Strategies for Seed Placement and Streamline Selection

Sudhanshu Sane

Abstract—Flow visualization is a vital component in the workflow of studying computational fluid dynamic simulations. Integral curves or streamlines are one of the most commonly used techniques to visualize flow fields and selecting a good set of streamlines is viewed as a challenge. Identifying a representative set of streamlines that captures the flow behavior can be achieved by either strategically placing seed points or selecting a subset of precomputed streamlines that exhibit desired properties. Over the past two decades, as desired characteristics and hardware resources have evolved, the approaches to solving the challenge have greatly varied. Primarily automatic techniques, these algorithms can be classified into density-based, feature-based, and similarity-based methods. In this report, we further subdivide each category to identify the strategies adopted by various researchers. We identified a total of 18 automatic technique strategies and 2 manual technique strategies. In addition to the qualitative and quantitative comparisons established by these works, we evaluate the strategies based on criteria specifically relevant to us, i.e., redundancy, regions of interest, and computation. We believe our classification and categorization reporting benefits future research endeavors with regard to in situ flow visualization techniques where seed placement and streamline selection will be key to successful flow exploration.

Index Terms—Flow visualization, Integral curve, Seed placement, Streamline selection

1 Introduction

Flow visualization is a prominent branch of scientific visualization and plays a key role in the scientific exploration and understanding of fluid dynamics phenomena. Flow visualization techniques are applied to a variety of fields ranging from modeling and simulation of weather and climate systems to aerodynamics to turbomachinery design processes. These techniques are used to capture and highlight regions of interest to better understand the behavior of the flow field enabling scientists to then improve computational fluid dynamics models.

A flow field is often defined as a vector field over a discretized mesh in the domain. If the vector field does not evolve over time the flow field is considered to be in a steady state. If the vector field evolves over time it is said to be in an unsteady state. Streamlines are a popular technique used to visualize the behavior of a steady state field. Likewise, pathlines are used to visualize unsteady state flow. Both streamlines and pathlines are integral curves that represent the trajectory of a particle. These curves are calculated by first placing a seed point in the domain, followed by integrating the path of the trajectory by considering the underlying vector field at each point. The majority of research done in this field has been limited to steady state flow, given that unsteady state flow introduces additional challenges. For example, when considering a flow field defined over a 2D grid, streamlines traced in that domain can not intersect paths, while pathlines can occupy the same spatial location at different points in time.

For over two decades researchers have attempted different approaches to solving a key problem when using streamlines to visualize a vector field — the strategic selection of a sparse representative set of streamlines to visualize a vector field. While a dense collection of streamlines can capture all features in a field, the resultant visualization might be hard to perceive or regions in the field might be completely occluded. Conversely, too few streamlines might miss interesting features and not provide sufficient coverage. Additionally, what comprises a representative set of streamlines has varied greatly based on the dimensions of the domain, the application context, and objectives of the research. Further, objectives of research have evolved from identifying a uniform distribution of streamlines to focusing on highlighting flow features to being able to respond to streamline similarity queries. More recently, with the emergence of in situ processing, techniques to extract flow field information in the form of integral curves have been proposed.

The problem of selecting a representative set of streamlines can be approached from different directions. Selecting a representative set of streamlines has often been framed as a seed placement problem and is commonly perceived as tackling the same challenge. Intelligently placing seed points such that the generated streamlines capture interesting flow features and maintain uniformity or coverage has been a commonly adopted method. Other works have considered selectively choosing streamlines that convey the most information from a dense set of streamlines or clustering similar streamlines before choosing representatives for each cluster. Approaches to the problem have varied to a great degree with most recent efforts involving machine learning. In this report, we categorize and analyze the contributions of various seed point placement and streamline selection research. Given the primary focus for most research has been limited to steady state flow, i.e., the use of streamlines, we will specifically highlight when a particular work has targeted unsteady state flow.

Following the discussion of our motivation, we identify the challenges associated with seed placement and streamline selection strategies, evaluation/comparison considerations, and applicability of these strategies in situ. Next, a

classification of the strategies in the field is followed by the contributions of this report.

1.1 Motivation

Solutions to the problem of generating or selecting streamlines that accurately capture the behavior of the flow and highlight important regions are valuable given their use to study fluid phenomena. Our motivation to conduct this survey was to identify the different strategies that have been adopted and the specific goals they aimed to achieve.

Typically utilized post hoc, these methods have largely solved the problem of streamline selection for steady state fields. The application of these methods in situ can be to perform visualization as the simulation progresses or to sample the flow field strategically in order to save useful information. With regard to generating visualizations, challenges of maintaining frame coherence for unsteady state flow, operating under the in situ constraints of memory and compute resources, and operating in a distributed environment would require extensions to existing work. With regard to sampling flow data in order to maximize information per byte stored, this challenge has a direct relation to calculating a sparse set of streamlines that allow reconstructing the vector field with minimal error. Some works even adopt this approach when generating visualizations post hoc, i.e., the vector field should be well preserved or divergence in the field should be captured.

With more research being directed toward in situ visualization methods and extracting or sampling flow data in situ, knowledge of existing seed placement and streamlines selection algorithms is valuable. Our research specifically focuses on information extraction of unsteady flow data to tackle the large data visualization problem and thus we are motivated to conduct this survey which can directly inform our research efforts.

1.2 Challenges in Seed Placement and Streamline Selection

Seed placement and streamline selection strategies have been proposed to address various flow visualization tasks. The most common research objective is the generation of an informative visualization image or set of images. Other tasks that we survey which have involved seed placement and streamline selection (without being too specific) are the extraction of flow field information in the form of streamlines as opposed to a vector field defined over a mesh, extraction of all streamlines that match a streamline query, or particle-based systems that control particle density distribution.

With regard to generating an informative visualization image, if an excessively large number of streamlines is used, it can result in a dense and cluttered image. This is particularly problematic for three dimensional flow fields, given streamlines will occlude one another and the flow direction can become hard to observe or follow. If a very small number of streamlines is used, important flow features can be missed. Uniformly placing seed points in the domain to generate streamlines often results in visual artifacts if the generated streamlines are too short. If the generated streamlines are complete, i.e., only terminate at boundaries or critical points, streamlines might cluster in certain regions resulting in a non-uniform distribution.

When considering extraction of flow field information to later reconstruct a vector field, occlusion is not a concern, however, computing a dense set of streamlines can have undesired computational and storage costs. This once again introduces a tradeoff with the desired goal being calculating a representative set of streamlines that results in minimal reconstruction error. Matching streamlines based on a query introduces the challenge of identifying all similar streamlines irrespective of orientation and scale. Particle-based methods suggest new ways in which particle density distribution can be controlled without cluttering or occluding regions of interest.

1.2.1 Desired Characteristics of a Streamline Visualization

Verma et al. [1] were the first to explicitly list characteristics that are desired of the selected set of streamlines. These characteristics are —

- **Coverage:** Streamlines should not miss interesting regions of the flow field.
- **Uniformity:** Streamlines should be uniformly distributed over the field.
- **Continuity:** From an aesthetic perspective, streamlines should be selected such that they show continuity in the flow, i.e., long streamlines are preferred.

The desired properties of the selected set of streamlines have been modified by researchers and scientists as this area of research has evolved. One of the primary additions to the above list has been visibility of regions of interest, i.e., to handle occlusion when considering the view of a set of streamlines in 3D. Thus, when considering the selection of streamlines in 3D, it is important that less informative streamlines do not occlude more interesting regions of the flow. Other desired characteristics are smooth transitions or frame coherence when visualizing unsteady flow or changing viewpoints, retaining spatial perception for depth cues, and representing maximum information content per byte stored. Contributions to this field have prioritized different characteristics while advancing or improving on previous work - either from a visualization or computational perspective.

1.2.2 Evaluation

The evaluation of streamline placement algorithms has largely been qualitative, i.e., based on the qualities of the generated streamline visualization. However, qualitative evaluations can be biased based on the specific requirements or objectives of each algorithm. Thus, capabilities such as the ability to maintain spatial perception, or highlight multiple interesting regions in the field are viewed as ”upgrades“ or improvements. In this report, rendering techniques, such as thinning of lines or lighting effects, are not discussed and are deemed out of scope given the focus of this report is identifying representative and important streamlines for a flow field. Few works have used quantitative measures such as accuracy of the reconstructed vector field using streamlines. Research in the field has typically strived to improve the flow exploration abilities via these streamlines selection algorithms. Thus, in addition to qualitative and

quantitative comparisons made by the methods themselves, we further compare and relate the works based on criteria important to us for our research. We evaluate these methods on the basis of:

- Regions of Interest
- Redundancy
- Computation

Using these three axes enables us to determine what approaches should be adopted in the future based on objectives and constraints.

Information content represented via streamlines is directly reflected by the method used and the efforts of the algorithm to capture regions of interest and minimize redundancy. The computation costs for solving challenges associated with streamline selection that meets prescribed criteria are worth accounting for, given it can be viewed as an expensive search problem. Developing techniques which can solve this problem efficiently or in parallel is another challenge with the majority of works operating in serial. Besides improving the qualitative or quantitative measure of the sets of chosen streamlines, several works are motivated by the need to improve on the computational costs of previous approaches.

The use of streamlines or particle-based systems for interactive flow exploration requires the proposed algorithms to compute at real-time processing speeds. Interactive methods could involve varying the level-of-detail, isolating regions of the domain, changing the view direction, and so on. With flow visualization algorithms being researched to operate in situ, these algorithms must operate under additional constraints introduced by the simulation environment and ideally be able to make use of accelerators.

1.2.3 In Situ Methods

The majority of works that propose seed placement and streamline selection algorithms operate under the post hoc visualization paradigm. In recent years, there has been a shift in performing visualization and analysis efforts in situ. In situ processing, through its successful frameworks and usage [2], [3], [4], [5], [6], [7], [8], has been demonstrated to be an important approach for large data analysis and visualization on upcoming supercomputers. With regard to flow visualization, in situ processing counteracts issues of temporal sparsity which can cause inaccuracies for unsteady state flow visualizations. Flow visualization performed in situ for large scale simulations allows access to the complete spatial and temporal resolution of the simulation data. Exploring flow visualization and analysis under the in situ processing paradigm is a growing research field with frameworks for visualization [9] and efforts to extract information in forms other than vector field grids [10], [11] being developed.

The Lagrangian paradigm can be used in situ to extract a smart sampling of the flow and then used post hoc for flow exploration by reconstructing the field. Several research works have supported this effort to improve the extraction and interpolation to visualize unsteady state flow [10], [12], [13], [14], [15], [16], [17], [18]. Operating under the constraints of an in situ environment requires addressing the following challenges —

- **Distributed computation of integral curves:** Efficiently computing particle trajectories in a distributed environment is viewed as a difficult problem given the scalability issues and lack of control of data distribution.
- **Flow map sampling strategy:** Efficient sampling or seed placement for unsteady state flow fields by evaluating the domain for flow features or maintaining desired particle distribution.

1.3 Classification

Research presenting seed placement and streamline selection have evolved over the past two decades. Methods have ranged from manually selecting seed points to a variety of automated techniques including most recently applying machine learning to the problem of streamline selection. While a chronological presentation of these works would provide an understanding of how the field has evolved, we categorize these works based on the general approach they have adopted. We select this approach to classification since distinct sets of approaches have progressed simultaneously or with significant overlap.

We classify seed placement and streamline selection techniques at a high-level into —

- Automated Techniques
    1) Density Based
    2) Feature Based
    3) Similarity Based
- Manual Techniques

These high-level categorizations are further divided and subdivided for a total of 20 categories - 18 automatic strategies and 2 manual strategies. Each of the density, feature and similarity-based techniques have a total of 6 base level categories. Figure 1 illustrates the classification and categorization for strategies of automatic techniques for seed placement and streamline selection.

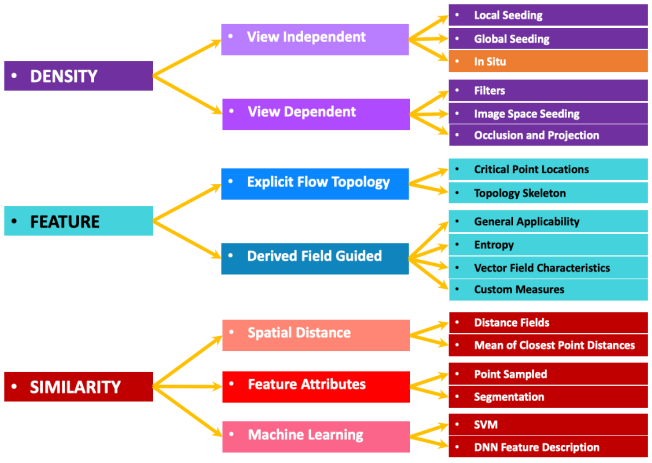

Fig. 1: Our classification and categorization for strategies of automatic techniques for seed placement and streamline selection.

The majority of research over the past two decades has proposed approaches for automatically generating sparse

sets of streamlines that attempt to convey flow field information accurately. These approaches encompass methods that build a sparse set of streamlines by choosing candidate seed locations to methods that select the most informative set of streamlines based on curve attributes and spatial location.

Density-based techniques primarily seek to evenly-space streamlines in object or image space. These approaches prioritize uniformity to cover the entire space without any particular focus on flow features (for example, a sink or source). Feature-based techniques adopt a focus + context approach prioritizing placement of streamlines that accurately capture flow features before placing additional streamlines that provide context information of the surrounding flow. Streamline similarity-based techniques approach the problem of streamline selection by first clustering similar streamlines based on distance metrics or streamline features and then choosing a streamline to represent each cluster.

Manual approaches are suitable when the flow field under consideration is being explored interactively in a virtual reality environment or when the phenomena in the vector field is very specific and known to the scientist. Flow field information, real-time rendering and visualization, and seed placement tools can be used to assist guiding seed placement in an interactive environment. However, this approach is not reliable when exploring complex unknown flows since important features can be missed if the generated streamlines from manually placed seeds do not capture this behavior.

1.4 Contribution

In this paper, we classify research contributed to the field of flow visualization proposing approaches for seed placement and streamline selection. We focus on the aspects of these works that contribute to domain coverage, feature focus, and extraction of information (in the form of an integral curve) that enables accurate flow field reconstruction. Our survey highlights techniques proposed to identify informative streamlines and intelligent placement of particles for coverage of regions of interest. We extend our seed point placement techniques survey to methods involving pathlines, flow maps sampling, and particle-based flow visualization methods. Additionally, we look at the use of streamline selection to respond to queries requiring identification of similar streamlines.

We do not focus on rendering/viewing aspects such as occlusion management implementation, thinning of streamlines, animation, viewpoint selection, etc. Further, for general feature extraction or vector field clustering techniques (often employed within seed placement and streamline selection workflows) we direct our reader to works by Post et al. [19], Laramee et al. [20], [21], McLoughlin et al. [22], and Pobitzer et al. [23]. We deem streamline selection methods that have been applied to domains other than flow visualization, for example, DTI fiber tracking, as outside the scope of this survey.

With this work, our contributions are as follows —

1) Categorize research to identify variation in approaches, application, benefits, shortcomings, and computational costs of techniques for streamline selection.
2) Identify methods that provide intelligent seed placement for calculating streamline and pathlines.
3) Provide a classification of research, which can act as a foundation for future in situ methods to assess the direction they want to adopt with regard to generating visualizations or extracting information.

1.5 Definitions

We first define a streamline and other relevant terms, followed by definitions of useful characteristics of streamlines.

A streamline is a line that is everywhere tangent to the instantaneous local velocity vector. A streamline represents the path followed by a massless particle moving with the flow. Streamlines are geometric objects used to visualize steady state flow, i.e., constant velocity field.

Pathlines, like streamlines, are integral curves and represent the path traced by a massless particle moving in an unsteady state flow field, i.e., velocity field evolves over time.

A critical point is a singularity in the vector field such that the velocity magnitude at that location is zero. Critical points can be classified by the eigenvalues of the Jacobian matrix at their position. Sources, sinks and saddle points are examples of critical points in a flow field.

The curvature of an integral curve is the absolute value of the rate of change of angle of inclination with respect to arc length. It measures how much the integral curve deviates from a straight line.

The torsion of an integral curve measures how sharply the curve is twisting or bending out of the plane of curvature or osculating plane.

The tortuosity of an integral curve is the ratio of the length of the curve compared to the shortest distance between its start and end points. It is a measure of deviation from the shortest path and quantifies how twisted the curve is.

Vorticity or curl of the velocity is a vector quantity that indicates the tendency of a fluid particle to rotate or circulate at a particular point.

Helicity is the absolute value of the dot product between velocity and vorticity, with high absolute values indicating vortex regions.

2 Automatic Techniques

Automatic seed placement or streamline selection algorithms follow a set of rules to generate a distribution/selection of streamlines (or particles in some cases). These algorithms may consider the view direction, properties of the vector field, properties of integrated streamlines and so on. Our classification identifies whether a particular algorithm is primarily a density-based, feature-based or similarity-based approach. We categorize automatic techniques based on the essence and primary motive of the algorithm. For example, an algorithm might first extract flow feature locations and strategically place seed points in these regions before placing additional seeds to generate an approximately uniform distribution of streamlines while

Technique Target/Context	Dims	State	View-Dependent	Distribution	Reference
Planar Surface Flow	2D	Steady	No	Uniform	[1], [24], [25], [26], [27], [28], [29], [30], [31], [32]
	2D	Steady	No	Non-Uniform	[33], [34]
	2D	Steady	Yes	Uniform	[35]
	2D	Steady	Yes	Non-Uniform	[36]
	2D	Unsteady	No	Uniform	[37], [38]
Curved Surface Flow	3D	Steady	No	Uniform	[39], [40]
	3D	Steady	Yes	Uniform	[41], [42], [43]
Volume Flow	3D	Steady	No	Uniform	[44], [45], [46], [47], [48], [49]
	3D	Steady	Yes	Uniform	[50], [51], [52], [53], [54], [55], [56]
	3D	Steady	Yes	Non-Uniform	[57], [58], [59]
	3D	Steady	No	Non-Uniform	[11], [60], [61], [62], [63], [64], [65], [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76]
	3D	Unsteady	No	Non-Uniform	[77], [78], [79], [80], [81], [82]
	3D	Unsteady	Yes	Uniform	[83]
Streamsurface Construction	3D	Steady	No	Uniform	[84]
Particle-Based Vis	3D	Unsteady	No	Uniform	[85]
	3D	Steady	No	Non-Uniform	[86]
Texture-Based Vis	3D	Steady	No	Uniform	[87]
	3D	Unsteady	No	Uniform	[88], [89]
	3D	Unsteady	No	Non-Uniform	[90]

TABLE 1: Grouping of algorithms based on the target or application context, dimensions, state of flow, dependency on viewpoint, and distribution.

highlighting flow features — we categorize this as a feature-based approach and not a density-based approach. It is common that approaches will overlap with respect to motivation given the desired characteristics are not mutually exclusive, i.e., an algorithm may strive to achieve several desired characteristics in some order of priority.

