cardiothoracic Imaging*, 1(1):e180009, April 2019.

[47] Andrew Sherrah, Fraser Callaghan, Rajesh Puranik, Richmond Jeremy, Paul Bannon, Michael Vallely, and Stuart Grieve. Multi-velocity encoding four-dimensional flow magnetic resonance imaging in the assessment of chronic aortic dissection. *AORTA*, 05(03):80–90, June 2017.

[48] Timothy Urness, Victoria Interrante, I. Marusic, Ellen Longmire, and Bharathram Ganapathisubramani. Effectively visualizing multi-valued flow data using color and texture. In *IEEE Visualization*, pages 115–21, November 2003.

[49] Andrea Brambilla, Robert Carnecky, Ronald Peikert, Ivan Viola, and Helwig Hauser. Illustrative flow visualization: State of the art, trends and challenges. In Marie-Paule Cani and Fabio Ganovelli, editors, *Eurographics 2012 – State of the Art Reports*, pages 75–94, 2012.

[50] Silvia Born, Michael Markl, Matthias Gutberlet, and Gerik Scheuermann. Illustrative visualization of cardiac and aortic blood flow from 4d mri data. In *2013 IEEE Pacific Visualization Symposium (PacificVis)*, pages 129–136. IEEE, February 2013.

[51] K. Lawonn, R. Gasteiger, and B. Preim. Adaptive surface visualization of vessels with animated blood flow. *Computer Graphics Forum*, 33(8):16–27, 2014.

[52] J. Krüger, P. Kipfer, P. Konclratieva, and R. Westermann. A particle system for interactive visualization of 3d flows. *IEEE Transactions on Visualization and Computer Graphics*, 11(6):744–756, November 2005.

[53] Peter Kipfer, Mark Segal, and Rüdiger Westermann. Uberflow: a gpu-based particle engine. In *ACM SIGGRAPH 2004 Sketches*, SIGGRAPH '04, page 24, New York, NY, USA, 2004. Association for Computing Machinery.

[54] J.J. van Wijk. Rendering surface-particles. In *Proceedings Visualization '92*, pages 54–61, 1992.

[55] Thorsten Holtkämper. Real-time gaseous phenomena: a phenomenological approach to interactive smoke and steam. In *Proceedings of the 2nd International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, AFRIGRAPH '03, pages 25–30, New York, NY, USA, 2003. Association for Computing Machinery.

[56] Jens Krueger and Ruediger Westermann. Gpu simulation and rendering of volumetric effects for computer games and virtual environments. *Computer Graphics Forum*, 2005.

[57] Wito Engelke, Kai Lawonn, Bernhard Preim, and Ingrid Hotz. Autonomous particles for interactive flow visualization. *Computer Graphics Forum*, 38(1):248–259, August 2018.

[58] Niels H. L. C. De Hoon, Kai Lawonn, Andrei C. Jalba, Elmar Eisemann, and Anna Vilanova. Inkvis: A high-particle-count approach for visualization of phase-contrast magnetic resonance imaging data. *Eurographics Workshop on Visual Computing for Biology and Medicine*, 2019.

[59] J.J. van Wijk. Implicit stream surfaces. In *Proceedings Visualization '93*, pages 245–252, 1993.

[60] G. Scheuermann, T. Bobach, H. Hagen, K. Mahrous, B. Hamann, K.I. Joy, and W. Kollmann. A tetrahedra-based stream surface algorithm. In *Proceedings Visualization, 2001. VIS '01.*, pages 151–553, 2001.

[61] Christoph Garth, Xavier Tricoche, Tobias Salzbrunn, Tom Bobach, and Gerik Scheuermann. Surface techniques for vortex visualization. In Oliver Deussen, Charles Hansen, Daniel Keim, and Dietmar Saupe, editors, *Eurographics / IEEE VGTC Symposium on Visualization*. The Eurographics Association, 2004.

[62] Tobias Schafhitzel, Eduardo Tejada, Daniel Weiskopf, and Thomas Ertl. Point-based stream surfaces and path surfaces. In *Proceedings of Graphics Interface 2007*, GI '07, pages 289–296, New York, NY, USA, 2007. Association for Computing Machinery.

