Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

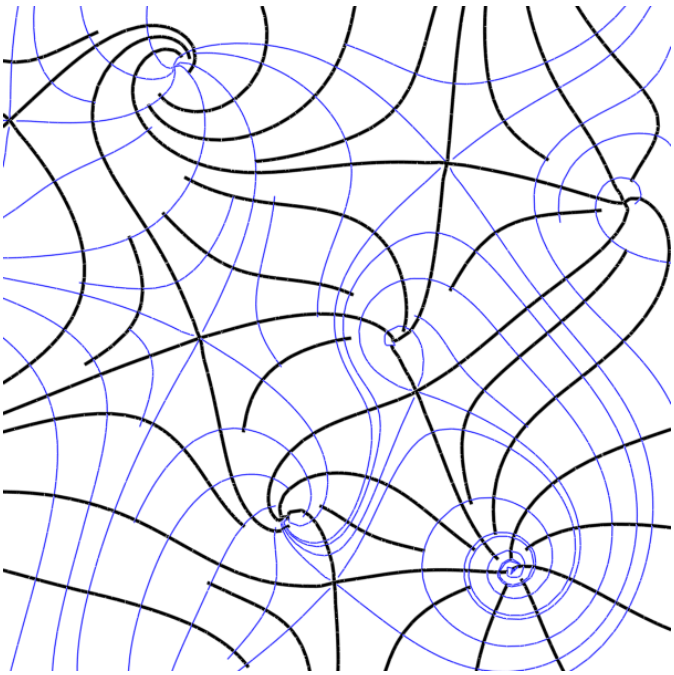
OLUFEMI ROSANWO, CHRISTOPH PETZ, STEFFEN PROHASKA,
HANS-CHRISTIAN HEGE, INGRID HOTZ

Dual Streamline Seeding - Method and Implementation

short version accepted for publication in Proc. IEEE Pacific Visualization Symposium 2009

ZIB-Report 08-49 (December 2008)

Dual Streamline Seeding - Method and Implementation



O. Rosanwo, I. Hotz, C. Petz, S. Prohaska, H.-C. Hege
Zuse Institute Berlin, Germany

December 15, 2008

Abstract

This work introduces a novel streamline seeding technique based on dual streamlines that are orthogonal to the vector field, instead of tangential. The greedy algorithm presented here produces a net of orthogonal streamlines that is iteratively refined resulting in good domain coverage and a high degree of continuity and uniformity. The algorithm is easy to implement and efficient, and it naturally extends to curved surfaces.

In dieser Arbeit wird eine neue Strategie zur Platzierung von Stromlinien vorgestellt. Hierzu werden zusätzliche duale Stromlinien verwendet, die –im Gegensatz zur üblichen Definition– orthogonal zum Vektorfeld verlaufen. Der vorgestellte Greedy-Algorithmus berechnet ein Netz aus orthogonalen Stromlinien, welches iterativ verfeinert wird, was zu einer guten Abdeckung der Domäne und einer gleichmäßigen Verteilung der Stromlinien führt. Es handelt sich um einen einfach zu implementierenden und effizienten Algorithmus, der direkt auf gekrümmten Oberflächen anwendbar ist.

Contents

List of Figures

1. Introduction

CFD-based flow visualization and analysis is nowadays a standard technique in many different application areas from technical engineering to life science research. For visual analysis of flow phenomena, among other aspects, tangent vector fields on 2D domains are especially important for at least two reasons. First, analyzing velocity fields in planar slice cuts of a flow domain is very common in practice, as these cuts provide a good overview and can be compared to experimentally observable Particle Image Velocimetry (PIV) measurements. Second, flow phenomena in near wall regions, i.e. close to fixed geometries that act as boundaries, are very important and mainly motivated our research. For instance, the tangent vector field of the shear stress at the curved boundary surface of a flow domain provides insight to important flow phenomena of viscous fluids like vortex formation and flow separation.

Several approaches for the visualization of tangent vector data on 2D surfaces exist, ranging from simple hedgehog plots to advanced feature extraction methods. Among them, streamline based methods are widely accepted for depicting flow at a glance. In contrast to texture based methods like line integral convolution (LIC), streamlines provide sparse visualizations that focus on significant structures and can thus be combined with other visualization techniques, or used to compare vector fields. Additionally, streamlines can be annotated with glyphs showing the flow direction or can be scaled, for example by the velocity magnitude.

The key problem of streamline based methods, which we will refer to as the *streamline placement problem*, consists of finding a good representative set of streamlines depicting the vector field visually comprehensibly and completely. To evaluate the quality of our results and compare them to other approaches, we refer to the following general criteria, proposed by Verma et al. [VKP00]:

Coverage: No important features of the vector field should be missed and the streamlines should cover the whole domain.

Uniformity: The distribution of streamlines should be more or less uniform across the domain.

Continuity: Long continuous streamlines are preferred over short ones.

1.1. Motivation

The presented visualization technique was developed in the context of research on the advance of patient-specific risk analysis for cerebral aneurysms. It is cooperatively worked on at ZIB, the Biofluid Mechanics Lab, Charité Berlin and the HELIOS Klinikum Berlin-Buch. Research is funded by the German Research Foundation.

Aneurysms are pathologically enlarged vessels. The main risk herein lies in rupture causing internal hemorrhage with high mortality rates. The main measurand to assess rupture risk is wall shear stress. Facilitating patient-specific risk analysis resolves into the following subtasks:

1. Reconstruct the wall geometry of an existing aneurysm from medical imaging (CT, MRI, 3DRA).
2. Conduct CFD-based simulation of flow behavior inside the reconstructed boundary geometry.
3. Extract data on wall shear stress and other relevant near-wall-flow characteristics from the simulation results.
4. Asses rupture risk and rupture location.

Although this technique is not fully matured, for different imaging and reconstruction techniques have significant effects on the simulation result, it is a promising approach towards patient-specific risk-analysis and treatment planning.

To gain an understanding of the structure of shear stress on an aneurysm, visualization of the numerical simulation results is crucial. Streamline-based visualization is especially useful here, since it allows simultaneous display of several flow characteristics and comparison of different flow fields. A prominent example in this context is simultaneous display of wall shear stress and pressure distribution (see Fig. 6.5).

1.2. Contributions

Our main contribution resides in the introduction of duality to the streamline placement problem. We define a dual vector field that is orthogonal to the original one along with the respective dual streamlines. Based on that, we present a primal-dual greedy algorithm that simultaneously solves the streamline placement problem for both fields. Since seeds for new streamlines are only placed along other streamlines, the search space for seed placement is reduced to a set of curves, which allows efficient placement on planar and curved surfaces alike. The primal-dual nature of our algorithm enables us to fully saturate the domain with streamlines, approximate and optimize streamline density distribution and infer a simple, yet effective streamline termination strategy.

Summarized, Dual Streamline Seeding comes with the following advantages:

- The algorithm has only a single parameter. It controls streamline density and is thus indispensable.
- The approach naturally extends to curved surfaces without the need of surface parametrization.
- All distances used are based on arc length. Thus, we avoid explicit computation of geodesic distances.
- Use of topology highlights important features, even at low density.

2. Basics

This section introduces the important concepts used in the following chapters. It is by no means exhaustive, but merely lays the foundation for understanding the rest of this report. Details can be found in the referenced works. If the reader is already familiar with vector field analysis and streamlines, this chapter may be skipped.

2.1. Vector fields

Formally, an n-dimensional steady vector field $\mathbf{V}$ is defined as a function $\mathbf{V} : D \to \mathbb{R}^n, D \subseteq \mathbb{R}^n$. Thus, it maps n-dimensional points to n-dimensional vectors. Many natural phenomena like velocity/direction of particle motions or physical forces can be modelled in this fashion. In this report, we focus on 3-dimensional *tangent vector fields*. In this special case, the domain of $\mathbf{V}$ is restricted to a two-dimensional surface embedded in $\mathbb{R}^3$ and the evaluation of the field at an arbitrary point of the domain yields a vector tangential to that surface.

In the following, we will assume that the surface is parametrizable, i.e. each point on it can unambiguously be described by a two-dimensional vector. In this case, the vector field can be expressed relative to this parametrization and is thus two-dimensional as well.

$$\mathbf{V} : D \to \mathbb{R}^2, D \subseteq \mathbb{R}^2 \tag{2.1}$$

Differentiating $\mathbf{V}$ yields the $2x2$ *Jacobi matrix* J. Let $\mathbf{V}(u,v) = (V_u, V_v)^T$, then

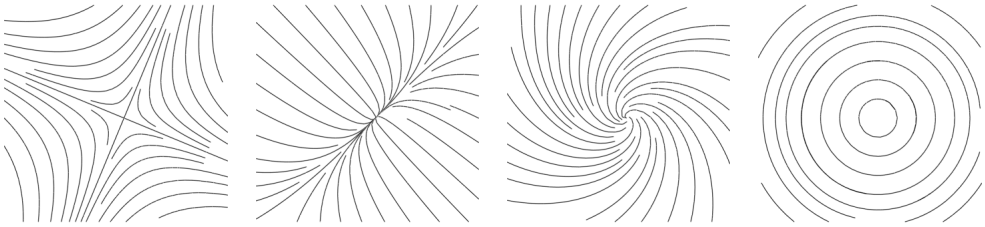
$$J(u,v) = \begin{pmatrix} \frac{\partial V_u}{\partial u} & \frac{\partial V_u}{\partial v} \\ \frac{\partial V_v}{\partial v} & \frac{\partial V_v}{\partial v} \end{pmatrix}. \tag{2.2}$$

The properties of this matrix are used for topology analysis, as described in Sec. 2.3.

2.2. Streamlines

A *streamline* – or tangent curve – is a mapping $l : I \to D, I \subseteq \mathbb{R}$, such that the curve is tangential to $\mathbf{V}$ for each $t \in I$. Since this direction is unambiguous, each point inside the domain where $\mathbf{V} \neq \vec{0}$ is passed by exactly one streamline. Streamlines do neither intersect, nor meet at any point in the field, but a streamline may form a closed path. More figuratively, a streamline describes the course of an imaginary massless particle through the field. They are intuitively comprehensible visualizations, for they bear resemblance to the way flows are traditionally depicted in drawings.