2.1 Density-based

Density guided automatic techniques are typically proposed when a uniform or user-defined distribution is the desired outcome of a seed point placement or streamline selection algorithm. A uniform density or distribution of streamlines provides the user with a view of the entire flow field. These techniques usually select or calculate approximately evenly-spaced streamlines in object or image space. We categorize the density based approaches as view-independent or view-dependent/image-guided. View-independent approaches don't consider the image-space, i.e., the resultant set of streamlines do not change if the view of the domain changes. Most two dimensional techniques are presented as view-independent given they do not consider change of the viewing angle (nor would it make sense to in 2D). Image-guided or view-dependent techniques consider the final representation in image-space. These techniques might select a different set of streamlines when the viewpoint changes.

2.1.1 View-Independent Techniques

View-independent approaches propose techniques to obtain a uniform or user-defined distribution or often even-spacing between streamlines in object space. In two dimensions, object and image space are the same, however, in three dimensions, the view direction of the domain can influence the quality of the visualization. In these set of works, they do not consider the viewpoint of the domain and the streamline selection is independent of the image-space representation. This approach is faster when producing a visualization which does not need to calculate and render a new set of streamlines when the viewpoint is changed.

In the context of streamsurface construction, Hultquist et al.'s early work utilized a key concept regarding maintaining distances between streamlines [84]. They consider seed point addition and removal during streamsurface construction. After seed points are initialized along a rake, the distance between particles is tracked as particle trajectories (streamlines) are integrated. Based on the premise that to achieve a good visualization an approximately uniform spacing between particles is desired, new seeds are added or existing seeds are merged based on a user-defined neighboring particle distance criteria.

Algorithms Using Local Seeding Strategy

Max et al. [39] use evenly-spaced short streamlines to visualize a 3D vector field on a contour surface. They consider several projections to visualize the streamlines. While they evaluate different projections (Eye, Normal, XY, and Cylinder) on the 3D surface and transitions of those projections as the view changes, a precomputation phase involves seed point selection and particle tracing in a view-independent fashion. To allow streamlines to be traced for

long distances before they get too close to each other the initial positions of seed points are chosen on an integer lattice in a spatially hierarchical manner. A streamline length threshold is used in order to determine the minimal length of accepted streamlines. A streamline grows until it reaches a surface edge, a singularity in the field, or becomes too close to another streamline.

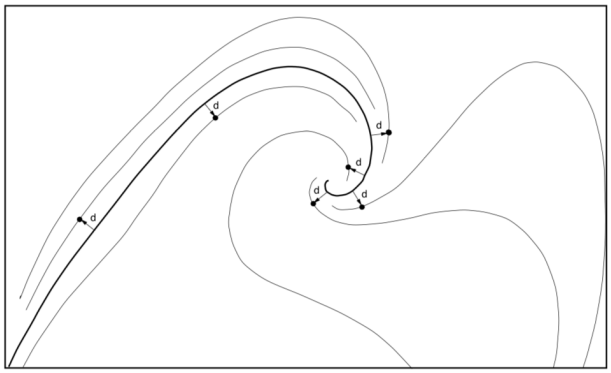

Fig. 2: New streamlines are derived from the first streamline calculated (thick) by identifying locations for candidate seed points a minimum separating distance away [24]. Image courtesy Jobard and Lefer.

Jobard and Lefer extended the work done by Max et al. [39] and proposed an effective single pass method for placement of long evenly-spaced streamlines in a 2D steady state field [24]. The method can achieve visualizations ranging from dense texture-like to sparse hand-drawing styles by only setting the separating distance, denoted by d_{sep}, between adjacent streamlines. The algorithm initially placed a random seed point and integrated a new streamline backward and forward until some termination criteria, similar to those in the work by Max et al., is met. The first streamline is used to calculate a set of candidate seed points d_{sep} distance away from the streamline. The candidate seed points are added to a queue to be evaluated. Each candidate seed point is used as a starting location to integrate a streamline until it is within some distance d_{test}, a fraction of d_{sep}, distance from existing streamlines. Figure 2 illustrates seed points, a user-defined distance away from an initial streamline, used to integrate new streamlines. If the integrated streamline is accepted, then the new streamline contributes a set of candidate seed points to the existing queue. To accelerate the computation process, they proposed two optimizations —

- Streamlines consist of a set of sample points that are evenly spaced and a distance smaller than d_{sep} apart. Only these sample points are considered in distance computations.
- A cartesian grid with cell side exactly d_{sep} is superposed on the domain with each cell containing pointers to the sample points located within the cell. Thus, distance computations are limited to the cells surrounding the cell containing the sample point under consideration.

These optimizations have been employed in several following research works. The evenly spaced placement algorithm achieves placement quality as good as previous techniques, i.e., work by Turk and Banks [35], while significantly improving computation speeds.

Jobard and Lefer extended their initial work to propose a multiresolution technique for steady state flow [25] and an approach to create animations for visualizing unsteady flow [37]. To generate a sequence of streamline-based images of a vector field with different density (multiresolution), they generate a set of streamlines for a particular separating distance value. The generated streamlines are then used as an initial set of streamlines for the next image in the sequence which has a higher density and uses a smaller separating distance value. This process continues for the desired number of levels of streamline density. The shortcoming of this approach is that streamlines generated for later levels were shorter due to the existence of an initial set of longer streamlines.

For the visualization of unsteady flow in 2D, they proposed a feed-forward algorithm which used *reference* streamlines from one time step to select *corresponding* streamlines in the next time step. Sample points of *reference* streamlines act as initial seed locations to generate candidate streamlines. The best candidate streamline, based on an L2-norm correlation criteria measure, was selected as a corresponding streamline in the next time step. If required, additional streamlines were calculated to obtain a uniform distribution. By correlating instantaneous visualizations of the vector field at the streamline level, they visualized unsteady flow. While the approach had wide applicability in 2D, the generalization of this approach to 3D fields raises perceptual problems and the need of a more accurate correlation criteria stands out.

Mattausch et al. [44] adopt the Jobard and Lefer evenly-spaced streamline algorithm while targeting strategies to improve focus+context techniques and spatial perception with respect to 3D flow fields. They extend the streamline placement algorithm to 3D, which involves calculating 6 candidate seed points at a distance d_{sep} for every sample point on a streamline. In addition to generating streamline for a 3D domain, they improve the multiresolution technique presented by Jobard and Lefer and prevent the generation of shorter and shorter streamlines for higher levels of detail.

The Jobard and Lefer algorithm has been utilized as an intermediate step for texture-based flow visualization techniques and for domains outside flow visualization such as DTI Fiber Tracking [91], [92]. Li et al. [87] present Chameleon, a texture-based rendering framework, which decouples the calculation of streamlines and the mapping of visual attributes allowing the user flexible control of the visual appearance of the vector field. They employ the evenly-spaced streamline placement algorithm to control the length and density of the generated streamlines. A trace volume is created using a dense set of streamlines and their geometric properties. The trace volume can then be combined with varying input appearance textures to produce a wide range of effects at runtime. Shen et al. [88] extend the Chameleon framework to support unsteady flow fields by calculating pathlines instead of streamlines. The approach involves bookkeeping to track pathline segment intersections and trace volume updates during rendering.

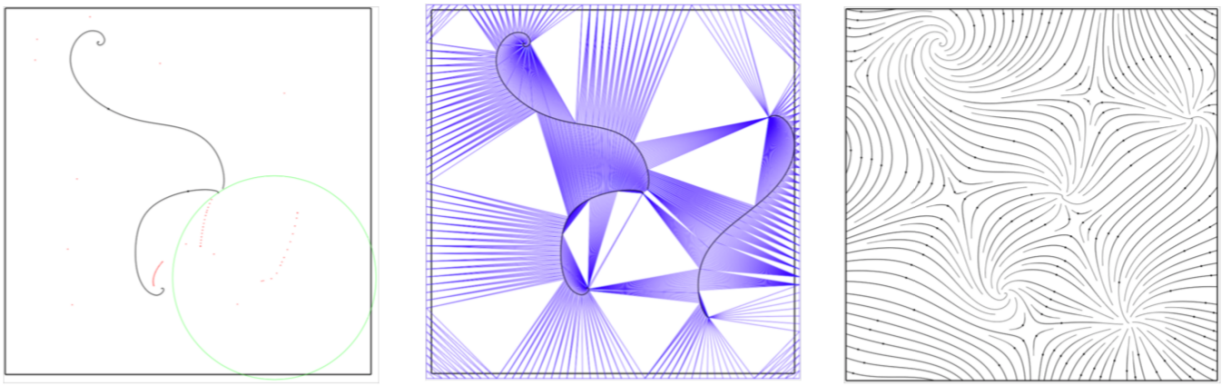

Fig. 3: The intermediate stages of the algorithm presented by Mebarki et al. [26]. Left: circles the largest void in the field. Middle: shows the use of Delaunay triangulation to identify triangles with a large circumradius. Right: shows the final placement of streamlines using the algorithm.

However, they do not address the distribution of pathlines across the domain over time.

Employing a pipeline similar to Chameleon, Helgeland et al. [89] propose a method to use evenly distributed particles as input for a texture-based visualization of unsteady flow in 3D. They propose a particle advection strategy, inspired by the Jobard and Lefer algorithm [24], that outputs a point set instead of a set of streamlines. Using an initially random pool of seed points, they apply the Jobard and Lefer evenly-spaced streamline algorithm to identify the subset of seed points that generate a set of streamlines d_{sep} distance apart. The resultant point set is used to generate streamlines using a texture-based method (for example, Seed LIC [93]) for each time step. When seed points are advected forward to the next time step, cluttering is avoided by removing particles that are lesser than d_{test} distance apart. While particles leaving the domain are naturally removed, they consider adding particles to account for inflow. A seed point is added to the center of a boundary voxels if a fixed length streamline traced from it is d_{sep} distance from existing streamlines. Overall, particle density is maintained by injecting particles into areas with low density without exceeding a user-defined maximum number of seed points for the domain.

Algorithms Using Global Seeding Strategy

While the technique proposed by Jobard and Lefer was a greedy placement of seeds in the neighborhood of previously placed streamlines, Mebarki et al. [26] proposed to place seeds furthest away from all previously placed streamlines. Using an approach, proposed by Chew et al. [94], that had already been successfully applied to point sampling and mesh refinement [95], [96], [97], Mebarki et al. placed new seed points at the center of the biggest voids within the domain. Using Delaunay triangulation to identify voids in the domain, the circumcenter of the triangle with the largest circumradius is chosen as the next seed location. Streamlines, approximated using a set of sample points, are inserted one at a time and are traced until a minimum separating distance criteria is violated. Processing a priority queue of triangles, sorted by circumradius and with circumcircle diameter larger than the separating distance, the algorithm ends when the priority queue is empty. The computation of the process is significantly optimized by only using every n^{th} sample point to calculate the Delaunay triangulation and further filtering the number of triangles added to the priority queue. Only triangles, one on either side of a streamline, which correspond to the local maxima (circumradius) and triangles incident to the streamline extremities are considered. Placing seed points farthest away from existing streamlines resulted in long streamlines, improving on the quality of streamlines placement by reducing streamline discontinuities. Figure 3 illustrates various stages of the streamline placement algorithm. Mebarki et al. demonstrated reduced execution time and placement quality compared to the Jobard and Lefer algorithm [24], while retaining the placement quality of Turk and Banks [35] but significantly reducing computation costs for 2D domains.

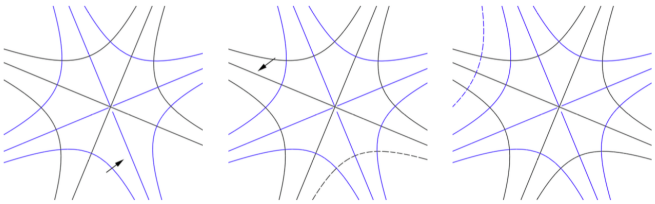

Fig. 4: The dual streamlines algorithm proceeds by identifying the largest segment in two sets (black and blue, i.e., primal and dual streamlines respectively) of streamline segments [40]. Left: arrow indicates midpoint of the largest segment, i.e., the new seed location to calculate a streamline. Middle: arrow indicates next largest segment, i.e., next streamline seed location. Image contains streamline generated from seed point in left image (dotted-line). Right: Result after placement of next streamline (dotted-line).

Motivated by the objective of studying flow phenomena near wall regions or boundaries, i.e., curved surfaces in 3D, Rosanwo et al. [40] propose a greedy algorithm for the streamline placement method. Similar to previous approaches a single distance δ is used to control streamline density. However, the method avoids the computation of geodesic distances and reduces the search space for seed

placement to a set of curves. The algorithm employs two sets of streamlines, namely, *primal* and *dual* streamlines. Primal streamlines are tangential to the vector field at every point and used to visualize flow phenomena. Used to approximate the largest uncovered areas in the domain, dual streamlines are a supplementary set of streamlines that are orthogonal to the vector field at every point. A small set of both primal and dual streamlines can be initialized either randomly or by using flow field topology. Given the orthogonal directions of the two sets of streamline, they intersect at several locations. Segments of primal streamlines are stored in a priority queue P, ordered by arc length. Similarly, segments of dual streamlines are stored in a priority queue D, ordered by arc length.

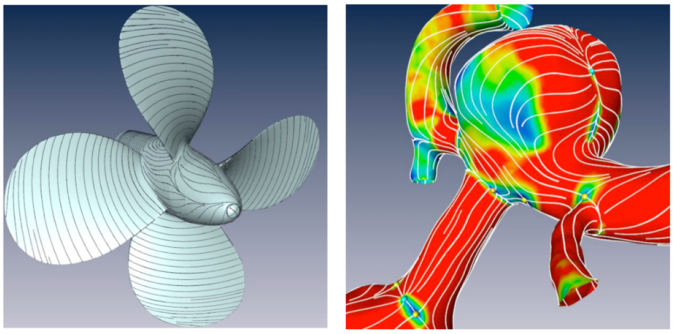

Fig. 5: Results of the dual streamline algorithm for surfaces in 3D flow fields [40]. Left: The image shows a ship propellor and streamlines of the wall shear stress. Right: The image shows a cerebral aneurysm and the wall shear stress streamlines.

The algorithm iteratively selects the longest arc in P or D and places a seed at the midpoint to calculate the next streamline, followed by both queues being updated based on new intersections and segments. The algorithm stops when the length of the longest segment is less than twice the value of δ. Figure 4 illustrates the steps involved in the algorithm. An informed placement of the initial set of streamlines can reduce time to convergence for the algorithm and highlight flow topology resulting in speedups of 2x-3x and improved streamline placement quality over previous approaches [24], [26], [35] when evaluating streamline placement for planar surfaces. Figure 5 demonstrates the use of the approach for curved surfaces in 3D vector fields. However, given the orthogonal structure of a 1D streamline in 3D is a 2D surface, the use of dual streamlines does not directly extend to 3D spaces.

Zhang et al. [30], [32] propose a method to place streamlines in parallel for 2D flow fields. They define local tracing areas (LTAs) as subdomains enclosed by streamlines and/or field borders, where the tracing of streamlines is localized. Using an irregular domain decomposition strategy, the initial LTA is recursively partitioned into hierarchical LTAs. Within an LTA, if a valid seeding area (VSA, determined by streamline proximity criteria) exists, a new seed point is placed at the centroid of the biggest VSA. They use a cell marking technique, instead of performing distance checking, to mark zones where seeds can be placed and streamlines traced. The authors further extended the algorithm to support multiresolution and 3D flow fields [31], [49]. A comparison with Mebarki et al. [26] showed equivalent or better placement quality but was a magnitude order faster computationally when using parallel hardware.

Algorithms In Situ

Within the context of extracting flow information in situ, Agranovsky et al. [10] place seed points along a uniform grid to calculate a flow map. Pathlines or *basis flows* are calculated for each seed for a fixed duration before being terminated and saved. The process repeats by placing seeds uniformly in order to maintain coverage of the domain. Agranovsky adopted this Lagrangian-based technique to tackle the inaccuracies resulting from temporal sparsity when visualizing unsteady flow data from large simulations on supercomputers. With a similar motive, Sane et al. [] propose an alternative Lagrangian-based technique which begins with seeds being placed along a uniform grid. Seeds are advected for a duration of time before evaluating the need to add or remove particles. However, as opposed to resetting the seeds periodically, the distribution of particles is evaluated using Delaunay triangulation. New seeds are introduced at circumcenters of cells with a large circumradius and seeds are terminated if they exit the domain or are found to be in regions with very small cells.

Summary

The view-independent algorithms presented in this section approached the problem of calculating a representative set of streamlines by placing seed points strategically in the domain. Further, the applications varied for generating a streamline visualization for 2D planes and 3D curved surfaces, to generating input streamlines for a texture-based technique, to generating flow map samples to capture the behavior of time-varying flow fields. The algorithms adopted one of two methods, i.e., generating nearby candidate seeds or place candidate seeds in the largest void of the domain. Algorithms adopting the former approach built on top of the Jobard and Lefer method. The Jobard and Lefer algorithm evolved to support an evenly-spaced generation of integral curves in 3D volume flow, on curved surfaces, and in unsteady state flow. Optimizations typically included methods to reduce the search space by either sampling streamlines using a subset of points along the curve or using a cartesian grid to reduce distance checking or mark cells. Algorithms adopting the second approach generated more streamlines of complete length revealing the flow topology better. Further, these techniques were proposed with computational improvements over a local seeding strategy and were demonstrated to be faster for 2D domains. However, global seeding strategies do not extend well to 3D domains. While most of these algorithms operate in serial, a single parallelization strategy irregularly decomposed the domain to place streamlines to improve computation time.