[63] Kai Bürger, Florian Ferstl, Holger Theisel, and Rüdiger Westermann. Interactive streak surface visualization on the gpu. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1259–1266, 2009.

[64] Florian Ferstl, Kai Burger, Holger Theisel, and Rudiger Westermann. Interactive separating streak surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1569–1577, 2010.

[65] Marwin Schindler, Aleksandr Amirkhanov, and Renata Raidou. Smoke surfaces of 4d biological dynamical systems. In T. Höllt and D. Jönsson, editors, *VCBM 2023: Eurographics Workshop on Visual Computing for Biology and Medicine*, page 5. The Eurographics Association, September 2023.

[66] Joshua Schpok, Joseph Simons, David S. Ebert, and Charles Hansen. A real-time cloud modeling, rendering, and animation system. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 160–166, Goslar, DEU, 2003. Eurographics Association.

[67] Keenan Crane, Ignacio Llamas, and Sarah Tariq. Real-time simulation and rendering of 3d fluids. *GPU gems*, 3(1), 2007.

[68] Kun Zhou, Zhong Ren, Stephen Lin, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Real-time smoke rendering using compensated ray marching. *ACM Transactions on Graphics (TOG)*, 27(3):1–12, 2008.

[69] A. Yenpure, S. Sane, R. Binyahib, D. Pugmire, C. Garth, and H. Childs. State-of-the-art report on optimizing particle advection performance. *Computer Graphics Forum*, 42(3):517–537, June 2023.

[70] Will Schroeder, Kenneth M. Martin, and William E. Lorensen. *The visualization toolkit (2nd ed.): an object-oriented approach to 3D graphics*. Prentice-Hall, Inc., USA, 1998.

[71] Jane Wilhelms and Allen Van Gelder. Octrees for faster isosurface generation. *ACM Trans. Graph.*, 11(3):201–227, July 1992.

[72] Nathan Andrysco and Xavier Tricoche. Matrix trees. *Computer Graphics Forum*, 29(3):963–972, 2010.

[73] Christoph Garth and Kenneth I. Joy. Fast, memory-efficient cell location in unstructured grids for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1541–1550, 2010.

[74] Michael Bußler, Tobias Rick, Andreas Kelle-Emden, Bernd Hentschel, and Torsten Kuhlen. Interactive particle tracing in time-varying tetrahedral grids. In Torsten Kuhlen, Renato Pajarola, and Kun Zhou, editors, *Eurographics Symposium on Parallel Graphics and Visualization*. The Eurographics Association, 2011.

[75] Marc Schirski, Christian Bischof, and Torsten Kuhlen. Interactive particle tracing on tetrahedral grids using the gpu. In *Proceedings of Vision, Modeling, and Visualization (VMV)*, pages 153–160, 2006.

[76] Ayachit Utkarsh. *The ParaView Guide: A Parallel Visualization Application*. Kitware Inc., 2015. ISBN: 978-1930934306.

[77] R J Carroll and H L Falsetti. Retrograde coronary artery flow in aortic valve disease. *Circulation*, 54(3):494–499, 1976.

[78] Nicholas S. Burris, Himanshu J. Patel, and Michael D. Hope. Retrograde flow in the false lumen: Marker of a false lumen under stress? *The Journal of Thoracic and Cardiovascular Surgery*, 157(2):488–491, February 2019.

[79] The Khronos® Group Inc J. Vulkan — cross platform 3D graphics, 2022.

[80] David Kinnear, Mark Atherton, Michael Collins, Jason Dokhan, and Tassos Karayiannis. Colour in visualisation for computational fluid dynamics. *Optics & Laser Technology*, 38(4–6):286–291, June 2006.

[81] Maarten H. Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Depth-dependent halos: Illustrative rendering of dense line data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1299–1306, November 2009.

[82] C. Bichlmeier, S.M. Heining, M. Feuerstein, and N. Navab. The virtual mirror: A new interaction paradigm for augmented reality environments. *IEEE Transactions on Medical Imaging*, 28(9):1498–1510, September 2009.