In general, streamlines cannot be described by a closed analytic expression but are given implicitly as a system of ordinary differential equations. A streamline $l(t) = (u(t), v(t))^T$ of a



(a) Saddle: $R_1 < 0$, $R_2 > 0, I_1 = I_2 = 0$ (b) Node: $R_1 \neq 0$, $R_2 \neq 0, I_1 = I_2 = 0$ (c) Spiral: $R_1 = R_2 \neq 0, I_1 = -I_2 \neq 0$ (d) Center: $R_1 = R_2 \neq 0, I_1 = -I_2 \neq 0$

Figure 2.1.: Types of 1st order critical points in a two-dimensional vector field. Let the complex eigenvalues of the Jacobian at the critical point be $R_1 + I_1 i$, $R_2 + I_2 i$ with $R_1 \leq R_2$. Nodes and spirals can further be subdivided into sources ($R_1, R_2 > 0$) and sinks ($R_1, R_2 < 0$).

vector field $\mathbf{V}(u,v) = (V_u(u,v), V_v(u,v))^T$ is thus described by

$$\left.\begin{array}{rcl} \frac{du}{dt} & = & V_u(u,v) \\ \frac{dv}{dt} & = & V_v(u,v) \end{array}\right\} \frac{dl}{dt} = \mathbf{V}(l(t)) \tag{2.3}$$

with initial condition $l(0) = x_0 \in D$, where x_0 is called the *seed point* of l.

The solution to the system of equations 2.3 is obtained by formal integration. This bidirectional integration is called *tracing* and is usually done via numerical methods like Runge-Kutta-Integration. Still, under certain premises, it is possible to solve the system of equations 2.3 directly, as will be discussed in Sec. 5. Both methods result in a series of sample points, whose direct connection (a polyline) yields an approximated streamline.

2.3. Vector field topology

The idea of topology is to segment a vector field into areas of equivalent streamline behavior, i.e. two streamlines seeded in the same area will converge to the same point in forward, as well as in backward tracing direction [HH89]. In the following, we will neglect boundary phenomena.

Key to this segmentation are points inside a vector field, where the field evaluates to $\vec{0}$. Streamline behavior in the vicinity of these *critical points* can be classified according to the eigenvalues of the Jacobi Matrix at these points. Fig. 2.1 illustrates the four resulting classes for two-dimensional linear vector fields.

Streamlines emerge from saddle points in the direction of the eigenvectors (and their inverse) of the Jacobian at the saddle point. These *separatrices* segment the vector field into areas of equivalent flow behavior. The set of all critical points, closed streamlines and separatrices is called the *topology* of the vector field. Fig. 2.2 exemplifies this for a simple analytic vector field. The topology represents the basic structure, therefore, each transformation or visualization of a vector field strives for preservation of the topology.

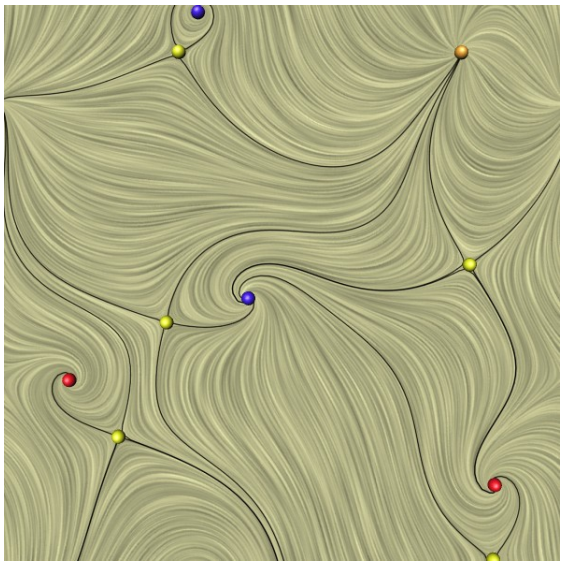


Figure 2.2.: Topology of a vector field. The field is visualized via LIC, separatrices are drawn as black lines and critical points are depicted by colored spheres, where color indicates the type of critical point.

Color scheme:

yellow	brown	red	blue
saddle	node(source)	spiral(source)	spiral(sink)

3. Related Work

Good overviews of flow visualization techniques in general can be found in Laramee et al. [LHD+04] and Post et al. [PVH+03]. In the following, we focus on streamline based flow visualization on 2D surfaces. A number of techniques with different objectives have been developed. We present them grouped into four categories: image based, direct, feature based and vector field dependent. Our contribution is closely related to the direct approaches.

Image based approaches define an energy function for the streamlines in image space and search for a minimum. Turk and Banks [TB96] use a low-pass filtered image of the streamlines to measure the deviation from a predefined saturation level across the image. Starting from an initial streamline set, the energy is minimized by iteratively lengthening, shortening, deleting, or moving streamlines in a global optimization process. Due to the large number of possible operations in each step, the method is computationally expensive with exponential running time. Still, visual appeal, uniformity and continuity of the streamlines is superior and serves as the benchmark for our method. Mao et al. [MHHI98] extend this approach to parametrizable curved surfaces in 3D. For the computation of the local filter size, they make use of the deformation tensor of the parameter space mapping. Schlemmer et al. [SHH+07] introduce an image based approach where the streamline density is steered by a user defined density function. They use a greedy algorithm where new streamlines are added one by one.

Direct methods place new streamlines with a certain heuristic that is expected to provide good overall seeding results without computing a global energy function. Jobard and Lefer [JL97] achieve good streamline uniformity, but streamline lengths, and thus continuity is unsatisfactory. They start new streamlines at a separation distance apart from all previously computed streamlines until the domain is fully covered. The technique is extended to unsteady flows in [JL00], its application for multiresolution flow visualization is described in [JL01] The method proposed by Liu et al. [LIG06] improves continuity by prioritizing streamline elongation over new streamline insertion. The seeding strategy presented by Mebarki et al. [MAD05] is closest to our approach. In a greedy algorithm, they start new streamlines in the center of the biggest remaining voids, and achieve good continuity and uniformity of the streamlines. Similarly, we search for the largest uncovered areas approximately on dual streamlines and start new streamlines at the midpoint of those segments. Mebarki et al. make heavy use of Delaunay triangulation, which makes their approach hard to apply on curved surfaces in 3D.

Verma et al. [VKP00] presented a feature based approach. By using fixed templates of equidistant seeds in the vicinity of the critical points of the vector field, they guarantee good coverage of the vector field topology. The template size is determined by a Voronoi decomposition of the domain. Remaining voids are filled by a Poisson random seeding.

The last category, vector field dependent methods, focuses on coverage of important features by defining similarity measures for streamlines and adding new streamlines at places with great streamline differences. Chen et al. [CCK07] propose a metric based on the distance, shape

and orientation of nearby streamlines, Li et al. [LHS08] propose a metric based on a distance transformation of the streamlines.

Most of these seeding techniques are primarily developed for planar 2D domains. Their application on curved surfaces embedded in 3D space is often not obvious, as distance and similarity measures are harder to compute. We propose an algorithm that does not rely on planarity of the domain. Thus, we form a bridge between methods for planar 2D domains and intrinsic 3D methods, as suggested by Salzbrunn and Scheuermann [SS08], Mattausch et al. [MTHG03], Ye et al. [YKP05], and Li and Shen [LS07].

4. Dual Streamline Seeding

Our algorithm takes as input a surface embedded in 3D, a tangent vector field $v \in \mathbb{R}^3$ and a surface normal field $n \in \mathbb{R}^3$. The two fields are often – but not necessarily – given as a discrete set of tangent data vectors, a set of surface normals and an associated interpolation scheme, such that v as well as n can be evaluated at each point on the surface. Furthermore, user-supplied separation distance δ defines the minimal distance from a seed to any existing streamline and thus controls the resulting streamline density. Our algorithm outputs a set of streamlines, which satisfy the quality criteria listed in Sect. 1.

For solving the streamline placement problem, we manage two types of streamlines simultaneously:

Primal Streamlines are curves inside the domain of v that are tangential to v at every point. These streamlines form our result.

Dual Streamlines are curves inside the domain of v that are tangential to the dual field $R(v) := v \times n$, i.e. orthogonal to v at every point. They are auxiliary streamlines, i.e. they are not visible in the resulting image.

Note, that repetition of the transformation R yields a vector field that is the inverse of the original vector field: $R(R(v)) = -v$. Since streamlines are bidirectional, this inverted field is equivalent to the original field in its streamline depiction. This motivates the use of the term duality in this work.

Intersections between primal and dual streamlines subdivide the involved streamlines into consecutive segments. These segments are key to our algorithm. We denote the set of primal streamline segments as P and the set of dual segments as D. Each set is represented by a priority queue ordered by the arc length of the segments. In the following, the term streamline refers to primal and dual streamlines alike.

Given suitable starting sets for P, D (see Sect. 4.1), our greedy seeding algorithm works as follows: In each iteration, the longest remaining segment is detected and a new bidirectional streamline is started from its midpoint in orthogonal direction. Fig. 4.1 illustrates this process. The algorithm terminates when no segment longer than 2δ remains.

Considering both sets of streamlines at the same time and keeping track of mutual intersections yields a local measurement for streamline density. Since all seeds are placed on streamlines, we approximate overall density distribution by the density distribution of intersections along the respective dual streamlines. We postulate:

> *Long streamline segments without intersection indicate voids in the dual streamline set.*

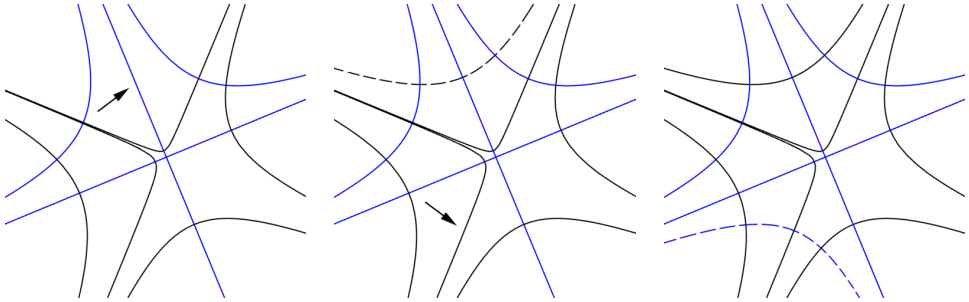


Figure 4.1.: Outline of algorithm behavior. Left: largest segment belongs to blue set (arrow), a new streamline for the black set will be started at its midpoint. Middle: new streamline has been inserted (shown as dotted line). Now, black set contains the largest segment (arrow) whose midpoint will again serve as seed point for the next streamline. Right: result of next streamline insertion. Newly inserted streamline is drawn dotted.