Overall, these approaches generate aesthetically pleasing flow visualizations and are considered to have solved the problem of density guided streamline placement in 2D and on 3D surfaces. However, a shortcoming of these techniques is that they can often contain redundant streamlines and volume flow visualizations, i.e., 3D domain, would contain

large amounts of occlusion requiring interactive tools to further explore the generated set of streamlines.

With regard to in situ seed placement, if the objective is to generate a 3D flow visualization, a view-dependent technique would likely be more suitable. If the objective is to extract flow map samples and thus occlusion is not a factor, coverage and accurately capturing behavior in regions of interest become priorities. There remains significant scope for research to be done in terms of selecting streamlines under in situ constraints.

2.1.2 View-Dependent or Image-Guided Techniques

While view-independent techniques presented in the previous section produced uniform or user-defined distribution of streamlines, they were primarily restricted to 2D techniques. Methods that produced a set of streamlines for 3D either required interactive exploration or used the streamlines as input for a texture-based flow visualization technique. By only considering object-space these approaches did not account for occlusion which plays a significant role when exploring 3D flow fields. View-dependent techniques presented in this section, take the image viewed by the user into account while determining seed point placement and streamline selection.

Algorithms Using Filters

Pioneering work in the field of streamline placement, Turk and Banks [35] proposed to use a stochastic mechanism to iteratively refine the placement of streamlines to visualize 2D steady state flow. The approach is based on the idea that for a given image containing a set of streamlines, a low-pass filter applied to its corresponding binary image should result in an evenly gray image if the streamlines are uniformly distributed. Areas with streamlines cluttered would have bright pixel values while sparsely represented areas would remain dark in the low-pass filtered image. The energy of the streamline image can be quantified as the sum of difference with a given gray-scale value at each pixel of the low-pass filtered image. The density of streamlines can be controlled by adjusting the size of low-pass filters and optimization of the streamline distribution is realized via iteratively minimizing the energy function. For this work, the applied filter is a circularly symmetric filter kernel from a basis function of cubic Hermite interpolation.

Beginning with streamlines generated from seed points at vertices of a 2D grid, where each streamline has an associated energy contribution, they modify the set of streamlines until the desired energy threshold is reached. The algorithm considers moving, lengthening, shortening, deleting, inserting, and combining streamlines based on energy. Modifications to streamlines are either proposed by an oracle (50%) or is a random modification (50%) to prevent any oracle bias. The Oracle can speed up the convergence of the optimization by a factor of 3x-5x. To propose effective changes, the oracle uses image information to identify sparse regions and maintains a priority queue containing streamlines based on their individual energy level. This way the oracle can suggest regions to insert streamlines or how to lessen the energy contribution of the most energetic streamlines. If the modification lowers the overall energy value of the image, the change is accepted, otherwise, the change is rejected. The process continues until the energy function reaches a threshold or the accepted changes are rare. While this approach produces high quality streamline placement, it is computationally expensive given it can have a long convergence time.

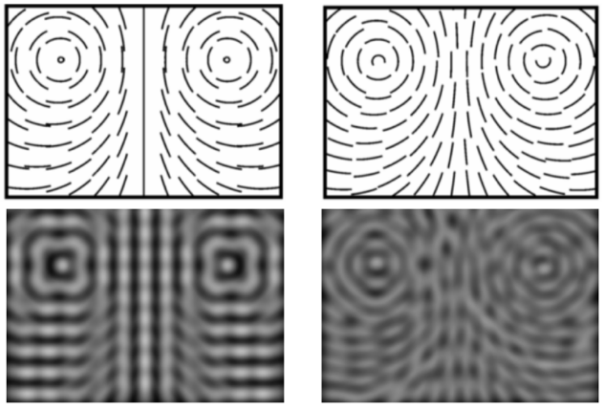

Fig. 6: A comparison of streamline placement and filtered versions using the Turk and Banks image-guided algorithm [35]. Left Column: Short streamlines generated from seeds placed on a regular grid and the corresponding filtered image. Right Column: Short streamlines placed by iteratively optimizing placement and the corresponding filtered image showing fairly even gray value.

Mao et al. [41] extend the Turk and Banks algorithm to uniformly distribute streamlines to a curvilinear grid. They use the expensive image-guided algorithm because density distribution on curvilinear grids, which are anisometric, is hard to achieve with distance-based approaches. The first step of their approach is a mapping of vectors on the curvilinear surface to computational space. To account for the mapping distortion caused by an uneven grid density of a curvilinear grid, a new energy function is designed. Using a Poisson ellipse sampling to distribute a set of rectangular windows in computational space, the streamline density is locally adapted to the inverse of the grid density in physical space. Use of such an energy function ensures the generated set of streamlines of the desired density are evenly distributed after being mapped back onto the 3D surface.

Algorithms Using Image Space Seeding

Uniformly distributed streamlines in 3D space are not guaranteed to be evenly-spaced in their 2D projection. Presenting an approach to avoid clutter when visualizing a 3D flow field using streamlines, Li et al. [50] perform seed placement and streamline termination in image space, and streamline advection in object space. The algorithm operates similar to the Jobard and Lefer algorithm, except that candidate seed points for a streamline are d_{sep} apart from the streamline in image space. Thus, even though a 3D domain is under consideration, for every sample point of the streamline only two possible candidate points are identified. A streamline, being advected in object space, is terminated if it is within d_{sep} from another streamline in image space.

Further, a streamline closer to the viewpoint is preferred to another far behind. This criterion is evaluated when determining whether to terminate the streamline advection. To support importance driven seed placement, they adopt an approach which decouples seed point generation and streamline spacing control. They first generate a set of seed points using a process that stochastically generates more seeds in a region of interest, followed by calculating the corresponding streamlines in object space. To avoid clutter, streamlines which violate spacing requirements in image space are deleted, while favoring longer streamlines. This approach by Li et al. was the first work which used an image space based seeding strategy.

Spencer et al. [43] present an evenly-spaced streamline seeding algorithm for vector fields defined on surfaces in 3D space. The algorithm is capable of generating both sparse and dense representations of the flow and can handle large, complex, unstructured, adaptive resolution grids with holes and discontinuities. Streamlines are only integrated for the portions of the surface visible in image space. The advection strategy removes the need to perform streamline tracing on a triangular mesh and instead projects the vector field onto the image plane. Seed placement and streamline integration are then done in image space. They store the flow data in a "velocity" image where each pixel stores the flow velocity on the surface and a 16-bit representation of the z-depth representing the distance of the surface. The use of a z-depth buffer allows the algorithm to disregard non-visible portions of the surface and plays an important role in detecting discontinuities or edges. The approach

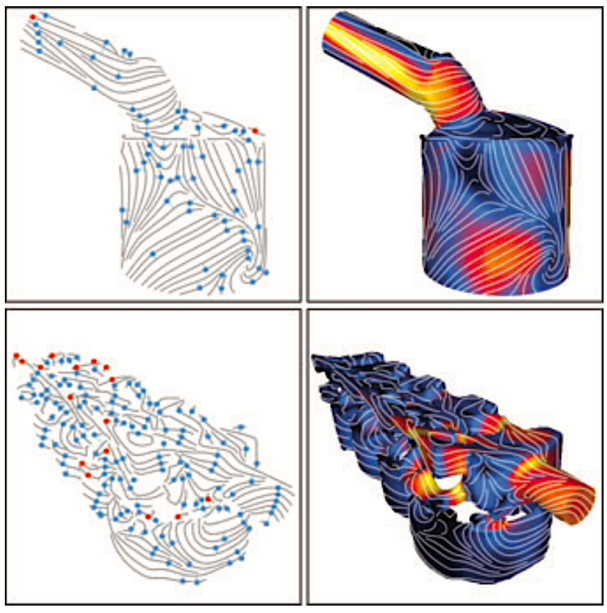

Fig. 7: Results for placement of streamlines on surfaces in 3D flow using the algorithm by Spencer et al. [43]. Seed placement is done in image space. Left Column: The images show the seed locations in image space. Grid-based seeds are shown in red and vector field-based seeds are shown in blue. Right Column: Final streamlines placement.

places seed points, called grid-based seeds, in every cell of the mesh with non-zero depth. In addition to grid-based seeds, it generates vector field-based seeds, i.e., candidate seed points, in a similar manner to the Jobard and Lefer algorithm. They terminate a streamline when the proximity to another streamline drops below d_{test} or when z-depth drops to zero or the change in z-depth exceeds a user-defined threshold. Using both sets of seeds in combination ensures all visible sections (there are potential geometric discontinuities arising from edges and occluding surfaces) have a uniform distribution of streamlines. To avoid terminating streamlines near edges due to proximity in image space (the streamlines are a greater distance apart in object space) they check if streamlines have approximately the same z-depth buffer. To create improved depth perception in the visualization, they vary the value of d_{sep} based on the depth. The idea of reducing any complex surface to a 2D problem results in a computationally efficient algorithm. Figure 7 illustrates a result of the approach. Spencer et al. used a GPU to improve rendering times and showed their streamline generation is faster than a simple Jobard and Lefer algorithm in 3D object space.

Algorithms Using Occlusion and Projection

Given the extensive use of contours to visualize scalar fields, Annen et al. [57] introduce the concept of vector field contours for flow exploration. They propose a streamline seed placement approach with the goal of generating isolated streamline lines which display behavior similar to that of classical surface contours. The approach is view-dependent in that seeding structures are identified by locating points where dot product of the view direction and the vector field is zero, and a seed which takes one infinitesimal integration step preserves that condition. Multiple rendering passes are applied to extract the seeding structure with curvature being used in a similar manner as an isovalue in a scalar field. Streamlines are then forward and backward integrated until the dot product of the vector at the streamline position and the view direction exceeds a threshold. The extraction and rendering of the vector field contours is inter-frame coherent, with the flow field capable of being interactively explored in real-time.

Marchesin et al. [51] select streamlines which contribute to understanding flow field characteristics, while simultaneously accounting for cluttering for a given view. The approach uses streamline features and the occlusion caused by it, to decide whether to include a particular streamline. An iterative method consisting of four main stages, the algorithm begins with the computation of a random pool of streamlines. Projecting all the computed streamlines onto an occupancy buffer helps identify highly occluded regions for a given view. Given the importance of swirling lines to understand flow behavior, the occupancy buffer does not account for self-occlusion caused by a single streamline and simply measures the screen space footprint. Thus, for each pixel, the number of streamlines projecting onto this pixel is calculated. Lastly, the values in the occupancy buffer are normalized to account for the depth of the domain for a given viewpoint. The third stage is a pruning step, with a streamline removal algorithm evaluating the relevance of a line and the amount of occlusion it produces. They use the linear and angular entropy values of segments of a streamline to determine the quantity of information

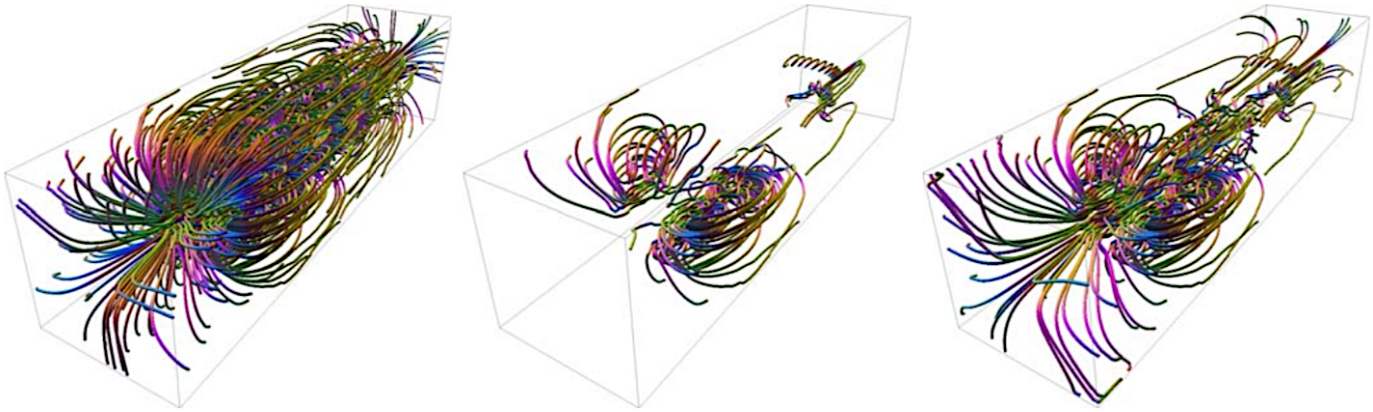

Fig. 8: Stages of the Marchesin et al. [51] algorithm. Left: an initial dense pool of streamlines. Middle: streamlines which are rated high based on the importance metric. Right: Final result after the addition of additional streamlines to create a more uniform distribution of streamlines in image space.

conveyed by the streamline. Additionally, they consider an overlap value to determine the occlusion caused by the streamline for the given view. Combining these values, they present a streamline metric which is a weighted sum of the linear and angular entropies divided by the average overlap. Sorting streamlines by their score, the streamlines with the lowest score are iteratively removed, followed by an update of the occupancy buffer, and affected streamlines. The final stage of the algorithm decomposes the occupancy buffer into a number of tiles and computes the average occupancy for each tile. Seeding a small pool of random streamlines from the tile with the lowest occupancy, the streamline resulting in the least occlusion is retained. This process is repeated until all tiles have a non-zero occupancy. Figure 8 illustrates a result of the approach and shows the evolution of the visualization over the algorithm stages. The approach captured features of the flow better than previous view-dependent methods and required a GPU for fast computation.

Gunther et al. [53] present an interactive, view-dependent, and inter-frame coherent flow visualization technique whose results are dependent on user-driven seed placement. The method has an initial preprocessing step which involves both user-guided seed placement using a seed box, and random placement to generate streamlines that cover the entire flow field. For each streamline, a *screen contribution* value is computed by using a cubic Hermite interpolation function to map the number of visible pixels of the streamline to a transparency value. The screen contribution values are used to determine which streamlines are visible to the user for a given view and fade out streamlines with only minor contributions. Given one important region of the flow can occlude another, the user can selectively place seed boxes in order to focus on certain regions. To support exploring regions of coherent flow, the user can highlight a set of similar streamlines by selecting a single streamline. Streamlines in a limited screen-space neighborhood window of the chosen streamline are evaluated for similarity using linear and angular entropy. Gunther et al. [54] extend their previous work by adopting a global line selection strategy. Starting with an initially dense sampling of the domain, the approach is based on computing the opacity for every streamline segment in the field as a solution to a bounded-variable least-squares optimization problem. Their system supports a flexible choice of importance measure of a streamline segment such as curvature, linear entropy, angular entropy, scalar entropy, segment length, or *screen contribution*. Depending on the metrics chosen the algorithm highlights relevant features in the flow field by minimizing the occlusion caused by other streamlines. Figure 9 shows a sample result of the algorithm. While the optimization problem is based on the total number of streamlines segments in the flow, when considering unsteady state flow the number of segments increases significantly and can become a bottleneck. To tackle the challenge of 3D unsteady flow, Gunther et al. [83] modify their approach and employ a hierarchical representation of an integral curve and consider only a view-dependent set of candidate segments for the optimization process. Gunther et al. use the GPU to achieve frame coherent, time coherent, and interactive flow exploration, thus improving on previous research.

Ma et al. [55] present a view-dependent streamline selection approach that evaluates the information content of streamlines. As a preprocessing step, a dense set of streamlines intersecting every voxel in the domain is computed. Next, for every sample viewpoint, the streamlines are sorted on the basis of importance. The streamline importance measure consists of entropy (considering both direction and magnitude) measured along the streamline, an evaluation of how much entropy is preserved for a given 2D projection, and a shape characteristic metric indicating how stereoscopic the streamline shape is for a given viewpoint. The last two factors together forming a view-dependent importance measure for each streamlines for each viewpoint. To identify a set of view-independent representative streamlines, streamlines are inserted into a priority queue based on the summation of their view-dependent importance measure for each view. A minimum threshold distance is used to avoid selecting redundant streamlines. To generate a view-

Fig. 9: Opacity optimization demonstrated by the algorithm by Gunther et al. [54]

dependent set of streamlines, the top-ranked streamlines for that viewpoint are combined with the highest rated streamlines from the view-independent set. Further, to maintain coherence as the viewpoint is changed, streamlines from a previous viewpoint are retained. A density map is employed to determine uncovered regions before rendering the final visualization. Their approach was able to generate fewer redundant streamlines compared to Marchesin et al. [51] given the use of the occupancy buffer.

Summary

To generate a view-dependent selection of streamlines, algorithms were based on either strategic seed placement or informed streamline selection. The earliest work used a filter over an image of a set of streamlines and iteratively improved the placement and selection of streamlines. However, these methods were computationally expensive given the time to convergence was long and they do not extend to 3D volumes. Li and Shen identify the selection of streamlines in 3D volumes as a view-dependent problem and propose to place seed points in image space and thus directly manage the density distribution of the streamlines. Another work reduced the problem of generating streamlines on a surface in 3D to a 2D problem by only considering visible surfaces for a given view and placed seed points in the image space as well. Thereafter, view-dependent algorithms used a dense set of streamlines, evaluating attributes along the curve and considering the 2D projection of each streamline. Streamlines were then selected for a given viewpoint, by considering the importance measure and the screen space occupancy of the streamline. Methods to handle occlusion involved completely removing a streamline that occludes a more important region or techniques such as opacity adaption to retain context information.

View-dependent methods have evolved from requiring long convergence times for a 2D plane visualization to interactive exploration rates for 3D volumes using GPUs. In addition to these view-dependent approaches, research in other sections also consider the viewpoint of the visualization, however, their streamline selection isn't primarily motived by the viewpoint.

2.2 Feature-based

Feature-based techniques make use of available vector field information to guide the seed placement or streamline selection. Prioritizing coverage of interesting regions of the flow field over a uniform distribution, these approaches aim to first take measures to ensure streamlines are placed taking into account either explicitly extracted flow field topology or derived informative scalar fields. We classify feature-based approaches depending on whether they explicitly extract flow field topology for precise information or allow a derived scalar field to guide the algorithm. Approaches choosing to use a derived scalar field, do so in order to either capture some particular flow field behavior or as an alternative approach to capture salient flow features, such as critical points (Figure 10), which are often difficult to find in a robust manner. As we will see in the following sections, several of the feature-based techniques, in addition to their specific approach to highlight features, adopt uniform seed placement strategies presented in the previous section.

