Surface-Based Flow Visualization

Matt Edmunds[1], Robert S. Laramee[1], Guoning Chen[2], Nelson Max[3], Eugene Zhang[4], Colin Ware[5]

[1]*Swansea University, UK.* [2]*University of Utah, US.* [3]*University of California, Davis, US.* [4]*Oregon State University, US.* [5]*University of New Hampshire, US.*

Abstract

With increasing computing power, it is possible to process more complex fluid simulations. However, a gap between increasing data size and our ability to visualize them still remains. Despite the great amount of progress that has been made in the field of flow visualization over the last two decades, a number of challenges remain. Whilst the visualization of 2D flow has many good solutions, the visualization of 3D flow still poses many problems. Challenges such as domain coverage, speed of computation, and perception remain key directions for further research. Flow visualization with a focus on surface-based techniques forms the basis of this literature survey, including surface construction techniques and visualization methods applied to surfaces. We detail our investigation into these algorithms with discussions of their applicability and their relative strengths and drawbacks. We review the most important challenges when considering such visualizations. The result is an up-to-date overview of the current state-of-the-art that highlights both solved and unsolved problems in this rapidly evolving branch of research.

Keywords: Survey, Flow Visualization, Surfaces

1. Introduction

Flow visualization is a powerful means for exploring, analyzing and communicating simulation or experimental results. Flow visualization is characterized by a range of differing techniques such as direct, feature, texture, and geometric-based representations [PVH*03, LHD*04]. Each technique has a range of differing accuracies and speeds [LEG*08]. The phenomena to be studied can be sampled using regular or irregular grids, which can stem from steady-state or unsteady flow. There are many challenges to overcome in this field of research. The topic of flow visualization with surfaces has become an increasingly important field of research in recent years (see Figure 1). This is due to the advantages that surface-based techniques offer over more traditional curve-based methods and the maturity of 2D flow visualization. This provides strong motivation for studying and categorizing the breadth and depth of surface-based research for flow visualization.

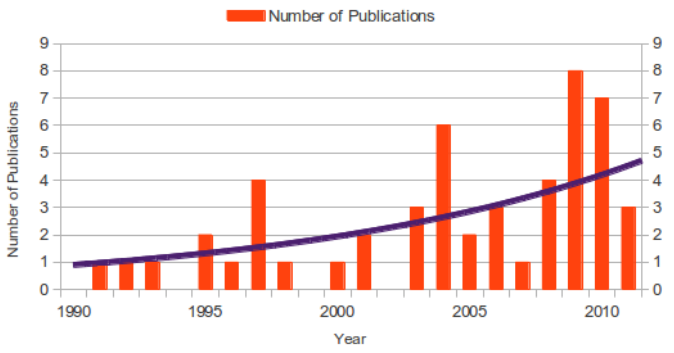

Figure 1: This histogram shows the number of publications per year focused on flow visualization with surfaces. It indicates the growing momentum and importance of this topic.

1.1. Sources of Flow Data

There are many different origins of vector data which can be categorized, for example: flow simulation and flow measurement. Flow simulation is often used to predict real-world conditions both as an aid to design and for the analysis of large physical systems which may not be captured or recorded with existing devices. This is especially true for very large and expensive projects such as aircraft, ship, and car design. The cost of building physical prototypes for testing can be great. Therefore the ability to simulate the test conditions using virtual representations such as CAD (Computer Aided Design) data combined with computational systems such as CFD (Computational Fluid Dynamics) is highly beneficial.

1.2. Applications

The application of flow visualization techniques to real-world problems is essential for engineers and practitioners to gain an understanding of the information the data contains. Current techniques have been integrated into a wide variety of test and simulation systems, for example Fluent [ANS10]. Allowing the engineers to explore and evaluate data from within these systems in visual forms is key to effectively gain insight. Manually processing large amounts of numerical data is time consuming, prone to error, and is only performed by specialists.