This is the central justification for our approach, because it bears a directive for seed placement. The next seed for a streamline in P should be placed at the center of the longest remaining segment in D and vice versa. This procedure enables us to efficiently find voids based on a one-dimensional distance metric (arc length) instead of Euclidean distance in image space or geodesic distance. This makes our approach very suitable for curved surfaces, where other measures are hard to apply correctly or are computationally expensive. Sect. 6 will demonstrate that this local density approximation leads to evenly spaced streamlines throughout the domain.

4.1. Initialization

To start our iterative seeding process, we need a first set of streamlines as initialization. This starting configuration should guarantee accessibility of all points inside the domain through a series of iterations, as described in Sect. 4.2, irrespective of the vector field structure. Furthermore, initialization must not leave large voids inside the domain. Although this criterion is not crucial for the success of the algorithm, a good starting configuration significantly reduces processing time. A structure that matches all these requirements is vector field topology.

We thus compute an initial set of dual streamline segments D by extracting the topological skeleton of $R(v)$, see e.g. [HH89]. This gives a reasonable starting coverage of the domain and good coverage of all critical points. Fig. 4.2 shows the result of initialization on an artificial vector field. Beware, that uniformity is not essential in the initialization procedure, since non-uniformities in the initial dual streamline set will not effect uniformity of the resulting primal streamlines.

In the rare case that this initialization fails to produce any starting curves, i.e. if v possesses no topology, we initialize D by tracing a small number of randomly seeded dual streamlines. Actually, a single dual streamline would suffice at this point, but an unfavorable seed location would slow down algorithm convergence. For all examined cases, the use of five of these streamlines is sufficient. Fig. 4.3 gives an example for the results of this alternative initialization.

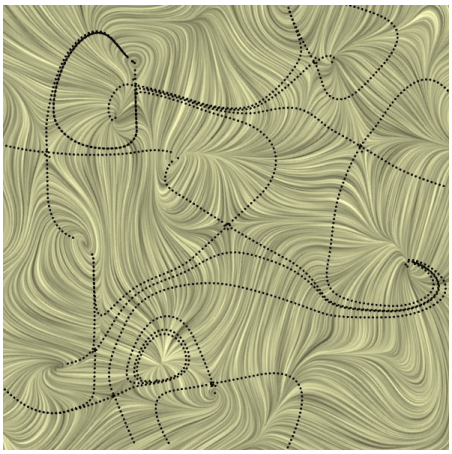


Figure 4.2.: Initial set of dual streamlines, i.e. dual topology shown as dotted lines. The vector field is shown via LIC. Although the initial density distribution is non-uniform, a good starting coverage is achieved and uniformity of primal streamlines is not affected.

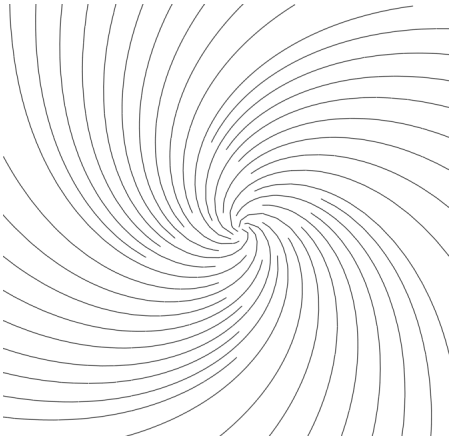


Figure 4.3.: Result of randomly seeded initial dual streamlines. Uniformity of the result does not depend on a uniform distribution of initial dual streamlines.

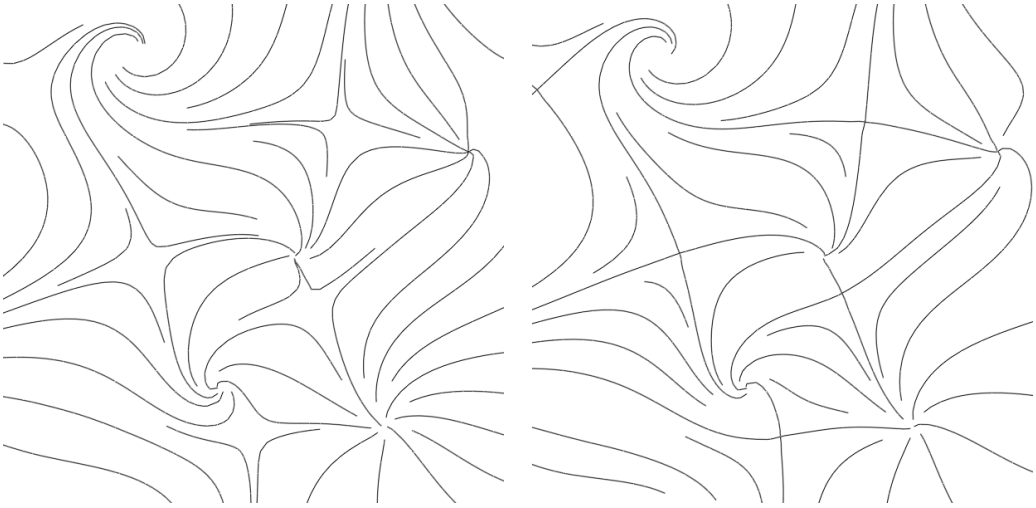


Figure 4.4.: Effects of different initializations. Left: P empty, Right: P initialized with topology of the vector field. The latter option provides clearly visible separatrices and guarantees conservation of perceptibility of topological structure at low densities.

After D has been initialized, P is initialized analogously with the topological skeleton of v, which ensures clearly visible separatrices especially at very low densities. Note, that initializing P includes intersection tests and streamline termination, as described in the following sections. If the emphasis of topological structure is not desired, P can be initialized empty. Fig. 4.4 compares these two alternatives.

4.2. Iteration

Each iteration consists of two major parts:

1. Extract the longest segment from P or D. The midpoint of this segment will serve as the next seed.

2. Trace a new streamline in forward and backward direction and test for intersections.

For illustration, let $s \in D$ be the longest remaining segment. Now, starting from the midpoint of s, a new primal streamline is traced in v. Intersections with segments in D split them into smaller segments at the intersection point. The currently traced streamline is also divided into segments at those intersection points. The newly traced segments are added to P.

If $s \in P$, the process is analogous, resulting in new streamline segments for D tested for intersection against all segments in P.

Avoiding clutter

Each streamline-based approach faces the problem of clutter, i.e. streamlines cluster at certain points or along certain lines in the field. These clusters are distracting if they do not coincide

with topological features of the field and should thus be avoided.

Similar to the seed criteria, we use the arc length of intersected streamline segments as termination condition for tracing. Each time an intersection between the currently traced streamline and another segment is encountered, the minimum of the two resulting segments' length d is determined. If d falls below a specified separation distance δ_{term}, tracing is stopped. δ_{term} is set relative to δ, a value of $\frac{\delta}{2}$ worked well for all our examples.

Note, that d is measured in arc length. Thus, contrary to standard Euclidean distance metric or similarity distance [CCK07], it is directly applicable on curved surfaces.

Besides, this streamline termination strategy yields a lower bound on the segment length. A segment of length $\leq \delta$ will terminate any intersecting streamline. It is also irrelevant for seed placement, since only segments of length $\geq 2\delta$ provide seeds. Thus, these segments do not need to be further subdivided at intersection.

4.3. Termination

The algorithm terminates, if no voids are left in either of the two streamline images. According to our main postulate, this corresponds to an upper bound on the length of all remaining streamline segments. This upper bound is set to 2δ and controls the density of the resulting streamline image. More formally, the termination condition can be formulated as:

$$\forall p \in P : ||p|| < 2\delta \wedge \forall d \in D : ||d|| < 2\delta.$$

5. Implementation

Our implementation is written in C++ as a module for the visualization framework AMIRA [SWH05], developed at the Zuse Institute Berlin. Our data setting comprises of a triangulated surface with 3D data vectors given at the vertices of the surface and normals given for each triangle. To construct a tangent vector field, data vectors are projected to the respective triangle plane. Inside a triangle, linear interpolation of the data vectors is employed.

There are three key aspects to an efficient implementation of Dual Seeding.

1. Streamline tracing
2. Representation of streamline segmentation
3. Intersection tests between segments

Sec. 5.1-5.3 will describe how our solutions to these individual problems are intertwined, followed by an analysis of their efficiency in Sec. 5.4 and concluded by a discussion on remaining issues in Sec. 5.5.

5.1. Tracing

We represent streamlines as polylines, where forward tracing extends a polyline at the end and backward tracing extends it at front. Streamline tracing is based on local exact tracing techniques described by Kipfer et al. [KRG03] and Nielson/Jung [NJ99]. This approach is based on solving the analytic form of the streamline curve equation inside a triangle to exactly calculate the exit point from the triangle. Thus, each integration step results in a line segment with endpoints on the boundaries of a triangle or a critical point inside the triangle. The approximation of a streamline through a polyline can be improved by calculating multiple vertices –and thus multiple line segments– along the course of a streamline inside a triangle. The necessity of this refinement depends on the average triangle size. In all examples presented in this report, the direct connection of entry and exit points of a triangle results in a satisfying streamline approximation. The decomposition of streamline segments into line segments that do not cross triangle boundaries is essential for our implementation of intersection tests.

During tracing, we assign each new sample point the arc length to the seed point. Tracing forward, a newly traced point is assigned the value of its predecessor plus euclidean distance between the two. Analogously, in backward direction, the newly traced point is assigned the value of its successor minus euclidean distance between the two. This way, the arc length of a segment between two points of a polyline can easily be calculated as the difference of these values.

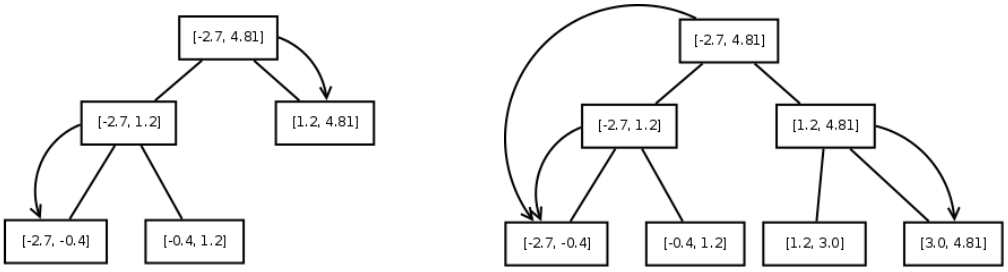


Figure 5.1.: Representation of streamline segmentation. Each node stores the interval of curve parameter values of the corresponding segment. Arrows indicate pointers to the largest contained subsegment at leaf level. Left: The streamline has been intersected at curve parameter values 1.2 and -0.4. Right: After another intersection at 3.0, another segment has become the largest one. Note, that an intersection affects only nodes on the path to the root.