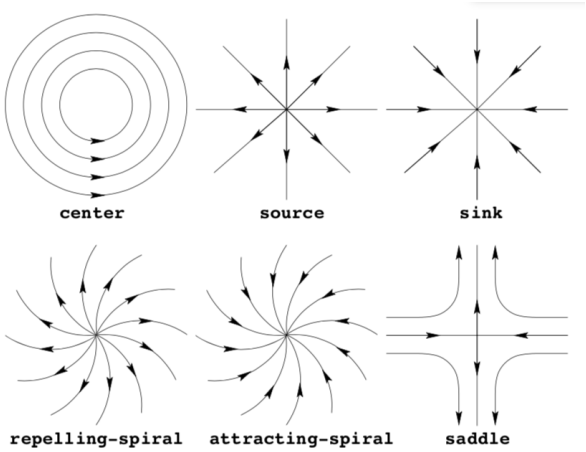

Fig. 10: Different types of critical points possible in 2D flows [1].

2.2.1 Explicit Flow Topology Guided Techniques

Algorithms Using Critical Point Locations

An early work presenting a multiresolution visualization technique for nested weather models, Treinish [61] proposed to use a combination of critical point analysis and a filter similar to Turk and Banks [35]. Deriving a set of seeds using a low-order approximate critical point analysis, an initial set of streamlines is computed. A low-bandpass filter is subsequently applied to the entire forecasted velocity field, as opposed to an image of the streamlines, to identify regions with a relatively large change in wind speeds. Seed points are placed in these regions to calculate additional

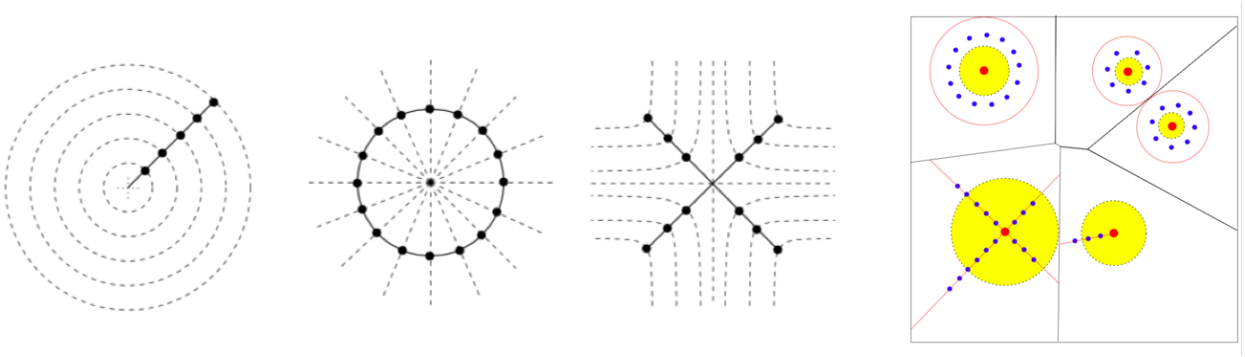

Fig. 11: Images show seed templates for various critical points and regions of influence identified by the flow-guided algorithm [1].

streamlines. The technique was superior to using uniformly sampled seed points and captured detailed features from the forecast. However, this particular work did not provide specifics regarding the placement of the seed points in relation to critical points or regions of interest.

Verma et al. [1] propose the use of critical point specific seed placement templates. The primary goal of their approach is capturing flow behavior in the vicinity of critical points. The algorithm first identifies the locations and types of critical points in the 2D flow field using FAST [98] followed by a segmentation of the domain into an approximation of critical point neighborhoods. For the approximation, a Voronoi diagram, computed using *triangle* [99], partitions the flow into regions containing similar flow behavior. A second objective is to provide sufficient coverage of non-critical regions. After tracing long streamlines using the template seeds, a region of influence is determined for each critical point. In the spaces outside the regions of influence, Poisson disk distribution is used to place additional seed points, with streamlines being generated using an approach similar to the Jobard and Lefer algorithm. Figure 11 shows the seed placement templates, and a sample field with critical points, corresponding templates, and regions of influence, and the field partitioning such that each partition contains a single critical point. The approach was able to better capture flow behavior around critical points in both dense and sparse flow representations when compared to previous techniques [35] while being computationally faster when selecting a greater number of streamlines.

In addition to extending the template-based approach to 3D steady state flow, Ye et al. [45] propose improvements to the algorithm. To account for the distance between and relative strength of critical points, they change the shape of the templates by mapping how eigenvalues of one critical point evolve into the eigenvalues for another. To determine the size of seed templates, separating regions need to be calculated. To calculate separating regions in 3D for each critical point, instead of using expensive full topological analyses [100], [101], as an optimization an approximation is used. The size of the seeding template is set to a quarter of the distance to the nearest other critical point. Poisson sphere distribution is used to fill areas between critical point regions of influence. Given the streamlines are placed in 3D, without any consideration of the view, the image would likely appear cluttered. This work included a post-processing step in order to filter streamlines to provide a less cluttered visualization. Streamlines are filtered on the basis of length, accumulated winging angle, and proximity to other streamlines. First removing short streamlines with low winding angles, followed by identifying a single representative streamline for a set of streamlines that have a similar start, end, and centroid location. For a cell with a high streamline count, denoting a dense region, streamlines with high winding angles are filtered to reduce cluttering.

Liu et al. [46] propose an evenly spaced streamline algorithm (ADVESS) that employs two queues of candidate seeds. The primary queue consists of an initial set of seeds generated using the templates used by Verma et al. [1]. Candidate seeds generated from these initial set of streamline seeds are also added to the primary queue in order to maximize the effect of the seeding patterns. The secondary queue consists of candidate seeds generated from sample points along the streamline. As an optimization, Cubic Hermite Polynomial interpolation using large sample spacing is used to reduce the number of sample points and consequently distance checking. An additional improvement on previous evenly spaced streamline algorithms is the use of an adaptive d_{test} value based on the local variance measured at each grid point in the 2D field. Appropriately scaling the value of d_{test} causes fewer cavities in the streamline placement. Further, the authors propose a robust loop detection technique which limits a streamline loop to a single cycle [47]. Employing the proposed algorithm as one part of a hybrid seed placement approach, Liu et al. [42] present a view-dependent approach for streamline placement on a planar or curved surface in 3D. Using the double queue strategy differently, Poisson disk distribution is used to push a set of seeds to the secondary queue and begin the process. Candidate seeds introduced by the seed of the accepted streamline are stored in the primary queue, and non-seed sample generated candidate seeds in the secondary queue. The approach is used for the purposes of image space streamline placement to fill spaces after a primary set of physical space seeds are used to generate streamlines that are reused and lengthened between view frames. The combination of the two strategies provides a temporally coherent visualization. In comparison to previous approaches, the algorithm achieved placement quality better than Jobard and Lefer [24] and as good as Mebarki et al. [26] with loop

detection, in addition to being computationally faster than both.

Ding et al. [38] present a technique to maintain temporal coherence when viewing unsteady 2D flow fields by using a moving mesh method. Another extension of the Jobard and Lefer algorithm to evenly space streamlines, they first extract critical points to calculate an initial set of candidate seeds. Using Poisson disk distribution [102], they place seed points in regions of importance and add them to the queue of candidate seeds. They move the seeds towards the critical features by creating and deforming an auxiliary mesh along with the evolution of the vector field. They use a similar feed forward pipeline system to identify corresponding streamlines to maintain temporal coherence across frames.

Algorithms Using Flow Topology As Initial Set

In addition to proposing a technique to systematically create and cancel fixed points and periodic orbits in a vector field, Chen et al. [27] modify the Jobard and Lefer algorithm to highlight the vector field topology. Motivated by the visual discontinuity in periodic orbits and separatrices in current techniques, before using the seed placement algorithm, they first extract periodic orbits and separatrices and make them the initial streamlines. Further, to avoid clutter near sources, sinks, and periodic orbits, they terminate a separatrix if it is within a user-defined distance from the non-saddle end.

Preceding the parallel hierarchical local tracing area algorithm presented in Section 2.1.1, Zhang et al. [28] proposed to extract the underlying flow topology and use the topological skeleton as the initial set of streamlines that segment the field. Following the partitioning of the field, additional streamlines are calculated by placing seed points at the center of topological areas in a recursive manner creating an approximately uniform distribution of streamlines. They extend the vector field domain in each direction by adding a layer of mirrored boundary cells [103]. Using the additional critical points from the extended vector field helps capture open separation and attachment lines.

Similar to the previous work, Wu et al. [29] extract the flow field topology and partition it into regions of uniform flow behavior. However, as opposed to adopting a recursive method, they search for the longest path that orthogonally crosses all streamlines within a region. Seeds are placed evenly along the longest path to produce approximately uniformly placed streamlines. They treat periodic orbits and saddle-connected loops as special cases. Using vector field reconstruction error as a quantitative measure for comparison (Figure 12), they demonstrate superior streamline placement than previous works [24], [26], [27], [46]. We can observe that as a sparser set of streamlines is used, i.e., as separating distance increases along the x-axis, the algorithm results in lower reconstruction error in comparison to other algorithms. Figure 13 illustrates the use of the longest orthogonal path as a seeding curve.

Summary

Algorithms using explicitly extracted flow topology information adopted both seed placement and streamline selection approaches. One set of works used critical point

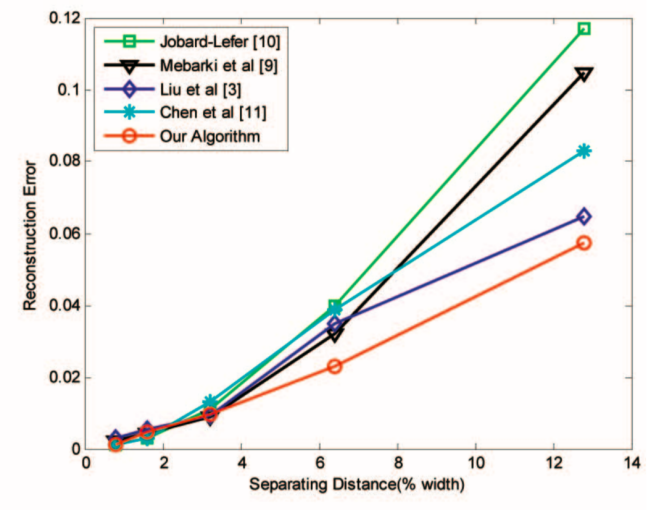

Fig. 12: A quantitative comparison using reconstruction error for five algorithms. Our algorithm in the plot is the method presented by Wu et al. [29].

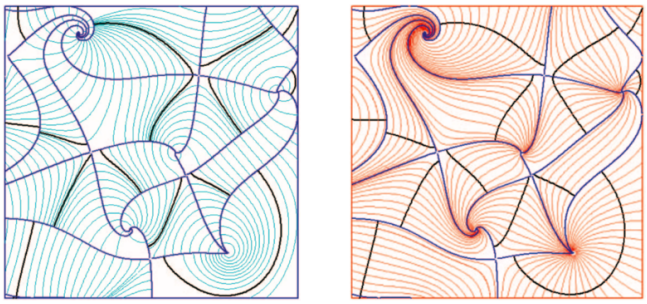

Fig. 13: Wu et al. place seed points along the longest orthogonal curves (black) [29]. Left: Orthogonal curves (light blue, black) and extracted flow topology (blue). Right: Orange streamlines are traced from seeds placed along longest orthogonal curves.

locations to use templates of seed points in the domain. The templates are critical point type specific and were designed to capture the flow feature accurately. Improvement on the original set of templates involved extending the 2D templates to 3D flow and modifying the shape of the template based on neighboring regions of the flow field. With regard to streamline selection, certain works directly selected streamlines showing periodic orbits and separatrices after extracting critical points. Other works use the extracted flow topology to divide the flow into regions of relatively uniform behavior before further seed placement in each region.

These works successfully highlighted flow topology and represented flow behavior. Wu et al. demonstrated a lower vector field reconstruction error compared to multiple previous approaches. However, some of these techniques are limited to 2D domains and require being able to robustly calculate flow topology.

2.2.2 Scalar or Derived Field Guided Techniques

Extracting the topological structure of the flow field can often be difficult to find in a robust manner for real-world applications, and several research efforts have proposed the use of derived fields as an alternative approach to guide seed placement and streamline selection to capture salient flow features.

Algorithms With General Applicability

Zockler et al. [60] propose to use a statistical method to facilitate the placement of streamlines with density proportional to some scalar quantity. Considering a uniform grid over the domain, for each cell, a local degree of interest is computed. Cells are selected on the basis of the parameterization of a probability distribution using the local degree of interest. Seeds are then added to those cells, with streamlines grown for a fixed length with forward and backward integration from the point of placement. If the range of values for the chosen scalar ranges over multiple orders of magnitude, streamline distribution can be unsatisfactory. To tackle this problem, a histogram equalization approach is used to obtain a more homogenous distribution of streamlines. This technique was applied by Weinkauf et al. [104], [105] using fields of curvature and torsion.

Schlemmer et al. [65] present a heterogeneous distribution of streamlines based on a density map derived using scalar fields (temperature, viscosity), derived vector field information (magnitude of velocity, vorticity, field topology), or a user-defined density function. They define streamline density as the number of occupied cells over the total number of cells in a domain. To calculate *priority* streamlines, they first define a density map used to guide seed placement, with the map updated after every streamline calculation. The first seed is placed at the location of the maximum value of the initial density map. The next location is chosen as the furthest of the next five maximum values and the process continues. Given the density map is monotonically decreasing over time as streamlines are added, the algorithm will eventually terminate. Figure 14 shows the results for a flow data set comparing a constant density map and a user-defined density map.

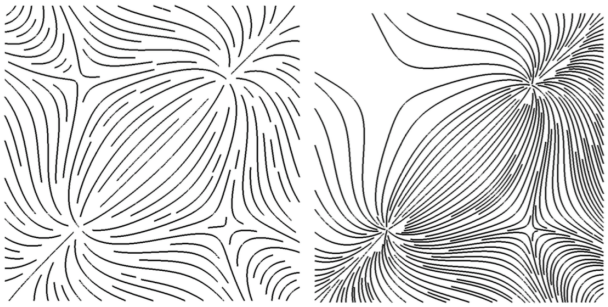

Fig. 14: Schlemmer et al. proposed the use of a density map to modify streamline distribution based on attributes of interest [65]. Left: a constant or uniform density map. Right: a non-uniform density map: density values decrease from lower right to the upper left corner.

Algorithms Using Entropy (Information Theory)

Furuya et al. [58] consider streamline selection when the visualization is combined with scalar field isosurfaces. Their approach generates a large number of seed points to capture interesting flow features by the integrated streamlines. The entropy of segments of individual streamlines is measured and used as a basis for selection. The measure of streamline entropy accounts for occlusion caused by isosurfaces by penalizing a streamline if segments of the curve are occluded. Finally, streamlines are sorted and selected in order of highest entropy. To control density, streamlines are only rendered if they are some minimum threshold distance away from existing streamlines.

Xu et al. [66] present a flow visualization framework based on the premise that the effectiveness of a visualization can be evaluated by measuring how much information in the original data is being communicated. In this work, they empirically demonstrate that entropy in regions near critical points and separation lines is higher than that of other regions. Modeling a vector field as a distribution of directions, Shannon's entropy is used to measure the information content in the flow domain. Figure 15 illustrates the use of a polar histogram to capture the distribution of vector directions for a given neighborhood of the field. The effectiveness of the streamline placement is measured

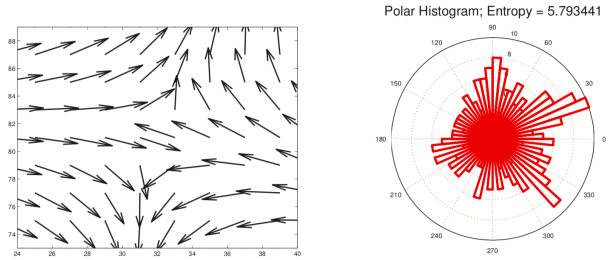

Fig. 15: Xu et al. measured entropy or information content in neighborhoods of the vector field [66]. Left: an example vector field neighborhood. Right: the distribution of the vectors approximated using a polar histogram.

by reconstructing a distribution of vectors derived from the selected streamlines. The approach begins by iteratively placing seed points in regions of high entropy. They use a diamond shape template to place 9 seed points in 2D and an octahedral shaped template consisting of 27 seed points in 3D. Further, to prevent large voids, they place the next seed at a point proportional to the conditional entropy computed from its local neighborhood. The streamline addition process ends when the value of conditional entropy of the entire domain converges to a small value. A final pruning step removes redundant streamlines. Figure 16 shows the placement of seeds based on entropy, streamlines traced in two stages, and a comparison of the selected streamlines for the proposed approach to previous works [24], [26]. The algorithm provided quantitative control of the selection of streamlines.

Lee et al. [52] extend the framework presented by Xu et al. [66], to support a view-dependent streamline selection aimed at minimizing occlusion and revealing important flow features. Using the derived entropy field, a maximal entropy projection (MEP) frame buffer is computed for a

Fig. 16: The seed placement stages of the entropy-guided algorithm by Xu et al. [66]. Seed points are highlighted in red and the images show (from left to right) initial placement of seed templates, additional seeds used to reduce conditional entropy, a result of the algorithm, and visualization results of the Jobard and Lefer method [24] and Mebarki et al. algorithm [26].

given image space. The MEP buffer stores maximal entropy values, as well as the corresponding depth values for the given viewpoint. To identify the optimal viewpoints, i.e., views which convey maximum entropy information, MEPs of 780 viewports are evaluated. Streamlines are assigned a higher priority if they reveal the flow near salient features and a lower priority if it occludes an important region of the flow. Streamlines are segmented and each fragment in the image plane is evaluated to compute a scalar score ω for each streamline. ω is calculated using information stored in the MEP buffer - depth and entropy. The streamlines are prioritized based on their value of ω which can be positive or negative. To maintain a streamline density proportional to the flow complexity, the screen space is divided into tiles and an expected streamline density equal to the average normalized entropy of the region in the MEP buffer is computed. For a given streamline, if the addition of the streamline affects more tiles with a density lower than the expected density, the streamline is added. This approach favors streamlines which reveal salient flow features without occluding other more important features. They demonstrated improved feature capturing compared to Marchesin et al. [51] for a view-dependent selection of streamlines. Further, even though they significantly benefitted from using a GPU, the serial selection is a bottleneck.