5.2. Segment trees

Given this setting, all segments can be ordered according to their arc length. This could be done by inserting them into a priority queue, or by searching for the maximum on demand, but we chose a different approach. Splits of segments occur quite frequently, thus, when they include copying the segment vertices, this can soon become the bottleneck of the algorithm. Therefore, we retain the original streamlines and indicate splits by a binary tree structure (the segment tree), where a node represents a segment of the streamline. Each node keeps track of its largest subsegment at leaf level, implemented as a pointer to the respective tree node. In particular, the root node possesses a pointer to the largest overall segment of this streamline. Fig. 5.1 illustrates a simple example configuration. This data structure is easy to update and performs well in practice, although its efficiency depends on the presumption of fairly balanced trees. Sect. 5.4 discusses efficiency aspects of this data structure. With each streamline keeping track of its largest segment, global ordering of streamlines is done by a priority queue.

5.3. Intersection tests

To do intersection tests efficiently, we use an index structure that assigns to each triangle the line segments passing through it, i.e. each index entry consists of a pointer to a streamline and the index of the sample point entering the triangle. This way, we can efficiently determine which line segments may intersect a newly traced line segment by querying the index at the current triangle and only test line segments passing through the same triangle for intersection.

5.4. Performance of Data structures

One of the most time-critical operations is splitting a segment. It consists of a tree traversal with complexity $O(h)$ with h being the height of the segment tree, the actual split with complexity

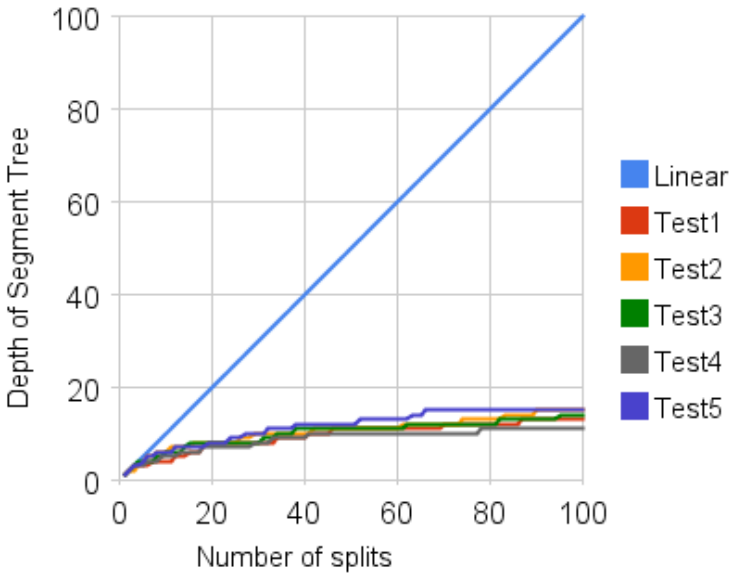


Figure 5.2.: Growth of depth of the segmentation tree in relation to number of splits. Splits were done at random locations. A logarithmic growth can be observed.

$O(1)$ and updating the pointers along the path to the root at a cost of $O(h)$. Thus, asymptotic worst-case running time is $O(n)$ in the number of splits n, irrespective of the number of streamline vertices. Still, practice shows that our data structure exhibits logarithmic growth both for real application as well as for random splits. Fig. 5.2 illustrates the results of five test series for the growth of depth under random splits.

Using an index to restrict the search for potentially intersecting streamlines is a common trade-off between memory consumption and running time. It allows us to detect all candidate line segments in $O(1)$ time at the cost of additional memory consumption of $O(\#triangles)$. Under the assumption that our algorithm produces evenly spaced streamlines, this index will also be evenly filled with its average load depending linearly on δ.

5.5. Known Implementation Issues

As of now, running time escalates for separation distances below approximately 0.5% of the domain width. Whether this is due to the exponentially rising number of intersections, which is inevitable, or mistakes in the implementation could not be fully resolved. Furthermore, in Sec. 6.3, we observe that running time grows exponentially in the number of streamline vertices. A linear growth should be feasible here. The source of this inefficiency is not resolved either.

In [KRG03], the authors propose to find the exit point of a triangle via Newton-iteration. This procedure has proven to be unstable for two reasons:

1. No guarantee of convergence can be given.

2. Even if the algorithm converges, there is no way to ensure that the detected exit point is the first time the curve exits from the triangle.

To face the first mentioned problem, a fallback mechanism has been implemented. We employ a simple adaptive sampling procedure to approximate the exit point. Yet, this technique exhibits slower convergence and equally suffers from the latter problem. In case of failure of both methods, tracing is aborted. This case occurs very rarely, but still should be addressed in further development work.

To extend the given discrete set of data vectors to a vector field, we employ triangle-wise linear interpolation projecting the given data vectors to the plane of the respective triangle. This interpolation scheme suffers from continuity problems. Along triangle boundaries, the interpolated values differ depending on the values of which of the two triangles are used for interpolation. This can cause anomalies in streamline tracing and critical point detection and thereby influences our implementation. The use of a more sophisticated interpolation scheme should be considered, if these problems occur frequently.

6. Results

In this section, we present the results of our technique and compare them – where applicable – to other seeding methods on artificial vector fields and CFD data. We implemented the method of Verma et al. [VKP00] and use an implementation of Mebarki et al. [MAD05], which is freely available [Meb]. For comparison to the techniques of Turk and Banks [TB96] and Jobard and Lefer [JL97], we use pictures from [JL97] by courtesy of the authors. Where possible, we further compare running times.

6.1. Planar Domains

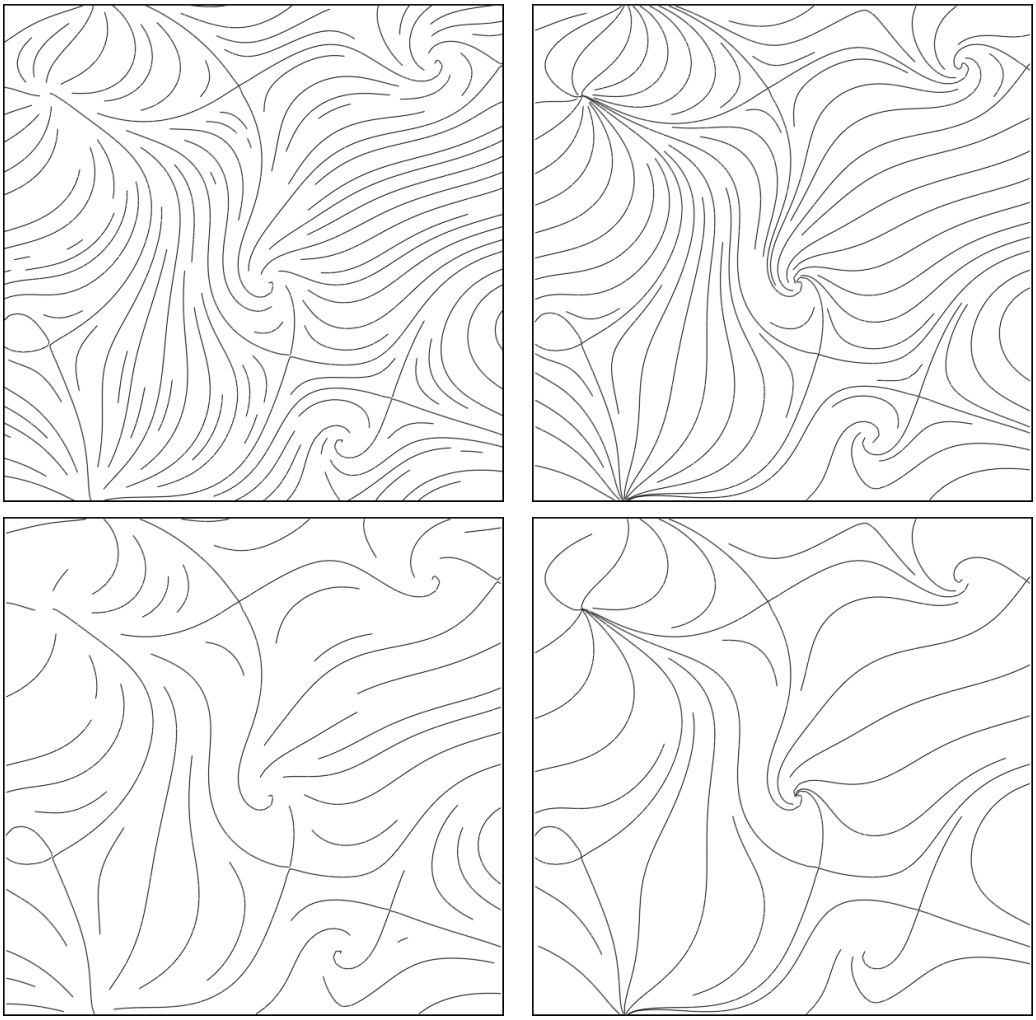
In the following, we compare Dual Streamline Seeding to four other streamline placing algorithms, representing three classes of seeding techniques:

- image-based optimization [TB96],
- direct seeding [JL97] and [MAD05], and
- feature-based seeding [VKP00].

As discussed in Sect. 1, quality criteria for streamline placement are uniformity, continuity, coverage. In the following examples, coverage of important flow features is given with all five methods, thus, we will focus on continuity and uniformity. Sect. 7 contains further discussion on coverage. Since there is no standard to compare continuity and uniformity quantitatively, we compare the results visually.

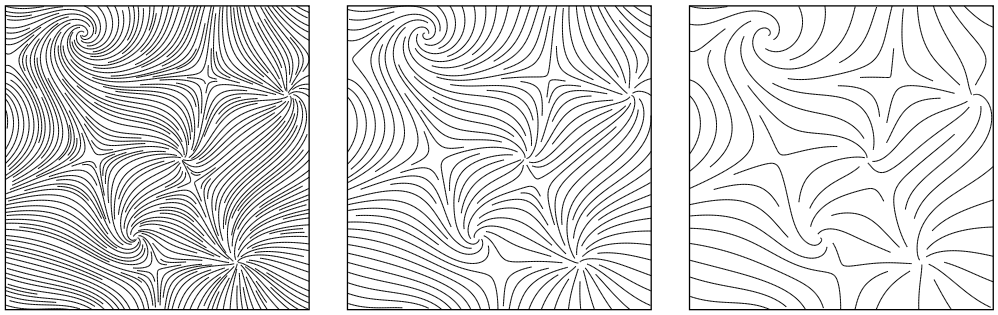
Fig. 6.1 compares Dual Seeding to the flow-guided seeding strategy of Verma et al. with additional Poisson-distributed seeds to fill the voids. Our algorithm gives better results in terms of continuity and uniformity, especially with high densities. Without additional seeds, the approach of Verma et al. lacks coverage for regions without critical points. On the other hand, the additional seeds further diminish continuity and uniformity.

Fig. 6.2 compares our technique to the listed image based and direct approaches. In terms of continuity, our algorithm gives comparable results to the Farthest Point technique and the method of Turk/Banks and better results than the algorithm of Jobard/Lefer. The latter tends to produce short, separated streamlines along the same path of the flow where the viewer would expect one continuous streamline as Fig. 6.3 demonstrates. This should be avoided to give an impression of flow coherency in a steady flow. Our algorithm does not enforce overall uniformity of streamline density as strictly as the other methods. Especially, clutter control is less regular than with Euclidean distance measure, which is employed by the other approaches. On the other hand, the resulting local deviation of density leads to a better accentuation of the critical points, as discussed in Sect. 7.

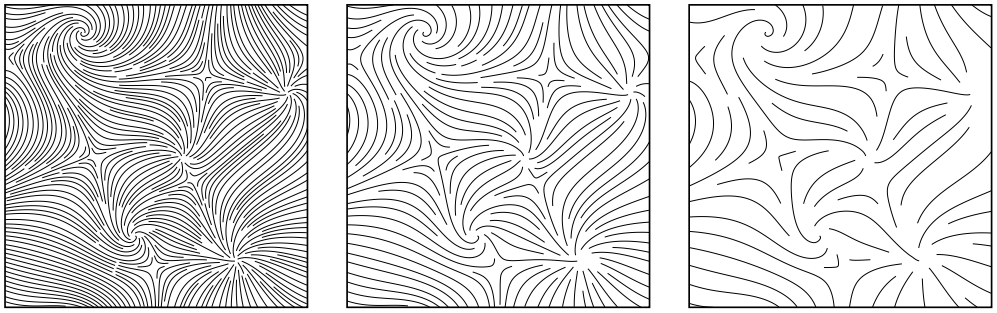


(a) Verma/Kao/Pang00 with additional seeds (b) Dual Seeding

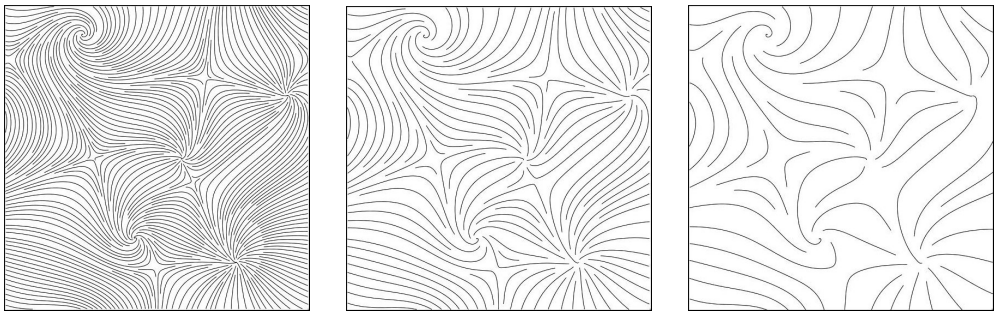
Figure 6.1.: Comparison to feature-based technique by Verma et al. Separation distances: top 2% bottom 3.5% of domain width. Poisson-distributed seeds employed by Verma et al. introduce significant discontinuities, whereas Dual Seeding results in long continuous streamlines, while still capturing topological features.



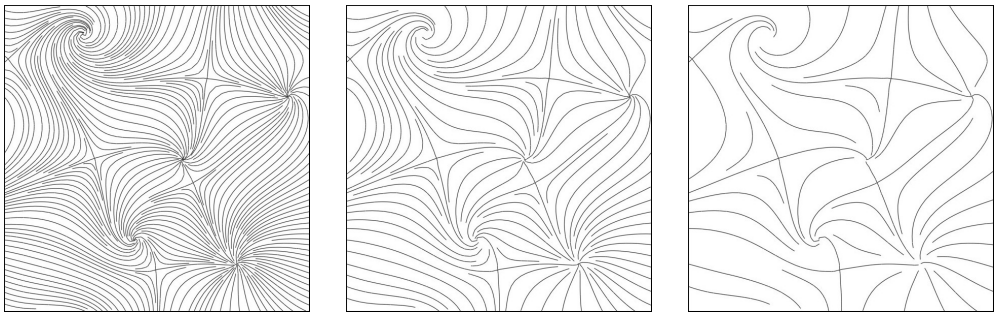
(a) Turk/Banks



(b) Jobard/Lefer



(c) Mebarki et al.



(d) Dual Seeding

Figure 6.2.: Comparison of streamline placement techniques with decreasing densities. Separation distance from left to right: 0.84%, 1.68%, 3.36% of domain width. Pictures on the first two rows were provided by the authors of [JL97].

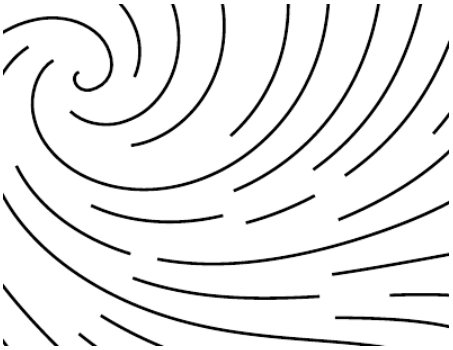


Figure 6.3.: Close-up of the result of Jobard/Lefer of Fig. 6.1. Cumulative discontinuities can be observed especially near critical points.

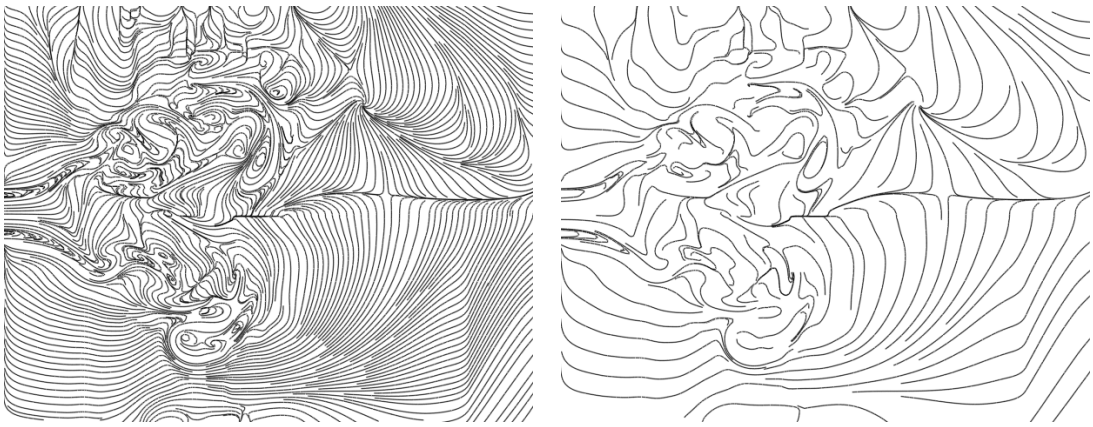


Figure 6.4.: Swirling jet entering fluid at rest. Depicted is a planar slice of a 3D velocity field. Separation distance is 0.5% (left) and 1.5% (right) of domain width. Areas of turbulent and laminar flow are well distinguishable at both densities.

Fig. 6.4 demonstrates results for different separation distances on a planar cut of a 3D vector field. It contains 12,524 data vectors and results from a simulation of a swirling jet entering fluid at rest. Despite the topological complexity of the data set, regions of laminar and turbulent flow can well be distinguished. Neither do the turbulent areas, which are densely covered by dual streamlines, exhibit cluttering, nor are there any voids in areas of laminar flow.

6.2. Non-Planar Domains

Fig. 6.5 combines a color-coded visualization of pressure with our streamline seeding result and extracted critical points of the wall shear stress. Both fields resulted from CFD simulation of blood flow in a reconstructed aneurysm. The streamlines exhibit the same degree of continuity, uniformity and coverage as the planar examples. Overall flow behavior as well as local topological details are well observable. The structure of the pressure field is still well perceptible due to

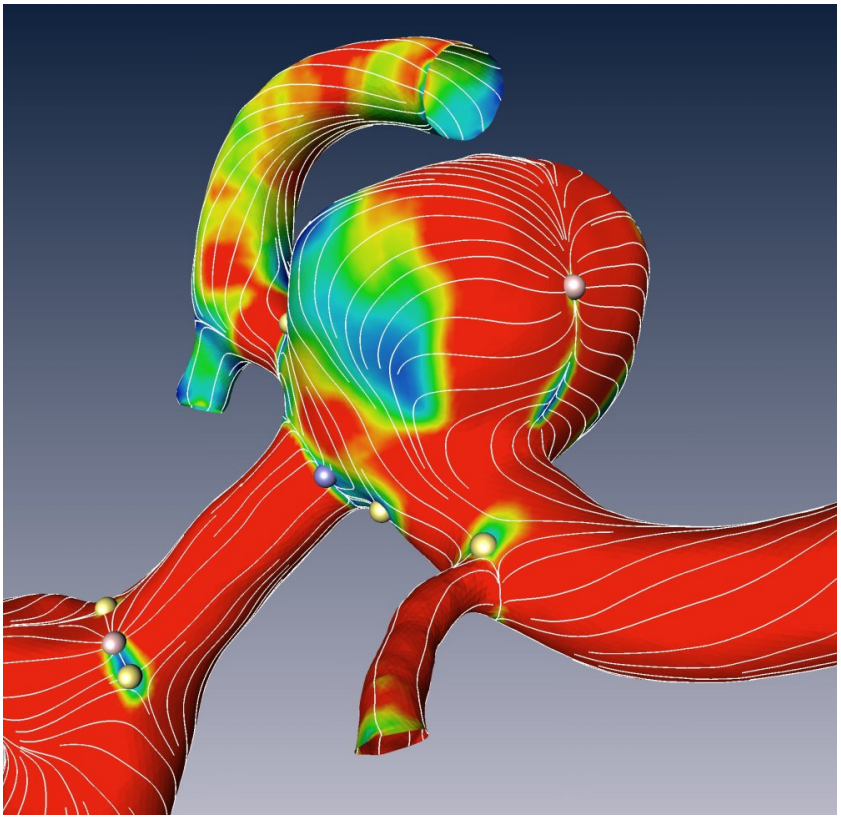


Figure 6.5.: Overlay of wall shear stress streamlines and color-coded pressure distribution on a cerebral aneurysm. Critical points are highlighted by small spheres.

the sparsity of the streamlines.