Ma et al. [71] present FlowTour, a framework which selects best viewpoints to explore a flow field visualized using streamlines. A skeleton-based seeding algorithm is employed to generate a set of streamlines which capture critical regions of the field. Variation of both direction and magnitude of vectors are considered to compute the entropy of voxels. Critical regions are identified as local neighborhoods in which voxel entropy values exceed a threshold. Sufficiently large regions of connected voxels with high entropy are used as input to a volume thinning algorithm that extracts the skeleton points. Skeleton points are connected by applying a minimum spanning tree algorithm to produce a tree-structured skeleton line. The density of streamlines is controlled by placement of seed points aligned along the skeleton at an equal user-defined separating distance. Candidate viewpoints are generated on the basis of the critical regions identified in the field. Finally, best viewpoints for each region are selected and connected into a view path using a B-spline curve.

Algorithms Using Derived Vector Field Characteristics

The problem of image space cluttering is exacerbated in 3D unsteady state flow, given pathlines of different particles can occupy the same location in space at different times. To study unsteady state 3D flow, Wiebel et al. [79] introduce the concept of an eyelet. Calculating a set of pathlines or streaklines that pass through the same single point (eyelet) in space at different times yields an insightful static visualization of the unsteady flow field. The collection of pathlines can be used to construct a surface to visualize the flow. If pathlines diverge more than a user-defined threshold, a new seed is added at the eyelet at a time step in between the time steps of its neighboring particles. While this approach is conceptually similar to Hulquist [84], instead of adding a new point at the location where divergence is detected, it is added at the eyelet to guarantee the particle would indeed pass through the eyelet. The placement of the eyelet plays a critical role in the usefulness of the algorithm to study the flow field. They identify regions of high activity by introducing measures to capture the change of a vector field over time. A dot product variation is computed by accumulating the positive dot products of vectors in consecutive time steps. A second measure, the vector variation is the norm of the computed difference of the two vectors considered. Isosurfaces drawn using these variation fields help identify regions of high and low activity for further investigation. Additionally, edges and corners, or regions behind flow passed objects, singularities, and vortex cores serve as good locations to place an eyelet.

Wang et al. [67] visualize explosion fields by strategically placing seed points to calculate streamlines. Given the nature of the simulation, they first generate an isosurface for the magnitude of velocity. The isosurface region is then divided into a series of subregions that are almost equal in area. The center of a subregion is used as a seed point with the integrated streamlines always starting from the center of the explosion and extending outward in a direction perpendicular to the isosurface.

Luo et al. [36] propose a streamline generation technique by combining the use of derived scalar fields and topological methods, such as contour trees and persistent homology. They measure the global importance in terms of persistence of topological features in the vector field and use streamline density in the generated visualization to reflect the same. Hodge decomposition [106] is a technique used to decompose the 2D vector field into a rotation-free component - gradient, a divergence-free component - curl and a harmonic component. Maxima and minima of the gradient field correspond to sources and sinks respectively

and guide the placement of a set of *gradient* seeds. The number of seeds placed is proportional to the persistence of the maxima and minima, measured by determining the amount of perturbation required to smooth out the mountain peak or valley. A contour tree encodes the evolution of level sets of the curl field and is used to generate a set of *curl* seeds. Each branch of the contour tree corresponds to a topological component of the domain. Each branch is assigned a number of seeds proportional to its range function to collectively produce a set of curl seeds. Every seed location is evaluated to determine gradient vector or curl vector magnitude dominance at that position. Only gradient seeds in positions of gradient dominance are used. Similar, only curl seeds in positions of curl dominance are used. They demonstrated superior placement quality in terms of reconstruction error compared to Li et al. [33] and Xu et al. [66].

Yu et al. [68] use the curvature and torsion fields of the flow field to generate a saliency map which is used to guide the placement of seed points. Based on the premise that the metrics themselves may not be able to capture the salient features, a saliency map is computed as the difference between Gaussian-weighted averages calculated at multiple scales, i.e., varying the standard deviation of the Gaussian filter. They compute the saliency map for five threshold distances and then combine all five saliency maps with a nonlinear normalization. While the computation of the saliency map is relatively expensive on CPUs, it can be computed within a few seconds on a GPU accelerator. Given a final saliency map, the seed placement algorithm selects locations in order of decreasing saliency. Generated streamlines are integrated for a large threshold maximum length (favor long streamlines while mitigating loops) and those that occupy the same voxel as existing streamlines are discarded. The use of the saliency map as opposed to directly seeding based on the curvature or torsion fields, allows streamlines to be placed closer to critical points highlighting flow features. Further, the generated set of streamlines is hierarchically clustered to enable exploration at different levels of detail and manage clutter (details in Section 2.3.1).

Zhang et al. [82] investigate the usage of a scalar field Φ derived from the input vector field by integrating the rotation of the integral curves. They place seed points where $|\nabla\Phi|$, the magnitude of the rate of variation of derived field Φ, is greater than a user-defined threshold. Randomly starting from a placed seed point, an integral curve is computed, followed by the filtering of nearby seeds. The process is repeated with the remaining seeds.

To visualize a vortex rope which builds up in the draft tube of a water turbine, Bauer et al. [85] propose a particle seeding scheme to visualize unsteady flow. They use Sobol quasirandom sequences [107] to obtain a uniform distribution while avoiding clustering and artifacts like regular patterns. Given the vortex rope is a rotating helical structure, they evaluate the helicity int the flow domain to identify regions of interest. They propose a scheme to introduce new particles to regions with a scalar value of helicity greater than a predefined threshold by offsetting the original point set of quasirandom sequences. They use a layer of invisible buffer cells, shown in Figure 17, which enable particles to fade in and out smoothly from the region of interest. Similar to the texture-based visualization methods which

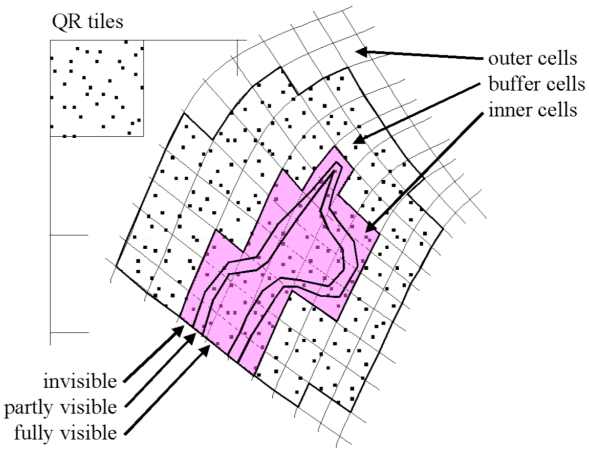

Fig. 17: Bauer et al. identify a region of interest using helicity and maintain particle density using Sobol quasirandom sequences and buffers to manage smooth transitions [85].

use evenly spaced streamlines presented in Section 2.1.1, Guthe et al. [90] present another seed placement approach which distributes more particles in regions of interest. The placement of seed points is based on an adaptive sampling of the field with the goal of achieving a higher sampling resolution in interesting regions and a lower sampling resolution in less interesting regions. They use the local gradient, divergence, and curvature of the vector field to influence the particle distribution. Additionally, they suggest the use of local shear and rotation of the vector field or distance to the closest critical point. An octree data structure is employed to maintain the distribution of particles in the domain. The distribution octree is updated as particles travel along streamlines with particle age being a deciding factor in regard to particle removal in overcrowded regions.

Particle-based visualization systems have seen efforts to improve the interactivity of the flow visualization [108], [109], [110], [111]. Engelke et al. [86] propose a particle system which results in an adaptive particle density by using autonomous particles. Particles operate in parallel without neighborhood information or inter-particle communication by following a set of rules. The rules dictate particle birth, death, and split events which influence the density of particles in different regions of the flow. The authors choose to use split criteria such as λ_2, the curvature of the particle trajectory, and distance to an object in the field. The parallel nature of the system allows interactive visualization while maintaining a smart sampling of the flow. It uses both context and feature particles, with context particles being regularly introduced into the domain to prevent underrepresented regions. These particles are randomly inserted. Feature particles are children of context particles and are introduced when a split event occurs. Split events are determined by a combination of properties such as energy, generation of the parent particle and the local importance measures in the flow.

Algorithms Using Custom Measures

Shen et al. [75] propose the use of fractal dimensions for streamline selection. Measured using the box-counting method, fractal dimensions can provide insight into the complexity of a streamline by considering its space-filling properties [112]. For each grid point in the domain, a scalar value is calculated using the local box counting ratio of each streamline at that location. The scalar grid is used to filter streamlines by fractal dimension and to identify regions containing vortices and turbulence. While this approach does evaluate the space-filling properties of streamlines and highlights regions where this behavior is exhibited, it does not guarantee to capture interesting regions in the flow. Further, they demonstrate the ability to capture a very feature focused set of streamlines and remove any redundant curves.

Summary

Motivated by the desire to highlight particular flow features using streamlines, several algorithms using seed placement and streamline selection have been proposed. Early work by Zockler et al. used the simple concept of associating a degree of interest scalar value for each cell in the domain. Several works measured the entropy or information content of a streamline or a neighborhood in the vector field to determine which streamlines to pick or where to place a seed point respectively. The use of entropy results in streamlines that accurately capture the vector field in regions of turbulent or swirling behavior, i.e., usually regions of interest. Multiple derived scalar values, such as curvature and torsion, are used to highlight streamlines that demonstrate interesting features. These works demonstrate the potential to accurately capture regions of interest without requiring to explicitly extract the flow topology.

In addition to generating streamline visualizations, the seed placement methods using derived scalar fields was applied to particle-based visualization techniques. Derived scalar field techniques offer the benefit of easily computing potential regions of interest in a field and placing seeds or selecting streamlines such that those features are the focus of the visualization.

2.3 Similarity-based

While traditional streamline placement algorithms were based on density distribution or feature extraction to guide streamline placement, these approaches have certain drawbacks. Density-based approaches result in sets of streamlines that contain significant redundancy and feature-based approaches require prior knowledge of the flow features in order to extract them (real-world problems pose challenges). Similarity-based approaches have been gaining popularity in the past decade by serving multiple flow field exploration tasks. They are used to identify a representative set of streamlines in the flow field, support flow exploration by answering streamline similarity queries, and perform streamline clustering to identify sets of similar streamlines or hierarchically group streamlines for a level-of-detail approach. With the goal of performing these tasks, various approaches have been proposed. We categorize these works based on the technique adopted to measure the similarity between streamlines. Further, when considering the similarity between streamlines there are multiple factors to consider such as proximity, shape, scale, and orientation. Certain works aim to only identify similarity in the proximity, while other methods aim to identify similarity invariant to rotation, scale, and orientation. However, the majority of works approach this task by reducing the problem to an evaluation of points along the streamline or segments of the streamline. Early works favored using spatial distance measures to group or identify similar streamlines, while more recently streamline features are being exploited to cluster streamlines and the use of machine learning has been explored as well. We categorize the methods into three classes, i.e., approaches using distance measures between streamlines to evaluate similarity, approaches using features along a streamline to evaluate similarity and the application of machine learning in this domain.

In addition to works identifying a representative set of streamlines in the flow volume, there exists research involving feature specific streamline selection. For example, in an effort to highlight structures of interest in the flow Salzbrunn et al. [113], [114] define streamline and pathline predicates to cluster similar integral curves that satisfy some criteria. Clustering of curves is an approach commonly found in visualizing diffusion tensor imaging (DTI) data [115], [116], [117], [118], [119]. These algorithms generally first define a similarity metric between two curves and then employ a clustering algorithm on the basis of it. With respect to flow visualization, Oeltze et al. [120] evaluate three different kinds of clustering techniques - k-means, agglomerative hierarchical clustering, and spectral clustering to reduce clutter when visualizing streamlines traced from simulated blood flow. Given the relatively high cost of using similarity metrics that often involve pairwise streamline comparison, Shi et al. [121] propose metrics for clustering integral curves that run in linear complexity.

Multiple works have addressed pattern searches to highlight complex distinguishable flow structures in a vector field. To retrieve flow patterns, Ebling et al. [122], [123] and Heiberg et al. [124] employ convolution with dense idealized filter masks, while Schlemmer et al. [125] and Bujack et al. [126] define moment-invariant pattern descriptors for 2D flow fields.

2.3.1 Spatial Distance Techniques

Algorithms Using Distance Measures

Motivated by the shortcomings density and feature guided methods have Chen et al. [48] present the first similarity guided streamline placement algorithm for 2D and 3D steady state flow fields. The proposed algorithm naturally accentuates regions of geometric interest while minimizing streamlines in areas of parallel flow. As a measure of similarity between two streamlines, they define a similarity distance metric which has two influencing factors. The first factor is a translational distance measured as the Euclidean minimum distance between a point on the first streamline under consideration and the second streamline. The second factor is a measure of shape and orientation similarity and is measured over a spatial window. A spatial

window is formed by identifying a predefined number of equally spaced sample points along the curve. Figure 18 shows sample point pairs for two streamline windows.

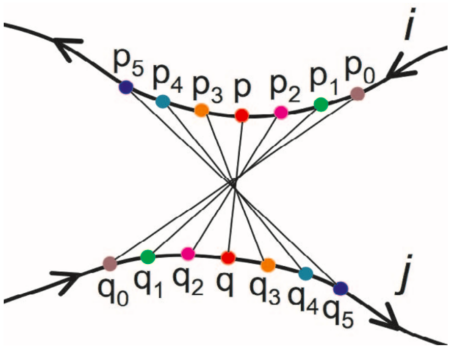

Fig. 18: The figure shows the pairs of points in a spatial window used by Chen et al. [48]. The pairing considers velocity direction along the streamline and thus measures shape and orientation similarity.

They measure the average deviation of sample point pair distances from the center point pair distance, i.e., the translational distance. Starting from a dense set of candidate seed points, they grow streamlines until its similarity distance for a window falls below a pre-specified similarity tolerance. If the streamline length is greater than a minimum length threshold, the streamline is added to the set of selected streamlines. The use of shape and orientation for similarity resulted in 30% better placement compared to only using the translational distance. However, the method was sensitive to the order in which candidate seed points were tested.

With the objective of selecting a small set of representative streamlines, Li et al. [33] propose an iterative streamline placement algorithm for 2D steady state flow fields. The algorithm exploits the spatial coherence in a flow field to achieve a minimal selection of streamlines that convey the underlying flow features. The density of selected streamlines, each of which is integrated for as long as possible, varies to reflect the different degrees of coherence in the field. Employing 2D distance fields that measure the distances from each grid point in the field to nearby streamlines a local and global metric are derived. The local metric measures the direction difference between the vectors of the original field and an approximate field computed from streamlines in the vicinity. The global metric measures streamline dissimilarity by accumulating the local dissimilarity at every integrated point along a streamline trajectory. To place seed points, the algorithm begins by placing a random or central seed point, followed by integrating the streamline and evaluating the local dissimilarity value at each grid point. A streamline is accepted if the local dissimilarity value of the original seed point is greater than a threshold and if the global dissimilarity value of the streamline is greater than a second threshold. The next candidate seed is picked by sorting the grid points into a sorted queue in descending order of the local dissimilarity value. The process ends when no remaining candidate seeds satisfy the dissimilarity threshold requirements. An optimization to the algorithm is reducing the number of candidate seeds considered by eliminating grid points on boundaries and marking cells visited by rejected streamlines, i.e., streamlines with global dissimilarity values below the threshold.

Algorithms Using Mean of Closest Point Distances

Building on the feature highlighting streamline generation technique presented in Section 2.2.2, Yu et al. [68] enable exploration at varying levels of detail via a hierarchical streamline bundling visualization. To calculate the bundles of streamlines, i.e., clusters, they use the mean of closest point distances [116] as a similarity measure. The mean of closest point distances is the mean of Euclidean distances between pairs of points formed by mapping each point of one streamline to the closest point of the other. Beginning with each streamline in a distinct cluster, they successively merge the two most similar streamlines in a bottom-up fashion until a stopping criterion is reached. Clusters are merged using the *single-link* method where the distance between two clusters is the minimum of the distances between all pairs of member streamlines. To represent each cluster of streamlines, the authors choose to consider the union of streamlines along the cluster boundary as opposed to streamlines close to the cluster centroid. The primary rationale being that while both representations capture sources and sinks, only boundary streamlines of a cluster best reveal the saddle together with other boundary streamlines of neighboring clusters. Figure 19 illustrates a result of their clustering visualization.

Fig. 19: Results of the hierarchical bundling visualization algorithm [68]. Left: hierarchical clustering of the flow field using 12 (top) and 70 (bottom) clusters. Right: A representative set of streamlines. Boundary streamlines of clusters are used and they highlight saddles in the flow.

While Yu et al. [68] used the mean of closest point distances as a measure to merge streamlines into a cluster, Tao et al. [69] use the measure to identify redundancy and limit the number of representative streamlines

selected. Measured using a *streamline information* metric, the selection is performed in a view-independent manner by considering the contribution of a streamline to a large set of sample viewpoints. Starting with a random pool of streamlines, they create a matrix containing the probabilities of seeing each streamline from each viewpoint considered. The probability of seeing a streamline is high if it contains a high amount of information in 3D and its 2D projection for a given view preserves the information well. Additionally, they score the shape characteristics of a streamline projection by evaluating each segment of a streamline subsampling and score segments higher if they form a 45 or 135-degree angle to the viewing direction. *Streamline information* represents the degree of dependence between a streamline and the set of viewpoints. A low value indicates a streamline contributes in a balanced manner to a large number of viewpoints, while a high value would indicate a streamline visible in a small set of viewpoints. Streamlines are then sorted into a priority queue, in decreasing order of information conveyed by a streamline for all viewpoints accumulatively, or in increasing order of *streamline information*. However, these streamlines have significant redundancy and likely cause clutter in 3D. They perform pairwise dissimilarity between streamlines using mean of closest point distances to avoid selecting streamlines that are very similar to each other. Further, they define a *viewpoint information* measure to similarly guide viewpoint selection for the chosen streamlines. In comparison to other works, the vector field reconstruction error for this algorithm is lower than both Xu et al. [66] and Marchesin et al. [51]. Han et al. [11] use this method of streamline selection in situ to save a compressed representative set of streamlines for post hoc flow field reconstruction and analysis.

Opacity adaption is a technique used to manage occlusion and cluttering of streamlines in a 3D field. However, it can result in the loss of spatial perception when streamlines are faded out to reveal interesting regions of the field. To retain the perception of spatial relationships between streamlines, Kanzler et al. [56] select a set of streamline such that the screen-space density of the streamlines is locally adapted to the importance of the streamlines. Figure 20 shows an example comparing the use of opacity optimization and adaption of screen-space density. The algorithm requires the computation of a fully balanced line hierarchy as a preprocess to facilitate the uniform removal of streamlines in the domain to obtain desired density at run time. The mean of closest point distances is computed for all pairs of streamlines and is used to define a fully connected distance graph. A minimum cost perfect matching algorithm is recursively used to identify pairs by minimizing the sum of all included edge weights, i.e., the similarity measure. Single linkage is used to merge clusters since it produces spatial coherent merging of clusters. Streamlines are then selected by using visibility values which are influenced by an importance measure, such as curvature, measured along the streamline and the occlusion caused by the streamline to other potentially more important lines. Visibility thresholds are assigned to lines in the hierarchy based on the level at which the line is the representative for its cluster. Further, the visibility values are then used to locally control the line density. While this method improves the spatial perception of the visualization, it incurs a long preprocessing time to build a balanced line hierarchy.

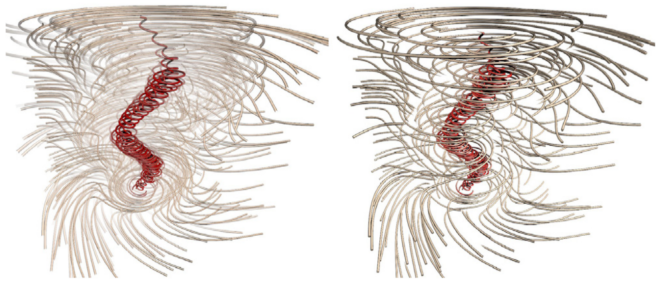

Fig. 20: A result of the approach adopted by Kanzler et al. to preserve spatial perception [56]. Left: a visualization using opacity adaption. Right: streamline selection performed using a balanced line hierarchy and visibility values based on importance.

Summary

Spatial distance measures were used in the form of distance fields and as a measure of similarity between two streamlines involving both seed placement techniques and streamline selection. Chen et al. were the first to evaluate the similarity in shape and orientation between two streamlines across a window and showed the improvement in streamline selection when the similarity of shape and orientation are considered. Li et al. introduced a streamline placement algorithm based on dissimilarity, which iteratively picked a dissimilar streamline to already chosen streamlines. Other works which were all based on clustering, adopt a streamline selection task by first forming distinct clusters and then choosing representatives at various levels of the hierarchy. To improve on the balance of the hierarchy, methods varied between variants of the agglomerative hierarchical clustering scheme and minimum cost perfect matching approaches. Clustering before selection of representative streamlines allows options for obtaining the desired streamline distribution, such as retaining a few occluding streamlines for spatial perception.

While these techniques are useful for selecting a representative set of streamlines, the use of similarity-based methods extends beyond this. Similarity-based methods have enabled flow exploration via queries. Methods using spatial distance measures are limited to comparing with streamlines in the proximity. This prevents similarity query searches from identifying similar streamlines in other regions of the domain, or streamlines of different scale and orientation.

2.3.2 Feature Attributes

Feature attributes have been extensively used to evaluate the similarity between streamlines. A common use case is to identify all streamlines similar to a given streamline. Distance based similarity measures primarily account for proximity and are sensitive to rotation, translation, and scaling. Feature attributes measured along the streamline provide a metric to evaluate a streamline similarity in a proximity, size, and orientation insensitive manner.

Algorithms Using Point Sampled Features

Wei et al. [59] propose a 3D streamline selection technique based on identifying similar streamlines to a user sketched streamline. The user sketch is a 3D curve whose 2D projection is used as the input to the algorithm. They approximate the sketched curve and the streamlines using a arc length parameterized cubic B-spline and sample the curvature at equal arc length intervals along the curve. Using a feature vector constructed by concatenating the curvature at sampled points, they employ a *string matching* approach to find similar streamlines, with the difference between two vectors being measured using the *edit distance* [127]. They identify the most similar 3D streamline and use it as a reference for clustering. All streamlines similar to the reference streamlines are selected as the result of the streamline query. For comparison between 3D streamlines the feature vector is constructed using both curvature and torsion. In addition to responding to streamline similarity queries, Wei et al. propose to choose cluster representatives of a agglomerative hierarchical clustering scheme on the basis of view-dependent quality. Viewpoint quality of a streamline is computed by accumulating the winding angle of the 2D projection of the streamline.

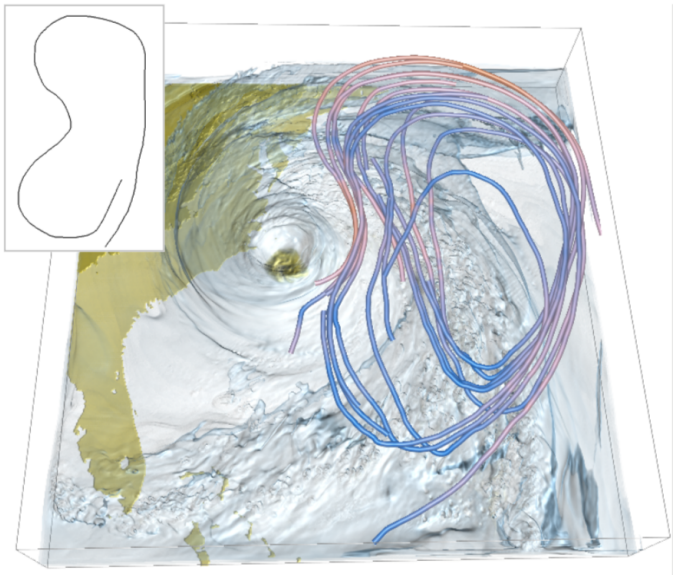

Fig. 21: Wei et al. presented a similarty-based approach to identify streamlines similar to a user-sketched query streamline [59].

Zheng et al. [34] present a streamline selection algorithm for 2D flow fields that uses streamline feature classification, similarity and entropy. Streamlines are classified on the basis of the feature they highlight, i.e., a vortex, source, sink, or saddle, and are also prioritized in that order when they capture more than one feature. In addition to feature type, each streamline has a feature position, i.e., the vortex center for a vortex streamline, the critical point for a source-sink streamline, or the point of highest entropy for a saddle streamline. Streamlines are then iteratively clustered on the basis of a combination of feature type and proximity of the feature position. They define a similarity metric that uses a combination of both geometric shape properties and proximity. Curvature and accumulated angle are used as geometric shape properties and sample points on two streamlines are mapped using dynamic time warping. For a proximity evaluation mean of closest point distances is employed. They first select a set of streamlines by identifying a streamline from each feature subset with the highest entropy accumulated along the points of the streamline. Streamlines within each feature subset that are dissimilar to the previously selected streamlines are picked next. The last step involves, selecting more streamlines by considering both streamline similarity to previously chosen streamlines and streamline entropy until a desired number of streamlines are selected. The algorithm is focused on limiting the number of redundant streamlines and is capable of generating a placement qualitatively equivalent to the work by Yu et al. [68] for a 2D flow field.

Algorithms Using Segmentation

To tackle the high computational expense of distance based similarity measures which involve performing large numbers of Euclidean distance tests, McLoughlin et al. [81] propose to measure streamline similarity by first computing an integral curve specific signature. The signature is computed by segmenting an integral curve and using a set of curve-based attributes, namely, curvature, torsion, and tortuosity, to describe the integral curve per unit length of the curve. The χ^2 test is used to measure dissimilarity between streamlines and is performed for all streamline pairs generating a similarity matrix which enables fast lookup for the clustering process. Additionally, given the streamline signature is streamline proximity independent, a Euclidean distance measure parameter can be used to supplement the χ^2 similarity measure. Addressing issues of alignment of bins when comparing a pair of streamlines, the authors limit placement of initial seed points to rakes orthogonal to the local flow and consider a hierarchical signature for each streamline. The algorithm was demonstrated to be orders-of-magnitude faster than the approach by Chen et al. [48].

Chen et al. [128] adopt an entropy-guided seed placement strategy to generate an initial set of streamlines. Streamlines are then clustered using a two-stage k-means clustering algorithm. The first stage only considers the start, middle, and end point of a streamline for clustering. Each cluster after the first stage is further subdivided into clusters by considering the linear and angular entropy of streamline segments. The use of the two stage k-means algorithm is preferred to single-linkage clustering with mean of closest point distances due to the quadratic computational complexity of the latter. The authors use additional visualization (directional information) cues to aid flow field exploration.

Lu et al. [70] propose a similarity measure based on the statistical distribution of measurements along a streamline trajectory. Given the approach is based on distributions, compared to previous approaches the similarity measure is less sensitive to length, spatial location, and orientation. A streamline is recursively segmented until a minimum length threshold is reached or a segment cannot be split into two segments which are dissimilar enough. Further, a 1D histogram is constructed to represent every segment, and a

2D histogram to represent the entire curve. The use of the 2D histogram is to essentially capture the order of the segments and avoid two dissimilar streamlines with similar feature distributions appearing similar. Given streamlines may be represented by a varying number of segments, a mapping between two sets of segments is performed by using Dynamic Time Warping [129]. The difference between two histograms is measured using earth mover's distance (EMD). The authors use curvature, torsion, and curl as measures along a streamline to demonstrate the distribution-based approach. Their proposed agglomerative hierarchical clustering scheme, in addition to using a distance measure (for example, *single-linkage*), they use a balance parameter which accounts for number of streamlines in a cluster and can force smaller clusters to merge early in the process to produce a more balanced tree. The algorithm was demonstrated to be much faster than using distance measures to identify similarity.

Li et al. [73] propose to use a similar feature vector description approach to measure streamline similarity. To address the issue of dissimilar streamlines having very similar feature distributions, they use a feature descriptor that encodes the spatial relations among the features. The encoding used mitigates the need to segment the streamline and find mappings between two streamlines during similarity evaluation. Beside using local streamline metrics like curvature and torsion, Li et al. use global geometric properties of tortuosity and velocity direction entropy to describe a streamline. A weighted Manhattan distance between constructed feature vectors of two streamlines measures the similarity between them. Following a pairwise similarity measure evaluation between all streamlines, *affinity propagation* [130] is used to cluster streamlines and form a hierarchy. The affinity propagation algorithm accepts the measured similarity values, used as preference values for each data point, as input and simultaneously considers all the data points as possible cluster centers. The algorithm then exchanges real-valued messages between data points until it converges to produce a set of cluster centers of high quality.

FlowString is a framework for partial streamline matching proposed by Tao et al. [72] that models streamlines as strings. Streamlines are first resampled on the basis of winding angle, using a threshold small enough to capture relatively simple patterns of the streamline segment between neighboring sample points. All sample points are then evaluated pairwise for a dissimilarity measure, the *Procrustes distance*(a metric to quantify dissimilarity for 3D shapes and is extensively used in biological morphometrics), followed by applying *affinity propagation* for clustering using a GPU. The resultant clusters after two levels of affinity propagation serve as the local shapes for the data set. The local shapes are used as characters, which together form an alphabet, using which words can be formed by concatenating characters together. A suffix tree, built to represent all strings, is used for efficient exact search, and approximate searches are achieved using dynamic programming. This work was the first to pursue labeling and classification of streamline segments.

Summary

Use of features along a streamline allowed for comparison between streamlines in a proximity, scale and orientation invariant manner. Streamlines were either identified by the feature attributes of points along the curve or by attributes of segments of the curve. The use of attributes to describe segments of a streamlines reduced the large number of distance tests performed when comparing similarity using points. Initially, similarity between streamlines was limited to those generated from a rake of seed points. Later works considered distribution of feature attributes and the order in which they appear to identify streamlines that are scale, proximity, and orientation invariant. The affine propagation method of clustering has found application in clustering similar segments of streamlines when tackling multiple tasks. In addition to the use of accelerators for clustering, the use of segmentation as opposed to performing large numbers of Euclidean distance checks has resulted in significant speed ups. However, these similarity measures are being performed on relatively small number of curves and extending these techniques to scale by using more computationally efficient similarity or dissimilarty measures is a current research area.

2.3.3 Machine Learning

The application of machine learning techniques to scientific visualization problems has been a recent development in the field. With respect to flow visualization and specifically the use of streamlines there has been recent activity. Li et al. [74] improve on their previous approach of similarity calculation using machine learning techniques. In previous works, similarity measures calculated via segmentation do not account for human perception, i.e., human-judged similarity may not match a weighted Manhatten distance or some such distance measure.

Algorithms Using SVM For Segmentation

Streamline segmentation has growing importance given their application in identifying the similarity between streamlines and assisting in selecting a representative set of streamlines. However, current measures of segmentation and similarity measurement do not account for human perception or what a human considers important. Li et al. [74] adopt a user-guided approach that employs a binary support vector machine (SVM) to perform streamline segmentation. The approach begins by first generating a pool of random streamlines, followed by the use of affinity propagation for clustering based on a similarity metric that uses curvature and torsion 1D histograms. Users are then required to choose segmentation points along the set of representative streamlines from each cluster. For every segmentation point, the algorithm computes a feature vector comprising velocity direction ratio, tortuosity ratio, the curvature and torsion histograms, and the volume ratio of minimum bounding ellipsoids using varying neighborhood sizes. User selected segmentation points are positive training samples, while all non-segmentation points are negative training samples used to train an SVM classifier. The streamline segmentation process is carried out for the remaining streamlines using the classifier. If a group of nearby points is selected as

segmentation candidates, a post-processing step chooses the point with the smallest ratio of minimum bounding ellipsoids as the final segmentation point. The algorithm presented by Li et al. is the first to use supervised machine learning for streamline segmentation. They demonstrated superior segmentation and feature capturing in comparison to previous similarity-based methods that used segmentation.

Algorithms Using DNN For Feature Description

Han et al. [76] use a deep neural net, named FlowNet, to identify a representative set of streamlines for a given flow field. They utilize an autoencoder which features both convolutional and fully-connected layers enabling the network to learn a complex data representation by using both local and non-linear combinations of neurons. FlowNet accepts voxelized and downsampled representations of streamlines as input to learn features by non-linearly mapping each representation to a feature descriptor of 1024 dimensions. They employ a binary cross-entropy loss function to train FlowNet. To explore the feature descriptors generated they apply t-SNE [131] for dimensionality reduction, followed by interactively tuning the parameters of their clustering method DBSCAN [132] to find a suitable number of clusters or set the minimum number of samples in a cluster. A representative streamline is then identified as one whose sum of Euclidean distance to all other points in the cluster is the minimum. Han et al. don't explicitly use any streamline attributes, but instead, are the first to employ a deep neural net to calculate flow features before clustering and selection. They demonstrate streamline selection which results in lower vector field reconstruction error compared to Tao et al. [69] and Xu et al. [66]. However, training the network can require days to complete and is a current shortcoming of this approach.

Summary

The use of machine learning for scientific visualization tasks is increasing in recent years. The first use of an SVM for streamline segmentation is by Li et al. who use it to segment streamlines in a cluster of similar streamlines, by a user choosing segmentation points along the streamline. The approach allowed the user to specify what is a considered important feature by selecting segmentation points. Han et al. use machine learning to learn feature attributes of streamlines automatically and then cluster them based on similarity to identify a representative set. The drawbacks of these methods as of now are they need to subsample the data set in order for it to be feasible to process and they still require long run times. Further, the features learned are data set specific and require each data set to be processed. Future efforts can aim to build a large database to train neural networks to identify flow features. Thus, there is room for further research with respect to the application of machine learning techniques to unsteady flow visualization and overall computational improvements.

3 Manual Techniques

Manual placement or specification of initial seed positions is a common first step when using streamlines to study a flow field. Interactive flow visualization techniques were introduced two decades ago and have since evolved to give the user varying degrees of control of the generated visualization. While several interactive flow visualization techniques exist, in this section, we limit our study to manual techniques involving tools for placement of seeds, density control, and the use of domain information to do the same.

Use of Interactive Seed Placement and Streamline Control Tools

The virtual windtunnel project [77] was an immersive virtual reality-based system used for the investigation of airflow around a Space Shuttle. In this work, Bryson et al. described the use of a hand position sensitive glove controller for injecting particles into a three-dimensional unsteady flow. To study areas of interest in the flow, such as boundary layers and turbulent regions, they interactively select the locations of seed points, numerically integrate their trajectories, followed by visualizing and manipulating the resultant graphics objects. In addition to rapid placement of seed points, their environment supported repositioning, the grouping of seeds as a rake, and deletion of existing seed points using hand gestures. Hardware limitations at the time meant spatial subsampling of the vector field was required for interactivity or use of a supercomputer and dedicated graphics resources.

Schulz et al. [78] aimed to improve the interactivity of a virtual reality-based exploration system. PowerFLOW, a lattice-based flow code with locally refined cartesian grids, used for aerodynamic simulations in car body development required particle tracing to account for collisions with the car body. They proposed specific data structures and interpolation techniques for fast particle tracing with the initial positions of seeds, specified using a freely movable probe similar to Bryson et al. [77], aligned on a rake or inside a cube. Laramee et al. [63] propose the use of a manual

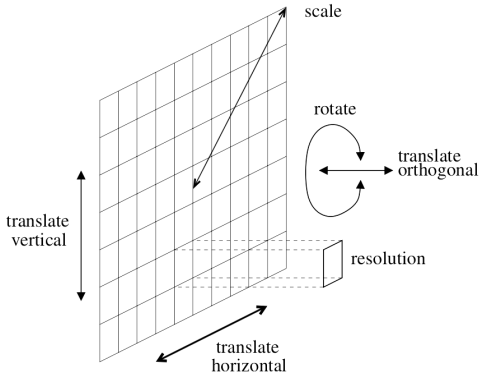

Fig. 22: Laramee et al. propose the usage of a seeding plane to manually place seed points in a 3D flow field [63]. The plane can be oriented and scaled as needed.

seeding tool which allows for six degrees of freedom. The seed placement tool, shown in Figure 22, is a two dimensional seeding plane grid. Initial seed locations are set at the grid points before streamlines are calculated. Besides offering grid resolution control, the seeding plane that can be translated, rotated, and scaled enabling convenient seed placement options.