We applied our method in Fig. 6.6 for the validation of a CFD solution with experimental data. A transparent silicon model of a cerebral aneurysm was used for the experimental flow, particle traces of the near-wall-flow were recorded in a gray-scale image with the wall PIV technique by Kertzscher et al. [KBGA08]. The same geometry and flow condition was used for the CFD computation. The PIV image and streamlines of the wall shear stress of the CFD solution are superimposed. Due to the sparsity of the streamline visualization, similarities and differences of the two flow fields are well observable.

Another visualization of wall shear stress resulting from CFD simulation on a curved surface is shown in Fig. 6.7. Flow separation can be observed at the back side of the propeller blades. Although the field barely possesses topological features as starting points, the resulting streamline image still exhibits complete coverage and a great degree of uniformity and continuity.

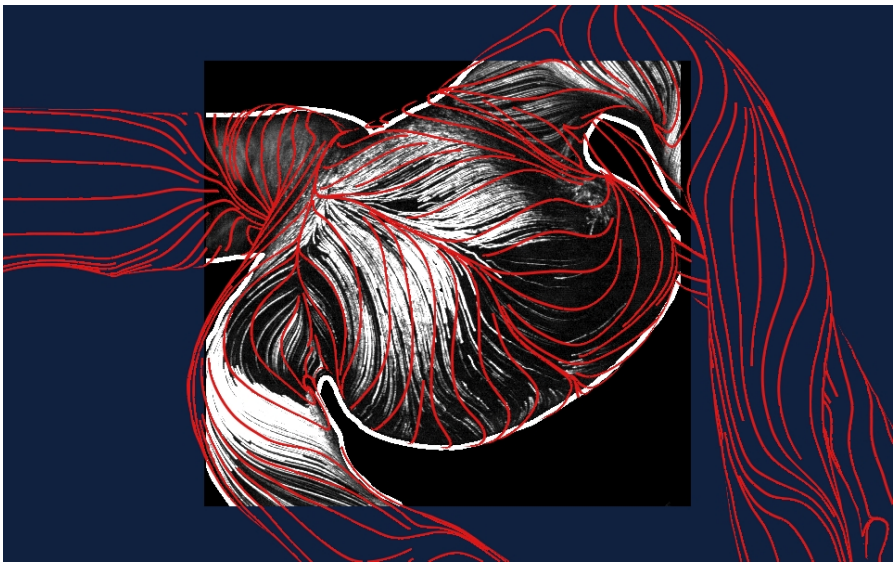


Figure 6.6.: Comparison of experimental flow and CFD results of a cerebral aneurysm. Particle traces of the near wall flow are shown in the background [KBGA08], the wall-shear-stress vector field of a roughly aligned CFD model is depicted with red streamlines. The comparison shows a good agreement of the fields.

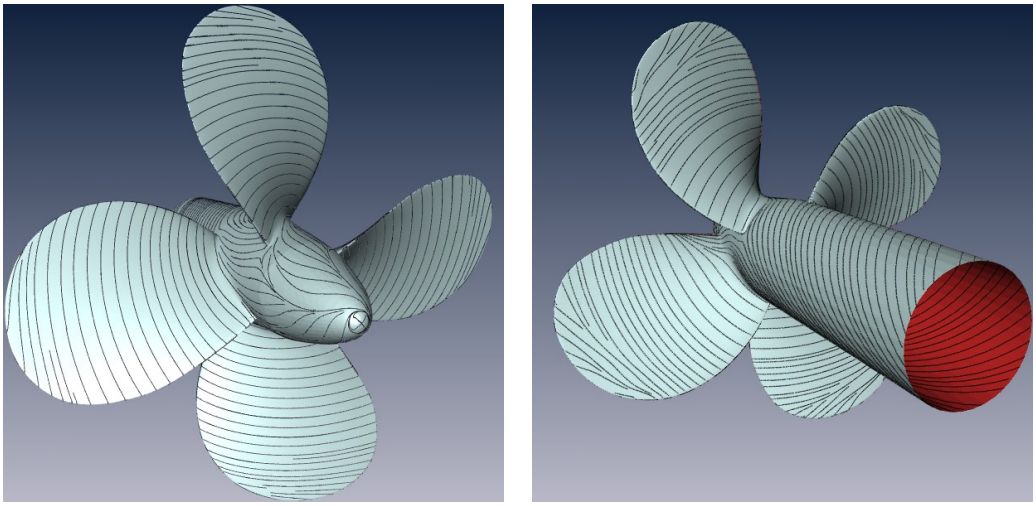


Figure 6.7.: Streamlines of the wall shear stress on a ship propeller. Left: Pressure side, Right: Suction side. Flow separation can be observed at the tip of the blades at the suction side.

Separation distance	Verma et al. [VKP00]	Mebarki et al. [MAD05]	Dual Seeding
0.5%	587ms	578ms	154ms
1%	182ms	250ms	71ms
2%	58ms	125ms	39ms
5%	28ms	46ms	21ms

Table 6.1.: Comparison of running time including initialization. For the examined case, our algorithm is faster by a factor of 2-3.

6.3. Timings and Scalability

Timings were measured on a standard AMD64 3GHz computer. We analyze running times in three ways:

- comparison for fixed densities and a given vector field,
- growth of running time with increasing streamline density,
- growth of running time with increasing complexity of the input data.

Table 6.1 compares running times of our method to [MAD05] and [VKP00]. The three techniques were applied to the vector field depicted in Fig. 6.2, which consists of 50x50 data vectors. Our algorithm is faster than the methods proposed in [MAD05] and [VKP00] by a factor of 2-3.

In a second step, we applied the three mentioned methods to the same vector field with increasing streamline densities. Dual Seeding exhibits a significantly slower growth than the other two approaches, as Fig. 6.8 illustrates.

We further adapted scalability measure proposed by [MAD05], where running time is analyzed relative to the number of streamline vertices. For this, we created several versions of an analytic vector field at different sampling densities. Fig. 6.9 illustrates the employed vector field and the results. An exponential growth of running time is observable.

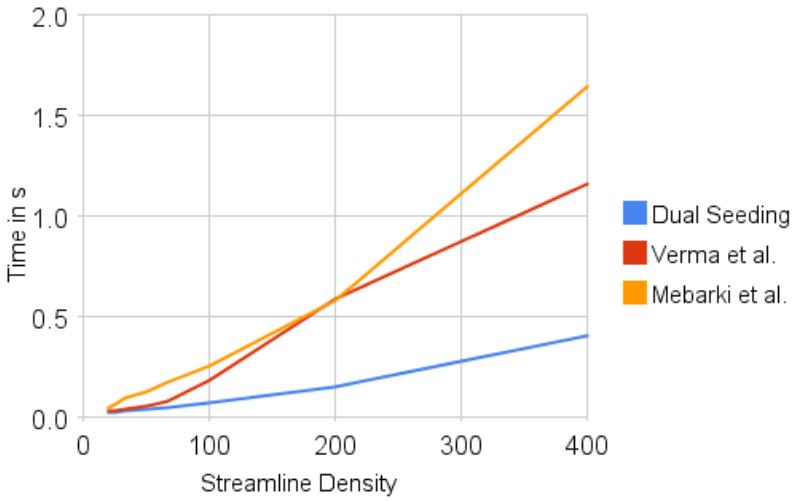
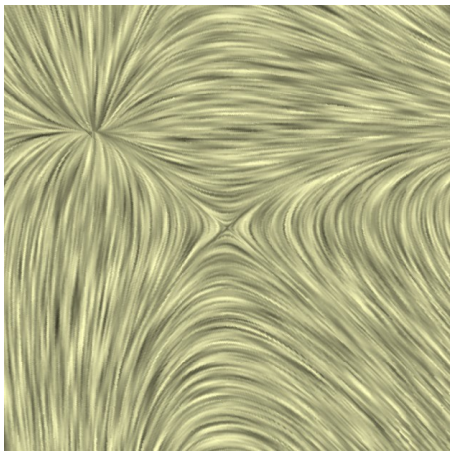
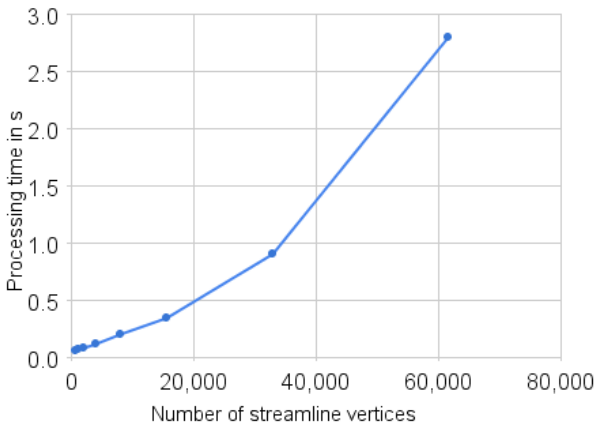


Figure 6.8.: Running times in relation to streamline density, which is the inverse of the separation distance. Dual Seeding exhibits a more moderate growth than the methods proposed in [MAD05] and [VKP00].



(a) Analytical field used for scalability measurement. It is described by: $v(x,y) = \begin{pmatrix} x^2+10xy+7\sin(y) \\ 5x+8xy^3+10\sin(xy) \end{pmatrix}$



(b) Results of scalability measurement. An exponential growth in the number of streamline vertices is observable.

Figure 6.9.: Running time growth with increasing number of streamline vertices.

7. Discussion

This chapter discusses features, limitations and the development history of Dual Seeding.