Laramee [62] proposed a system, named Streamrunner, which attempted to tackle previously unaddressed problems of occlusion, lack of directional and depth cues when interactively using streamlines in 3D flow fields. Streamrunner gave the user control over placement of seeds and then the evolution of streamlines from the time they are seeds until they reach full length. This provided users with a sense of direction of flow and depth when observing the growth of the streamlines. Additionally, it allowed the user to manually control the streamline seeding density in the flow field.

Use of Domain-Information for Seed Placement

Manual placement of seed points or rakes is most suitable when the nature of the flow field is known and scientists can generate informative visualizations by intelligently placing seed points. For example, Figure 23 illustrates seed points placed in multiple rakes to capture the behavior of an unsteady state flow field containing a Tornado vortex [133]. Alternatively, seed points may be manually placed near local maxima of an interesting scalar derived from the vector field or near critical points of the vector field topology [134]. Several flow visualization works adopt this approach.

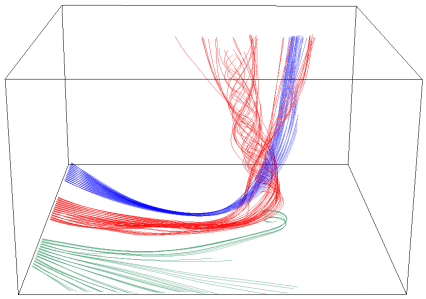

Fig. 23: Visualization of a Tornado vortex using pathlines, with initial seeds placed along multiple rakes (left-bottom of the image)

For certain applications, knowledge of moving objects in the domain or specific component design assists in deciding seed point locations. Engineers are often required to evaluate the pattern of flow, such as a swirl or tumble flow, in the combustion chamber of an automobile in order to achieve efficient and stable combustion. Laramee et al. [63] strategically place a seeding plane near the intake ports of a combustion chamber from where fluid enters to evaluate swirl flow. They manually place multiple seeding planes and limit the length of the generated streamlines in the combustion chamber to capture tumble motion. In another study involving the complex geometry of an automotive engine cooling jacket, Laramee et al. [64] generate seed points at the inlet of the cooling jacket. To maintain seed density they adopt a scheme similar to that of Bauer et al. [85] and allow particles to travel along integral curves until they hit a boundary or leave through the outlet. Interactively exploring the cooling jacket proved tedious given the rapid visual clutter created by complicated twisted paths and the difficulty in identifying recirculation zones unless particles reached those regions.

In a biology-inspired CFD simulation, Koehler presents a visual flow analysis of insect flight, with the domain containing multiple dynamically deforming flapping wings of a dragonfly [80]. With the objective of capturing the formation, attachment, and shedding of leading-edge vortices and identifying unsteady lift production mechanisms which could be applied to developing efficient micro air vehicles, the authors present a novel seed placement method. Traditional methods of using static seed points suffer in situations where there are immersed boundaries in the flow field. The proposed method is based on the premise that interesting flow phenomena generally occur near and move with the wings.

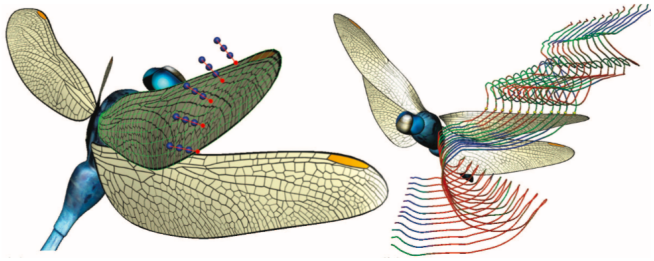

Fig. 24: Placement of seeds (left-hand side image) and the corresponding generated seed curves (right-hand side image) [80]

Seeds are bound in the direction of the vertex normal of user selected points near the surface of the immersed objects (for this application the wings). The user is given control of the seed density and how far in the normal direction seeds are placed. Seed curves are obtained by connecting points at neighboring time steps that are the same distance in the normal direction of the same point on the wing mesh. Seed curves are then color-mapped to a scalar of interest such as velocity, vorticity or λ_2. The user can then choose seed curves that are informative and be used further to generate various integration-based flow lines. Figure 24 illustrates the seed placement and the generated seed curves.

4 Conclusion

In this report, we presented a survey of strategies for seed placement and streamline selection. The extensive use of streamlines for flow visualization has resulted in several methods and suggested approaches regarding how to use them to explore a flow field. Our classification and categorization of these algorithms resulted in 18 strategies for automatic techniques and 2 strategies for manual techniques. Further, we show a grouping of works based on the context, dimension, and state of flow they are applied to. In addition to existing qualitative and quantitative comparisons, we evaluate techniques to compare and relate them along three axes, namely, redundancy, regions of interest and computation (Figure 25). Our evaluation assists in identifying viable and suitable approaches for in situ methods to adopt or build on.

Future work will focus on utilizing the advantages of feature and similarity-based approaches within the in situ context to guide particle placement to extract or generate a smart sampling of the flow field. We believe there is potential in leveraging accelerators to reduce the burden on the simulation execution time. To a large extent, the problem of selecting a representative set of streamlines has

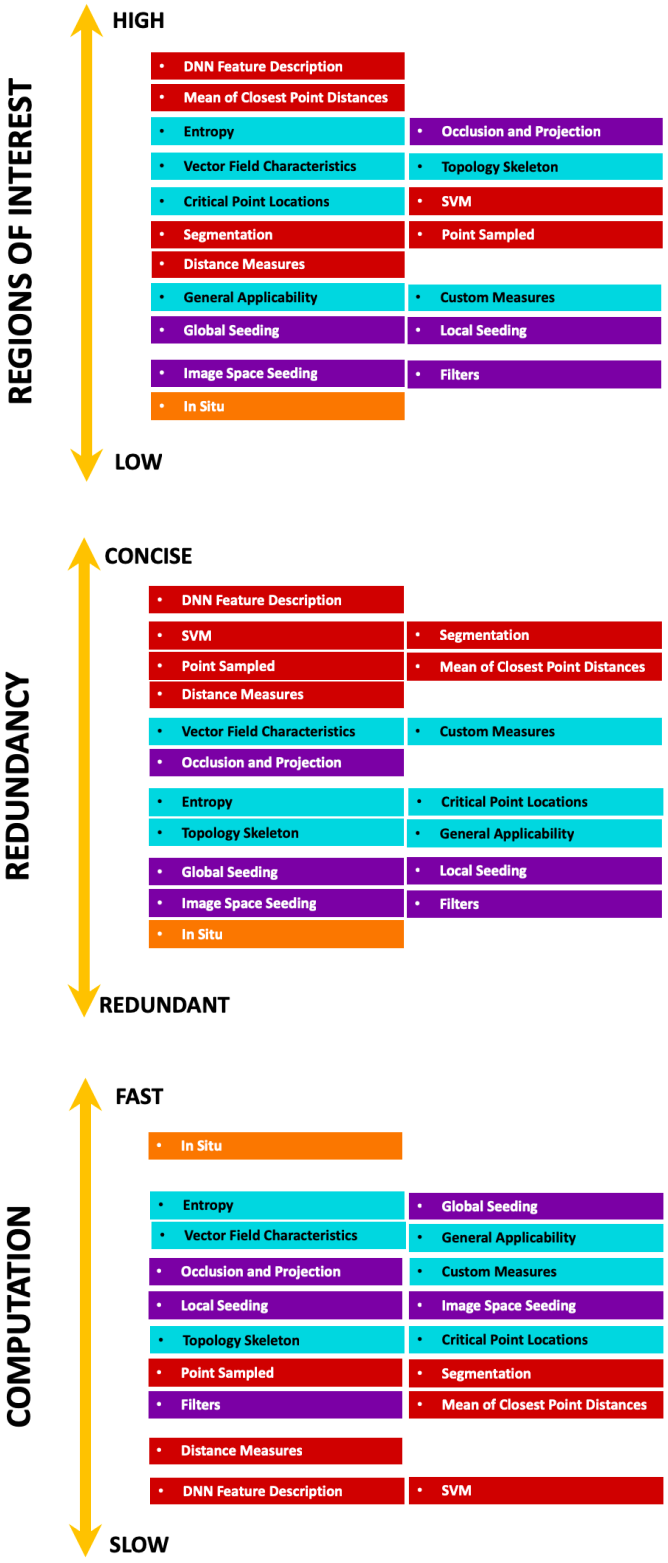

Fig. 25: Using axes representing regions of interest, redundancy, and computation cost to compare and relate strategies for seed placement and streamline selection.

been solved under the post hoc paradigm. Application of these techniques to pathlines and under in situ constraints will present new challenges to flow visualization and information extraction.

REFERENCES

[1] V. Verma, D. Kao, and A. Pang, "A flow-guided streamline seeding strategy," in *Proceedings of the conference on Visualization'00*. IEEE Computer Society Press, 2000, pp. 163–170.

[2] N. Fabian, K. Moreland, D. Thompson, A. C. Bauer, P. Marion, B. Gevecik, M. Rasquin, and K. E. Jansen, "The paraview coprocessing library: A scalable, general purpose in situ visualization library," in *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*. IEEE, 2011, pp. 89–96.

[3] B. Whitlock, J. M. Favre, and J. S. Meredith, "Parallel in situ coupling of simulation with a fully featured visualization system," in *Proceedings of the 11th Eurographics Conference on Parallel Graphics and Visualization*, ser. EGPGV '11. Eurographics Association, 2011, pp. 101–109. [Online]. Available: http://dx.doi.org/10.2312/EGPGV/EGPGV11/101-109

[4] K. Moreland, R. Oldfield, P. Marion, S. Jourdain, N. Podhorszki, V. Vishwanath, N. Fabian, C. Docan, M. Parashar, M. Hereld *et al.*, "Examples of in transit visualization," in *Proceedings of the 2nd international workshop on Petascal data analytics: challenges and opportunities*. ACM, 2011, pp. 1–6.

[5] V. Vishwanath, M. Hereld, and M. E. Papka, "Toward simulation-time data analysis and i/o acceleration on leadership-class systems," in *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*. IEEE, 2011, pp. 9–14.

[6] J. Ahrens, S. Jourdain, P. O'Leary, J. Patchett, D. H. Rogers, and M. Petersen, "An image-based approach to extreme scale in situ visualization and analysis," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Press, 2014, pp. 424–434.

[7] H. Yu, C. Wang, R. W. Grout, J. H. Chen, and K.-L. Ma, "In situ visualization for large-scale combustion simulations," *IEEE computer graphics and applications*, vol. 30, no. 3, pp. 45–57, 2010.

[8] A. C. Bauer, H. Abbasi, J. Ahrens, H. Childs, B. Geveci, S. Klasky, K. Moreland, P. O'Leary, V. Vishwanath, B. Whitlock *et al.*, "In situ methods, infrastructures, and applications on high performance computing platforms," in *Computer Graphics Forum*, vol. 35, no. 3. Wiley Online Library, 2016, pp. 577–597.

[9] M. Vetter and S. Olbrich, "Development and integration of an in-situ framework for flow visualization of large-scale, unsteady phenomena in icon," *Supercomputing Frontiers and Innovations*, vol. 4, no. 3, pp. 55–67, 2017.

[10] A. Agranovsky, D. Camp, C. Garth, E. W. Bethel, K. I. Joy, and H. Childs, "Improved post hoc flow analysis via lagrangian representations," in *Large Data Analysis and Visualization (LDAV), 2014 IEEE 4th Symposium on*. IEEE, 2014, pp. 67–75.

[11] J. Han, J. Tao, H. Zheng, H. Guo, D. Z. Chen, and C. Wang, "Flow field reduction via reconstructing vector data from 3d streamlines using deep learning."

[12] M. Hlawatsch, F. Sadlo, and D. Weiskopf, "Hierarchical line integration," *IEEE transactions on visualization and computer graphics*, vol. 17, no. 8, pp. 1148–1163, 2011.

[13] A. Agranovsky, C. Garth, and K. I. Joy, "Extracting flow structures using sparse particles." in *VMV*, 2011, pp. 153–160.

[14] J. Chandler, H. Obermaier, and K. I. Joy, "Interpolation-based pathline tracing in particle-based flow visualization," *IEEE transactions on visualization and computer graphics*, vol. 21, no. 1, pp. 68–80, 2015.

[15] S. Sane, R. Bujack, and H. Childs, "Revisiting the Evaluation of In Situ Lagrangian Analysis," in *Eurographics Symposium on Parallel Graphics and Visualization*, H. Childs and F. Cucchietti, Eds. The Eurographics Association, 2018.

[16] J. Chandler, R. Bujack, and K. I. Joy, "Analysis of error in interpolation-based pathline tracing," in *Proceedings of the Eurographics/IEEE VGTC Conference on Visualization: Short Papers*. Eurographics Association, 2016, pp. 1–5.

[17] R. Bujack and K. I. Joy, "Lagrangian representations of flow fields with parameter curves," in *Large Data Analysis and Visualization (LDAV), 2015 IEEE 5th Symposium on*. IEEE, 2015, pp. 41–48.

[18] M. Hummel, R. Bujack, K. I. Joy, and C. Garth, "Error estimates for lagrangian flow field representations," in *Proceedings of the Eurographics/IEEE VGTC Conference on Visualization: Short Papers*. Eurographics Association, 2016, pp. 7–11.

[19] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch, "The state of the art in flow visualisation: Feature extraction and tracking," in *Computer Graphics Forum*, vol. 22, no. 4. Wiley Online Library, 2003, pp. 775–792.

[20] R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, and D. Weiskopf, "The state of the art in flow visualization: Dense and texture-based techniques," in *Computer Graphics Forum*, vol. 23, no. 2. Wiley Online Library, 2004, pp. 203–221.

[21] R. S. Laramee, H. Hauser, L. Zhao, and F. H. Post, "Topology-based flow visualization, the state of the art," in *Topology-based methods in visualization*. Springer, 2007, pp. 1–19.

[22] T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen, "Over Two Decades of Integration-Based, Geometric Flow Visualization," in *EG 2009 - State of the Art Reports*, 2009, pp. 73–92.

[23] A. Pobitzer, R. Peikert, R. Fuchs, B. Schindler, A. Kuhn, H. Theisel, K. Matković, and H. Hauser, "The state of the art in topology-based visualization of unsteady flow," in *Computer Graphics Forum*, vol. 30, no. 6. Wiley Online Library, 2011, pp. 1789–1811.

[24] B. Jobard and W. Lefer, "Creating evenly-spaced streamlines of arbitrary density," in *Visualization in Scientific Computing?97*. Springer, 1997, pp. 43–55.

[25] ——, "Multiresolution flow visualization," 2001.

[26] A. Mebarki, P. Alliez, and O. Devillers, "Farthest point seeding for efficient placement of streamlines," in *Visualization, 2005. VIS 05. IEEE*. IEEE, 2005, pp. 479–486.

[27] G. Chen, K. Mischaikow, R. S. Laramee, P. Pilarczyk, and E. Zhang, "Vector field editing and periodic orbit extraction using morse decomposition," *IEEE Transactions on Visualization & Computer Graphics*, no. 4, pp. 769–785, 2007.

[28] W. Zhang and J. Deng, "Topology-driven streamline seeding for 2d vector field visualization," in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*. IEEE, 2009, pp. 4901–4905.

[29] K. Wu, Z. Liu, S. Zhang, and R. J. Moorhead II, "Topology-aware evenly spaced streamline placement," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 5, pp. 791–801, 2010.

[30] W. Zhang, B. Sun, and Y. Wang, "A streamline placement method highlighting flow field topology," in *Computational Intelligence and Security (CIS), 2010 International Conference on*. IEEE, 2010, pp. 238–242.

[31] W. Zhang, M. Zhang, and B. Sun, "Multiresolution streamline placement for 2d flow fields," in *Computational Intelligence and Security (CIS), 2011 Seventh International Conference on*. IEEE, 2011, pp. 1174–1178.

[32] W. Zhang, Y. Wang, J. Zhan, B. Liu, and J. Ning, "Parallel streamline placement for 2d flow fields," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 7, pp. 1185–1198, 2013.

[33] L. Li, H.-H. Hsieh, and H.-W. Shen, "Illustrative streamline placement and visualization," in *Visualization Symposium, 2008. PacificVIS'08. IEEE Pacific*. IEEE, 2008, pp. 79–86.

[34] L. Zheng, W. Wang, and S. Li, "Feature-based streamline selection method for 2d flow fields," in *Computer-Aided Design and Computer Graphics (CAD/Graphics), 2015 14th International Conference on*. IEEE, 2015, pp. 129–136.

[35] G. Turk and D. Banks, "Image-guided streamline placement," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 1996, pp. 453–460.

[36] C. Luo, I. Safa, and Y. Wang, "Feature-aware streamline generation of planar vector fields via topological methods," *Computers & Graphics*, vol. 36, no. 6, pp. 754–766, 2012.

[37] B. Jobard and W. Lefer, "Unsteady flow visualization by animating evenly-spaced streamlines," in *Computer Graphics Forum*, vol. 19, no. 3. Wiley Online Library, 2000, pp. 31–39.

[38] Z. Ding, X. Zhang, W. Chen, X. Tricoche, D. Peng, and Q. Peng, "Coherent streamline generation for 2-d vector fields," *Tsinghua Science and Technology*, vol. 17, no. 4, pp. 463–470, 2012.

[39] N. Max, R. Crawfis, and C. Grant, "Visualizing 3d velocity fields near contour surfaces," in *Proceedings of the conference on Visualization'94*. IEEE Computer Society Press, 1994, pp. 248–255.

[40] O. Rosanwo, C. Petz, S. Prohaska, H.-C. Hege, and I. Hotz, "Dual streamline seeding," in *Visualization Symposium, 2009. PacificVis' 09. IEEE Pacific*. IEEE, 2009, pp. 9–16.

[41] X. Mao, Y. Hatanaka, H. Higashida, and A. Imamiya, "Image-guided streamline placement on curvilinear grid surfaces," in *Visualization'98. Proceedings*. IEEE, 1998, pp. 135–142.

[42] Z. Liu and R. J. Moorhead, "Interactive view-driven evenly spaced streamline placement," in *Visualization and Data Analysis 2008*, vol. 6809. International Society for Optics and Photonics, 2008, p. 68090A.

[43] B. Spencer, R. S. Laramee, G. Chen, and E. Zhang, "Evenly spaced streamlines for surfaces: An image-based approach," in *Computer Graphics Forum*, vol. 28, no. 6. Wiley Online Library, 2009, pp. 1618–1631.

[44] O. Mattausch, T. Theußl, H. Hauser, and E. Gröller, "Strategies for interactive exploration of 3d flow using evenly-spaced illuminated streamlines," in *Proceedings of the 19th spring conference on Computer graphics*. ACM, 2003, pp. 213–222.

[45] X. Ye, D. Kao, and A. Pang, "Strategy for seeding 3d streamlines," in *Visualization, 2005. VIS 05. IEEE*. IEEE, 2005, pp. 471–478.

[46] Z. Liu, R. Moorhead, and J. Groner, "An advanced evenly-spaced streamline placement algorithm," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 965–972, 2006.

[47] Z. Liu and R. J. Moorhead II, "Robust loop detection for interactively placing evenly spaced streamlines," *Computing in Science & Engineering*, vol. 9, no. 4, pp. 86–91, 2007.

[48] Y. Chen, J. Cohen, and J. Krolik, "Similarity-guided streamline placement with error evaluation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1448–1455, 2007.

[49] W. Zhang, J. Ning, M. Zhang, Y. Pei, B. Liu, and B. Sun, "Multiresolution streamline placement based on control grids," *Integrated Computer-Aided Engineering*, vol. 21, no. 1, pp. 47–57, 2014.

[50] L. Li and H.-W. Shen, "Image-based streamline generation and rendering," *IEEE Transactions on Visualization & Computer Graphics*, no. 3, pp. 630–640, 2007.

[51] S. Marchesin, C.-K. Chen, C. Ho, and K.-L. Ma, "View-dependent streamlines for 3d vector fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1578–1586, 2010.

[52] T.-Y. Lee, O. Mishchenko, H.-W. Shen, and R. Crawfis, "View point evaluation and streamline filtering for flow visualization," in *Visualization Symposium (PacificVis), 2011 IEEE Pacific*. IEEE, 2011, pp. 83–90.

[53] T. Günther, K. Bürger, R. Westermann, and H. Theisel, "A view-dependent and inter-frame coherent visualization of integral lines using screen contribution." in *VMV*, 2011, pp. 215–222.

[54] T. Günther, C. Rössl, and H. Theisel, "Opacity optimization for 3d line fields," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, p. 120, 2013.

[55] J. Ma, C. Wang, and C.-K. Shene, "Coherent view-dependent streamline selection for importance-driven flow visualization," in *Visualization and Data Analysis 2013*, vol. 8654. International Society for Optics and Photonics, 2013, p. 865407.

[56] M. Kanzler, F. Ferstl, and R. Westermann, "Line density control in screen-space via balanced line hierarchies," *Computers & Graphics*, vol. 61, pp. 29–39, 2016.

[57] T. Annen, H. Theisel, C. Rössl, G. Ziegler, and H.-P. Seidel, "Vector field contours," in *Proceedings of Graphics Interface 2008*. Canadian Information Processing Society, 2008, pp. 97–105.

[58] S. Furuya and T. Itoh, "A streamline selection technique for integrated scalar and vector visualization," *Vis Š08: IEEE Visualization Poster Session*, vol. 2, no. 4, 2008.

[59] J. Wei, C. Wang, H. Yu, and K.-L. Ma, "A sketch-based interface for classifying and visualizing vector fields," in *Visualization Symposium (PacificVis), 2010 IEEE Pacific*. IEEE, 2010, pp. 129–136.

[60] M. Zockler, D. Stalling, and H.-C. Hege, "Interactive visualization of 3d-vector fields using illuminated stream lines," in *Visualization'96. Proceedings.* IEEE, 1996, pp. 107–113.

[61] L. A. Treinish, "Multi-resolution visualization techniques for nested weather models," in *Proceedings of the conference on Visualization'00*. IEEE Computer Society Press, 2000, pp. 513–516.

[62] R. S. Laramee, "Interactive 3d flow visualization using a streamrunner," in *Conference on Human Factors in Computing Systems: CHI'02 extended abstracts on Human factors in computing systems*, vol. 20, no. 25, 2002, pp. 804–805.

[63] R. S. Laramee, D. Weiskopf, J. Schneider, and H. Hauser, "Investigating swirl and tumble flow with a comparison of visualization techniques," in *Visualization, 2004. IEEE*. IEEE, 2004, pp. 51–58.

[64] R. S. Laramee, C. Garth, H. Doleisch, J. Schneider, H. Hauser, and H. Hagen, "Visual analysis and exploration of fluid flow in a cooling jacket," in *Visualization, 2005. VIS 05. IEEE*. IEEE, 2005, pp. 623–630.

[65] M. Schlemmer, I. Hotz, B. Hamann, F. Morr, and H. Hagen, "Priority streamlines: A context-based visualization of flow fields." in *EuroVis*, 2007, pp. 227–234.

[66] L. Xu, T.-Y. Lee, and H.-W. Shen, "An information-theoretic framework for flow visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1216–1224, 2010.

[67] Y. Wang, W. Zhang, and J. Ning, "Streamline-based visualization of 3d explosion fields," in *Computational Intelligence and Security (CIS), 2011 Seventh International Conference on*. IEEE, 2011, pp. 1224–1228.

[68] H. Yu, C. Wang, C.-K. Shene, and J. H. Chen, "Hierarchical streamline bundles," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 8, pp. 1353–1367, 2012.

[69] J. Tao, J. Ma, C. Wang, and C.-K. Shene, "A unified approach to streamline selection and viewpoint selection for 3d flow visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 3, pp. 393–406, 2013.

[70] K. Lu, A. Chaudhuri, T.-Y. Lee, H.-W. Shen, and P. C. Wong, "Exploring vector fields with distribution-based streamline analysis." in *PacificVis*. Citeseer, 2013, pp. 257–264.

[71] J. Ma, J. Walker, C. Wang, S. Kuhl, and C. K. Shene, "Flowtour: An automatic guide for exploring internal flow features," in *Visualization Symposium (PacificVis), 2014 IEEE Pacific*. IEEE, 2014, pp. 25–32.

[72] J. Tao, C. Wang, and C. K. Shene, "Flowstring: Partial streamline matching using shape invariant similarity measure for exploratory flow visualization," in *Visualization Symposium (PacificVis), 2014 IEEE Pacific*. IEEE, 2014, pp. 9–16.

[73] Y. Li, C. Wang, and C.-K. Shene, "Streamline similarity analysis using bag-of-features," in *Visualization and Data Analysis 2014*, vol. 9017. International Society for Optics and Photonics, 2014, p. 90170N.

[74] ——, "Extracting flow features via supervised streamline segmentation," *Computers & Graphics*, vol. 52, pp. 79–92, 2015.

[75] H.-W. Shen, R. Vasko, and R. Wenger, "Visualizing flow fields using fractal dimensions," in *Proceedings of the Eurographics/IEEE VGTC Conference on Visualization: Short Papers*. Eurographics Association, 2016, pp. 25–29.

[76] J. Han, J. Tao, and C. Wang, "Flownet: A deep learning framework for clustering and selection of streamlines and stream surfaces," *IEEE transactions on visualization and computer graphics*, 2018.

[77] S. Bryson, S. Johan, L. Schlecht, B. Green, D. Kenwright, and M. Gerald-Yamasaki, "The virtual windtunnel," in *Computational Fluid Dynamics Review 1998: (In 2 Volumes)*. World Scientific, 1998, pp. 1113–1130.

[78] M. Schulz, F. Reck, W. Bartelheimer, and T. Ertl, "Interactive visualization of fluid dynamics simulations in locally refined cartesian grids (case study)," in *Proceedings of the conference on Visualization'99: celebrating ten years*. IEEE Computer Society Press, 1999, pp. 413–416.

[79] A. Wiebel and G. Scheuermann, "Eyelet particle tracing-steady visualization of unsteady flow," in *Visualization, 2005. VIS 05. IEEE*. IEEE, 2005, pp. 607–614.

[80] C. Koehler, T. Wischgoll, H. Dong, and Z. Gaston, "Vortex visualization in ultra low reynolds number insect flight," *IEEE Transactions on Visualization & Computer Graphics*, no. 12, pp. 2071–2079, 2011.

[81] T. McLoughlin, M. W. Jones, R. S. Laramee, R. Malki, I. Masters, and C. D. Hansen, "Similarity measures for enhancing interactive streamline seeding," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 8, pp. 1342–1353, 2013.

[82] L. Zhang, G. Chen, R. S. Laramee, D. Thompson, and A. Sescu, "Flow visualization based on a derived rotation field," *Electronic Imaging*, vol. 2016, no. 1, pp. 1–10, 2016.

[83] T. Günther, C. Rössl, and H. Theisel, "Hierarchical opacity optimization for sets of 3d line fields," in *Computer Graphics Forum*, vol. 33, no. 2. Wiley Online Library, 2014, pp. 507–516.

[84] J. P. M. Hultquist, "Constructing stream surfaces in steady 3d vector fields," in *Proceedings Visualization '92*, Oct 1992, pp. 171–178.

[85] D. Bauer, R. Peikert, M. Sato, and M. Sick, "A case study in selective visualization of unsteady 3d flow," in *Proceedings of the conference on Visualization'02*. IEEE Computer Society, 2002, pp. 525–528.

[86] W. Engelke, K. Lawonn, B. Preim, and I. Hotz, "Autonomous particles for interactive flow visualization," in *Computer Graphics Forum*. Wiley Online Library, 2018.

[87] G.-S. Li, U. D. Bordoloi, and H.-W. Shen, "Chameleon: An interactive texture-based rendering framework for visualizing three-dimensional vector fields," in *Visualization, 2003. VIS 2003. IEEE*. IEEE, 2003, pp. 241–248.

[88] H.-W. Shen, U. D. Bordoloi, and G.-S. Li, "Interactive visualization of three-dimensional vector fields with flexible appearance control," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 4, pp. 434–445, 2004.

[89] A. Helgeland and T. Elboth, "High-quality and interactive animations of 3d time-varying vector fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1535–1546, 2006.

[90] S. Guthe, S. Gumhold, and W. Straßer, "Interactive visualization of volumetric vector fields using texture based particles," 2002.

[91] A. Vilanova, G. Berenschot, and C. Van Pul, "Dti visualization with streamsurfaces and evenly-spaced volume seeding," in *Proceedings of the Sixth Joint Eurographics-IEEE TCVG conference on Visualization*. Eurographics Association, 2004, pp. 173–182.

[92] D. Merhof, M. Sonntag, F. Enders, P. Hastreiter, R. Fahlbusch, C. Nimsky, and G. Greiner, "Visualization of diffusion tensor data using evenly spaced streamlines," 2005.

[93] A. Helgeland and O. Andreassen, "Visualization of vector fields using seed lic and volume rendering," *IEEE Transactions on Visualization & Computer Graphics*, no. 6, pp. 673–682, 2004.

[94] L. P. Chew, "Guaranteed-quality mesh generation for curved surfaces," in *Proceedings of the ninth annual symposium on Computational geometry*. ACM, 1993, pp. 274–280.

[95] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, "The farthest point strategy for progressive image sampling," *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1305–1315, 1997.

[96] H. Edelsbrunner and D. Guoy, "Sink-insertion for mesh improvement," in *Proceedings of the seventeenth annual symposium on Computational geometry*. ACM, 2001, pp. 115–123.

[97] S. Oudot and J.-D. Boissonnat, "Provably good surface sampling and approximation." in *Symposium on Geometry Processing*, 2003, pp. 9–18.

[98] G. V. Bancroft, F. J. Merritt, T. C. Plessel, P. G. Kelaita, R. K. McCabe, and A. Globus, "Fast: a multi-processed environment for visualization of computational fluid dynamics," in *Visualization, 1990. Visualization'90., Proceedings of the First IEEE Conference on*. IEEE, 1990, pp. 14–27.

[99] J. R. Shewchuk, "Triangle: Engineering a 2d quality mesh generator and delaunay triangulator," in *Applied computational geometry towards geometric engineering*. Springer, 1996, pp. 203–222.

[100] H. Theisel, T. Weinkauf, H.-C. Hege, and H.-P. Seidel, "Saddle connectors-an approach to visualizing the topological skeleton of complex 3d vector fields," in *Visualization, 2003. VIS 2003. IEEE*. IEEE, 2003, pp. 225–232.

[101] K. Mahrous, J. Bennett, G. Scheuermann, B. Hamann, and K. I. Joy, "Topological segmentation in three-dimensional vector fields," *IEEE Transactions on Visualization & Computer Graphics*, no. 2, pp. 198–205, 2004.

[102] R. L. Cook, "Stochastic sampling in computer graphics," *ACM Transactions on Graphics (TOG)*, vol. 5, no. 1, pp. 51–72, 1986.

[103] W. Zhang and J. Su, "Extraction of limit streamlines in 2d flow field using virtual boundary," in *Computational Intelligence and Security, 2009. CIS'09. International Conference on*, vol. 1. IEEE, 2009, pp. 171–175.

[104] T. Weinkauf and H. Theisel, "Curvature measures of 3d vector fields and their applications," 2002.

[105] T. Weinkauf, H.-C. Hege, B. R. Noack, M. Schlegel, and A. Dillmann, "Coherent structures in a transitional flow around a backward-facing step," *Physics of Fluids*, vol. 15, no. 9, pp. S3–S3, 2003.

[106] R. Abraham, J. E. Marsden, and T. Ratiu, *Manifolds, tensor analysis, and applications*. Springer Science & Business Media, 2012, vol. 75.

[107] D. Stalling and H.-C. Hege, "Fast and resolution independent line integral convolution," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, 1995, pp. 249–256.

[108] A. Fuhrmann and E. Gröller, "Real-time techniques for 3d flow visualization," in *Proceedings of the conference on Visualization'98*. IEEE Computer Society Press, 1998, pp. 305–312.

[109] J. Kruger, P. Kipfer, P. Konclratieva, and R. Westermann, "A particle system for interactive visualization of 3d flows," *IEEE Transactions on visualization and computer graphics*, vol. 11, no. 6, pp. 744–756, 2005.
[110] K. Burger, P. Kondratieva, J. Kruger, and R. Westermann, "Importance-driven particle techniques for flow visualization," in *Visualization Symposium, 2008. PacificVIS'08. IEEE Pacific*. Citeseer, 2008, pp. 71–78.
[111] R. Van Pelt, J. O. Bescos, M. Breeuwer, R. E. Clough, M. E. Groller, B. ter Haar Romenij, and A. Vilanova, "Interactive virtual probing of 4d mri blood-flow," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2153–2162, 2011.
[112] A. Chaudhuri, T.-Y. Lee, H.-W. Shen, and R. Wenger, "Exploring flow fields using space-filling analysis of streamlines," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 10, pp. 1392–1404, 2014.
[113] T. Salzbrunn and G. Scheuermann, "Streamline predicates," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1601–1612, 2006.
[114] T. Salzbrunn, C. Garth, G. Scheuermann, and J. Meyer, "Pathline predicates and unsteady flow structures," *The Visual Computer*, vol. 24, no. 12, pp. 1039–1051, 2008.
[115] A. Brun, H. Knutsson, H.-J. Park, M. E. Shenton, and C.-F. Westin, "Clustering fiber traces using normalized cuts," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2004, pp. 368–375.
[116] B. Moberts, A. Vilanova, and J. J. Van Wijk, "Evaluation of fiber clustering methods for diffusion tensor imaging," in *Visualization, 2005. VIS 05. IEEE*. IEEE, 2005, pp. 65–72.
[117] L. ODonnell and C.-F. Westin, "White matter tract clustering and correspondence in populations," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2005, pp. 140–147.
[118] A. Tsai, C.-F. Westin, A. O. Hero, and A. S. Willsky, "Fiber tract clustering on manifolds with dual rooted-graphs," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007, pp. 1–6.
[119] M. Maddah, W. E. L. Grimson, S. K. Warfield, and W. M. Wells, "A unified framework for clustering and quantitative analysis of white matter fiber tracts," *Medical image analysis*, vol. 12, no. 2, pp. 191–202, 2008.
[120] S. Oeltze, D. J. Lehmann, A. Kuhn, G. Janiga, H. Theisel, and B. Preim, "Blood flow clustering and applications invirtual stenting of intracranial aneurysms," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 5, pp. 686–701, 2014.
[121] L. Shi and G. Chen, "Metric-based curve clustering and feature extraction in flow visualization," 2017.
[122] J. Ebling and G. Scheuermann, "Clifford convolution and pattern matching on vector fields," in *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*. IEEE Computer Society, 2003, p. 26.
[123] ——, "Segmentation of flow fields using pattern matching," in *Proceedings of the Eighth Joint Eurographics/IEEE VGTC conference on Visualization*. Eurographics Association, 2006, pp. 147–154.
[124] E. Heiberg, T. Ebbers, L. Wigstrom, and M. Karlsson, "Three-dimensional flow characterization using vector pattern matching," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 3, pp. 313–319, 2003.
[125] M. Schlemmer, M. Heringer, F. Morr, I. Hotz, M. Hering-Bertram, C. Garth, W. Kollmann, B. Hamann, and H. Hagen, "Moment invariants for the analysis of 2d flow fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1743–1750, 2007.
[126] R. Bujack, I. Hotz, G. Scheuermann, and E. Hitzer, "Moment invariants for 2d flow fields using normalization," in *Visualization Symposium (PacificVis), 2014 IEEE Pacific*. IEEE, 2014, pp. 41–48.
[127] R. A. Wagner and M. J. Fischer, "The string-to-string correction problem," *Journal of the ACM (JACM)*, vol. 21, no. 1, pp. 168–173, 1974.
[128] C.-K. Chen, S. Yan, H. Yu, N. Max, and K.-L. Ma, "An illustrative visualization framework for 3d vector fields," in *Computer Graphics Forum*, vol. 30, no. 7. Wiley Online Library, 2011, pp. 1941–1951.
[129] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10, no. 16. Seattle, WA, 1994, pp. 359–370.
[130] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, vol. 315, no. 5814, pp. 972–976, 2007.
[131] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.
[132] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.
[133] L. Orf, R. Wilhelmson, and L. Wicker, "Visualization of a simulated long-track ef5 tornado embedded within a supercell thunderstorm," *Parallel Computing*, vol. 55, pp. 28–34, 2016.
[134] D. Sujudi and R. Haimes, "Identification of swirling flow in 3-d vector fields," in *12th Computational Fluid Dynamics Conference*, 1995, p. 1715.