7.1. Streamline termination and loop detection

Our streamline distance measurement is induced by the respective dual streamline set, and is thus iteratively refined. This particularly impacts streamline termination, allowing early streamlines to gather, which they usually do at critical points, while streamlines added later on are terminated at farther distances. Thereby, our algorithm prioritizes feature coverage over uniformity. In effect, critical points are more accentuated at low streamline density, as Fig. 7.1 demonstrates. This accentuation improves the perceptibility of critical points compared to methods, whose streamline termination strategy is based on Euclidean distance, like [MAD05], [JL97], [VKP00] and [TB96]. Apart from this desirable deviation of density, some undesirable clutter that does not provide additional information remains, as can be ovserved in Fig. 6.4 (left). Since streamlines can only be terminated at intersection points, where inter-streamline distance is again approximated, clutter is significantly reduced but not fully eliminated.

To ensure effective streamline termination, including loop detection, it is important to initialize D first. During the initialization of D, P is empty, and thus, the streamline termination cri-

(a) Mebarki at al. (b) Dual Seeding

Figure 7.1.: Comparison at low streamline density. Our streamline termination algorithm is less strict, allowing streamlines to gather at critical points, which improves their perceptibility.

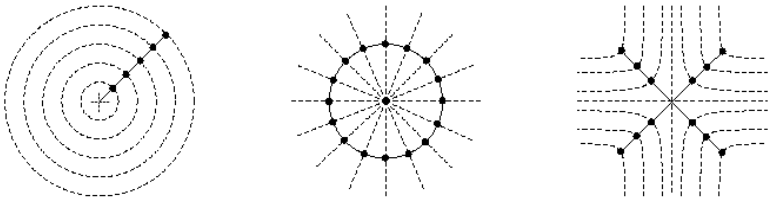


Figure 7.2.: Seeding templates proposed by Verma et al.

terion never applies, resulting in very long lines that may clutter in the vicinity of critical points or loop, as can be seen in Fig. 4.2. Therefore, the number of tracing steps has to be limited at this stage. On the other hand, loops in primal streamlines are detected, as they would intersect the same dual streamline twice. Thus, this initialization order ensures, that after initialization, tracing always stops gracefully without the need for an artificial step limit.

7.2. Extensibility to 3D

Our method is especially tailored for seeding on arbitrarily shaped curved 2D surfaces and cannot directly be extended to a 3D seeding technique. We rely on the symmetry of the streamline duality. In 2D, the dual (or orthogonal) structure of a line is again a line and we can apply the same method for the addition of subsequent new streamlines. In 3D, the dual structure of a 1D streamline is a 2D surface; the symmetry of our method would be lost. Apart from the structural difference, it is not desirable to seed streamlines in 3D with the same objectives as in 2D, as mutual occlusion is the main problem to solve in 3D.

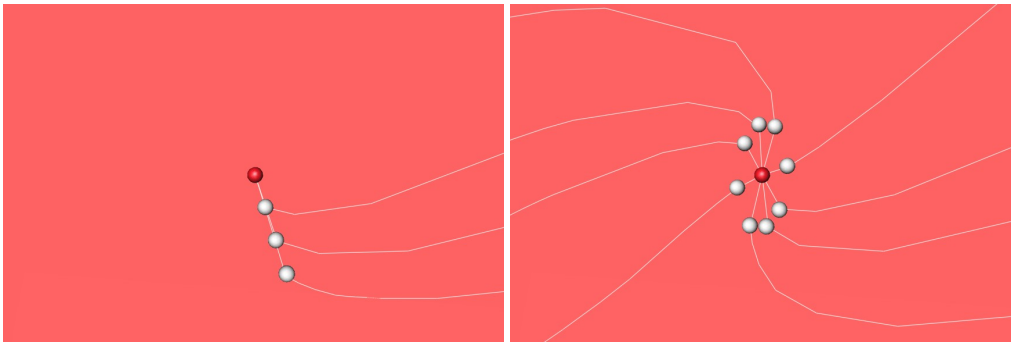
7.3. Emergence of the algorithm

By construction, our seed locations on the dual streamlines have similarities to the seeding templates of Verma et al. [VKP00] shown in Fig. 7.2. For each critical point, the seeding templates run orthogonal to idealized (linear) flow fields, extending to the boundaries of a Voronoi cell centered around the critical point. In fact, our algorithm emerged from this principal idea. This section will discuss the adaptions to [VKP00] that finally resulted in Dual Seeding.

A direct application of [VKP00] on curved surface domains exhibits several drawbacks.

- Projecting the planar seeding templates to a curved surface distorts their carefully chosen shape.
- Static templates do not adequately reflect local flow behavior.
- Coverage of spirals is unsatisfactory.

Thus, we first generalize the static templates to dynamic dual streamlines, running orthogonal to the vector field. This extends the symmetrical templates and departs from the assumption of planarity. Fig. 7.3 compares this dynamic seeding strategy to the static template.



(a) Static seeding template (b) Orthogonal seeding

Figure 7.3.: Comparison of static vs. dynamic seeding strategy. The single spiral in the middle of the field (red sphere) is solely depicted by streamlines, whose seeds are marked by gray spheres. Static seeding along a straight line, as proposed by Mebarki et al., often fails to capture spirals thoroughly. Placing seeds with regard to the vector field results in better coverage.

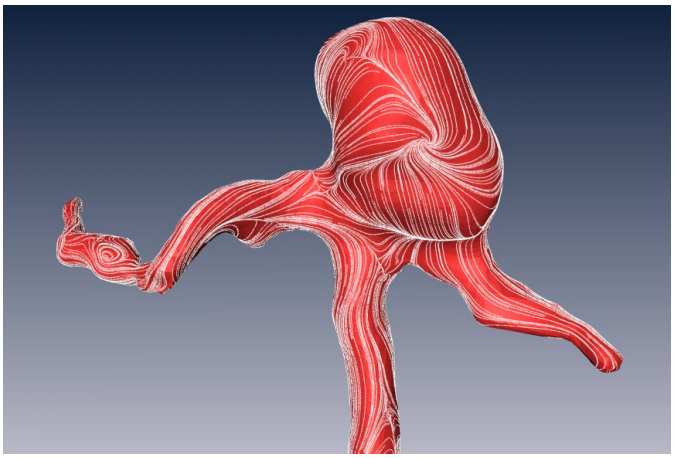


Figure 7.4.: Good domain coverage can be achieved by seeding along the dual topology of the vector field. Yet, equidistant seeding leads to a non-uniform streamline distribution.

We further drop the explicit handling of each critical point. Instead, four dual streamlines are started at the bisectors of each saddle point. Placing seeds only on these topology-guided streamlines yields good domain coverage if they are not restricted to a single Voronoi cell. Still, the regular seeding along these curves –as inherited from Verma's method– results in a non-uniform streamline distribution, as can be observed in Fig. 7.4. It further leads to exponential running time behavior due to the quickly increasing number of seed points.

To limit the count of possible seed locations that have to be tracked, we employ a greedy approach, where only the midpoints of streamline segments can serve as seeds. This way, the complexity depends on the number of segments, instead of the number of streamline vertices.

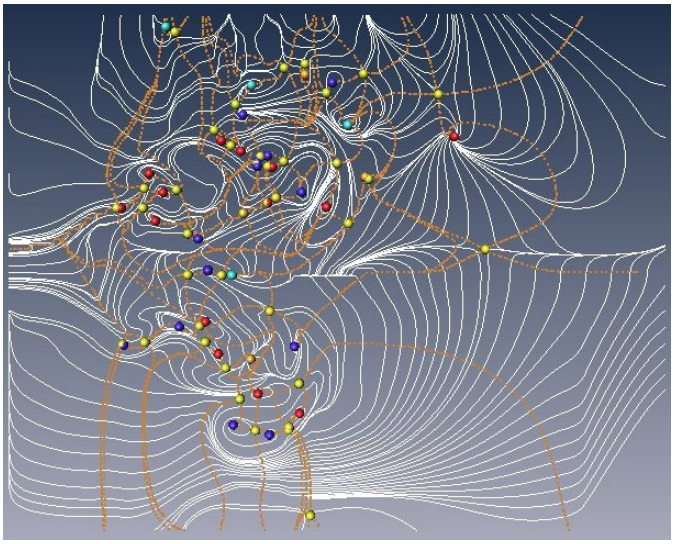


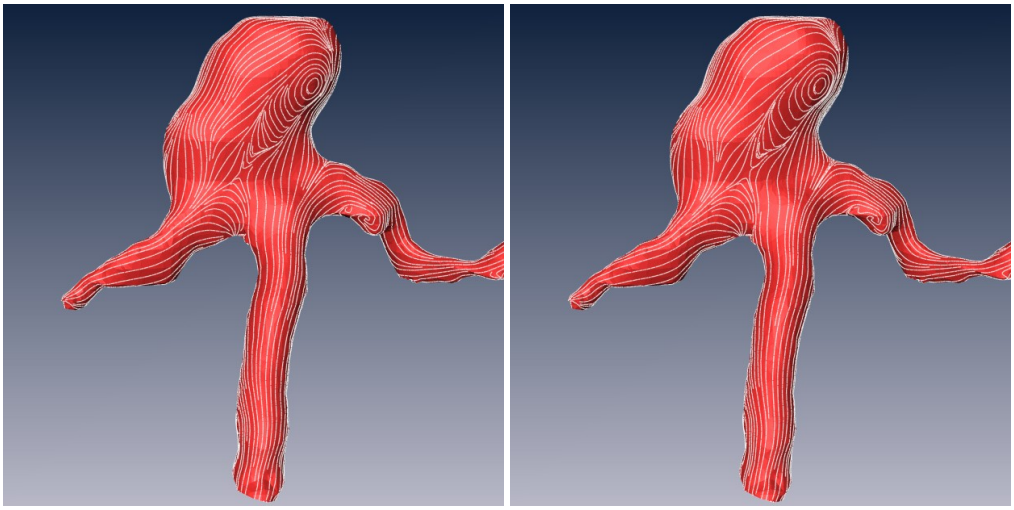
Figure 7.5.: Dual Streamline Seeding with a fixed (initial) set of dual streamlines. Critical points are depicted with colored spheres, primal streamlines are shown as white lines, dual streamlines are shown via brown dotted lines. Although coverage of the main flow features is satisfactory, clutter avoidance and coverage in areas with a low count of dual streamlines is not.

The introduction of segments and the need to keep track of segment intersections also allowed to realize streamline termination as part of intersection testing. Although this preliminary algorithm already yields good results, as Fig. 7.5 demonstrates, coverage and streamline termination is unsatisfactory in areas that are sparsely covered by dual streamlines.

So, we finally meet this problem by allowing dual streamlines to be added in the course of the algorithm by the same mechanism primal streamlines are seeded. Thereby, voids in the dual field are filled as well, improving uniformity and coverage of the streamline result.

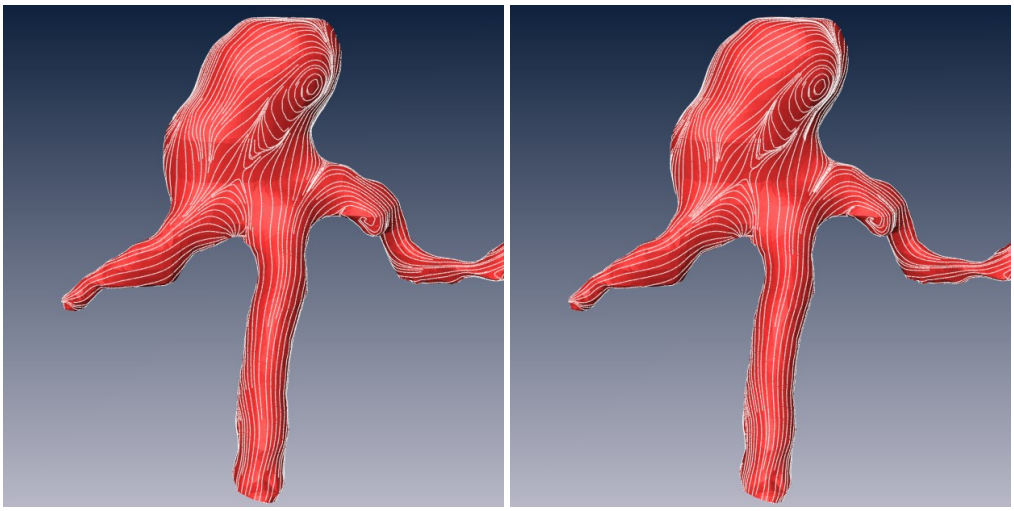
7.4. Discarded Optimizations

Concluding this chapter, we discuss two optimizations that were tried but abandoned. Our method applies the same separation distance δ to primal and dual streamlines. We experimented with the use of two separation distances: δ_p bounds the length of primal segments steering density of dual streamlines and δ_d bounds the length of dual segments steering density of primal streamlines. This affects seed selection, streamline termination and algorithm termination. Setting $\delta_d < \delta_p$ (primal streamlines denser than dual streamlines) does not generally decrease running time, as expected. The resulting lower count of dual streamlines does decrease effectiveness of streamline termination though. Setting $\delta_d > \delta_p$ (dual streamlines denser than primal streamlines) results in a more uniform streamline termination, but with an immediate running time penalty. Fig. 7.6 summarizes the results of one test series. Since no general improvement could be observed, only a single parameter is used.



(a) $\delta_p = 0.5\delta_d$, processing time:1.24s

(b) $\delta_p = \delta_d$, processing time:0.96s



(c) $\delta_p = 2\delta_d$, processing time:0.95s

(d) $\delta_p = 4\delta_d$, processing time:1.02s

Figure 7.6.: Effects of separate δs for the two segment sets. Setting the separation distance for the dual streamlines higher, decreases effectiveness of streamline termination without generally improving running time.

We also experimented with alternative termination criteria. Instead of setting an upper bound to the longest segment, one could also bound the average segment length or the length of the median. Yet again, we could not observe a general improvement of streamline uniformity or running time, therefore, we chose the most simple option.

8. Conclusion and Future work

We have presented a novel seeding technique for tangent vector fields on curved surface domains. Unlike other approaches, our method is not build upon Euclidean distance, Delaunay-triangulation, Voronoi-decomposition, or other concepts, which are hard to apply efficiently on curved surfaces.

Beside means of streamline tracing, only simple algorithmic concepts like priority queues and binary trees are used, which makes our method easy to implement. At the same time, it is highly efficient, comparable in placement quality and superior in processing time to state-of-the-art methods.

The main contribution of this work resides in the introduction of dual streamlines. By analyzing and selectively refining the resulting net of orthogonal streamlines we explore the domain and control streamline density efficiently.

This approach yields a new measurement for density distribution, i.e. long streamline segments indicate cavities in the dual field. As further work, instead of complying to the greedy paradigm, one could also use that measurement as a basis to apply optimization approaches, such as the one employed by Turk and Banks. This could improve overall uniformity at the cost of simplicity and efficiency.

We also presented main features of our implementation. Although it is efficient in a wide range of densities and domain complexities, it still exhibits inefficiencies at very high densities or very complex input data. The source of these drawbacks still has to be resolved.

A. Acknowledgments

This work was supported be the German Research Foundation (DFG Grant HE 2948/5-1 and Emmy-Noether Research group) and the European Commission FP6 Project VIRTUE.

Thanks to L. Goubergrits and J. Pöthke, Biofluid Mechanics Lab, Charité Berlin and A. Spuler, HELIOS Klinikum Berlin-Buch for the aneurysm dataset and fruitful discussions. We further thank B. Jobard, Laboratoire d'Informatique de l'Université de Pau et des Pays de l'Adour for providing some of the comparative vector field figures and W. Kollmann, University of California Davis for the provision of the swirling jet dataset.

The INSEAN E779A propeller geometry in Fig. 6.7 has been provided by INSEAN, CFD simulation has been performed by HSVA.

Bibliography

[CCK07] Yuan Chen, Jonathan D. Cohen, and Julian H. Krolik. Similarity-guided streamline placement with error evaluation. In *IEEE Transactions on Visualization and Computer Graphics*, volume 13, pages 1448–1455, 2007.

[HH89] J. Helman and L. Hesselink. Representation and display of vector field topology in fluid flow data sets. *Computer*, 22(8):27–36, Aug 1989.

[JL97] Bruno Jobard and Wilfrid Lefer. Creating evenly-spaced streamlines of arbitrary density. In W. Lefer and M. Grave, editors, *Visualization in Scientific Computing '97. Proceedings of the Eurographics Workshop in Boulogne-sur-Mer, France*, pages 43–56, Wien, New York, 1997. Springer Verlag.

[JL00] Bruno Jobard and Wilfrid Lefer. Unsteady flow visualization by animating evenly-spaced streamlines. *Comput. Graph. Forum*, 19(3), 2000.

[JL01] Bruno Jobard and Wilfrid Lefer. Multiresolution flow visualization. In *WSCG (Posters)*, pages 34–35, 2001.

[KBGA08] U. Kertzscher, A. Berthe, L. Goubergrits, and K. Affeld. Particle image velocimetry of a flow at a vaulted wall. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, 222(4):465–473, 2008.

[KRG03] Peter Kipfer, Frank Reck, and Günther Greiner. Local exact particle tracing on unstructured grids. *Computer Graphics Forum*, 22:133–142, 2003.

[LHD+04] Robert S. Laramee, Helwig Hauser, Helmut Doleisch, Benjamin Vrolijk, Frits H. Post, and Daniel Weiskopf. The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum*, 23(2):203–221, 2004.

[LHS08] Liya Li, Hsien-His Hsieh, and Han-Wei Shen. Illustrative streamline placement and visualization. In *Proc. of IEEE Pacific Visualization*, pages 79–86, 2008.

[LIG06] Zhanping Liu, Robert J. Moorhead II, and Joe Groner. An advanced evenly-spaced streamline placement algorithm. In *IEEE Transactions on Visualization and Computer Graphics*, volume 12, pages 965–972, 2006.

[LS07] Liya Li and Han-Wei Shen. Image-based streamline generation and rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):630–640, 2007.

[MAD05] Abdelkrim Mebarki, Pierre Alliez, and Olivier Devillers. Farthest point seeding for efficient placement of streamlines. In *Proc. of IEEE Visualization*, pages 479–486, 2005.

[Meb] Abdelkrim Mebarki. http://www-sop.inria.fr/geometrica/team/Abdelkrim.Mebarki/Streamlines.html.

[MHHI98] Xiaoyang Mao, Yuji Hatanaka, Hidenori Higashida, and Atsumi Imamiya. Image-guided streamline placement on curvilinear grid surfaces. In *VIS '98: Proceedings of the conference on Visualization '98*, pages 135–142, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[MTHG03] Oliver Mattausch, Thomas Theußl, Helwig Hauser, and Eduard Gröller. Strategies for interactive exploration of 3d flow using evenly-spaced illuminated streamlines. In *SCCG '03: Proceedings of the 19th spring conference on Computer graphics*, pages 213–222, New York, NY, USA, 2003. ACM.

[NJ99] Nielson and Jung. Tools for computing tangent curves for linearly varying vectorfields over tetrahedral domains. *Transactions on Visualization and Computer Graphics*, 5:360–372, 1999.

[PVH+03] Frits H. Post, Benjamin Vrolijk, Helwig Hauser, Robert S. Laramee, and Helmut Doleisch. The state of the art in flow visualisation: Feature extraction and tracking. *Computer Graphics Forum*, 22(4):775–792, 2003.

[SHH+07] Michael Schlemmer, Ingrid Hotz, Bernd Hamann, Florian Morr, and Hans Hagen. Priority streamlines: A context-based visualization of flow fields. In Ken Museth, Torsten Möller, and Anders Ynnerman, editors, *Eurographics/ IEEE-VGTC Symposium on Visualization*, pages 227–234, 2007.

[SS08] Tobias Salzbrunn and Gerik Scheuermann. Flow structure based 3d streamline placement. In Hans-Christian Hege, Konrad Polthier, and Gerik Scheuermann, editors, *Topology-Based Methods in Visualization II*. Springer, to appear 2008.

[SWH05] Detlev Stalling, Malte Westerhoff, and Hans-Christian Hege. Amira: A highly interactive system for visual data analysis. In Charles D. Hansen and Christopher R. Johnson, editors, *The Visualization Handbook*, pages 749–767. Elsevier, 2005.

[TB96] Greg Turk and David Banks. Image-guided streamline placement. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 453–460, New York, NY, USA, 1996. ACM.

[VKP00] Vivek Verma, David Kao, and Alex Pang. A flow-guided streamline seeding strategy. In *Proc. of IEEE Visualization*, pages 163–170, 2000.

[YKP05] Ye, Kao, and Pang. Strategy for seeding 3d streamlines. *Visualization*, 5:60, 2005.