[1]email: {csmatti, r.s.laramee}@swansea.ac.uk
[2]email: chengu@sci.utah.edu
[3]email: max@cs.ucdavis.edu
[4]email: zhange@eecs.oregonstate.edu
[5]email: cware@ccom.unh.edu

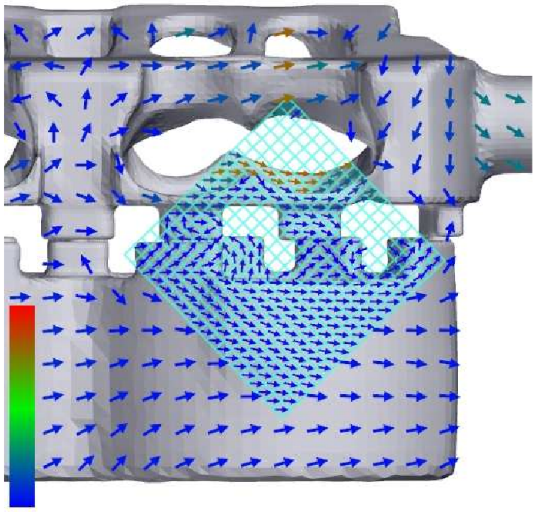

Figure 2: A multi-resolution visualization using glyphs illustrating the flow at the surface of a cooling jacket as part of an engine simulation. Color is mapped to velocity magnitude. Image courtesy of R.S.Laramee et al. [PL08].

The graphical representation and exploration of data not only allows for much faster analysis but also enables non-experts to understand the underlying phenomenon.

As datasets increase in size and complexity it becomes more important to support effective exploration of their features and characteristics. Many of today's applications for flow visualization are centered around fields such as aerospace, automotive, energy production, and other scientific research. Automotive design tends to focus on aerodynamic drag to help improve the efficiency of the vehicle, and airflow around the engine compartments which provide much needed cooling for the various engine/cockpit heat exchange systems. The heat exchange systems themselves are also an important focus for analysis. (see Figures 2 and 3). Aerospace also focuses on aerodynamics and engine design for aircraft, with similar such focuses for space craft. Electromagnetism and turbine design within the energy production industries are common applications of vector field visualizations [RP96].

An application example within astrophysics is the visualization of 2008 IEEE Visualization Design Contest [IEE10]; the simulation of ionization front instability. Another area of study is the topic of acoustic flow to simulate and visualize such things as engine exhaust, and speaker cabinet design [Tan10]. The medical field uses visualizations to study such phenomena as blood flow, for example, when designing heart pumps to support failing hearts. Another area of utilization is weather systems analysis by organizations such as the MET office [GOV10].

1.3. Challenges

Surface-based approaches share some common problems associated with flow visualization in general. Examples of these challenges include: large, time-dependent simulation data requiring the utilization of out-of-core techniques; and the handling of unstructured data. Surface-based methods also face their own unique challenges which we discuss in more detail throughout the paper.

Construction. Surface construction is a key topic for this survey. Surfaces must represent an accurate approximation of the underlying simulation. Adequate sampling must be maintained while reducing the extra computational overhead associated with over-sampling. Resulting meshes must also remain smooth in the presence of various flow phenomena such as vortex cores, and highly divergent or convergent flow. A large amount of effort has been put into the creation of various types of surfaces and these form a large portion of this survey. See Section 2 for literature that addresses this challenge.

Occlusion. When using surfaces the problem of occlusion occurs frequently. This may stem from multiple surfaces that occlude one another, a large surface that produces self occlusion, or a combination of both. There are several approaches that can be taken depending on the surface type to reduce this problem. A general approach is to use transparency. With integral surfaces, i.e., surfaces to which the flow field is tangent, we have more options. Advanced texture mapping may also be used. Additionally, integral surface seeding positions may be changed to reduce clutter. See Section 3 for literature that addresses this challenge.

Information Content. While surfaces offer many advantages in terms of perception, a basic visualization of the surface alone may not provide sufficient information about the underlying data. For example a stream surface alone does not show the behavior of inner flow contained within the surface. A review of the research that enhances the resulting visualization of surfaces is also provided in this survey. See Section 3 for literature that addresses this challenge.

Placement and Seeding. Interactive placement is the most common method currently used. There is a strong correlation

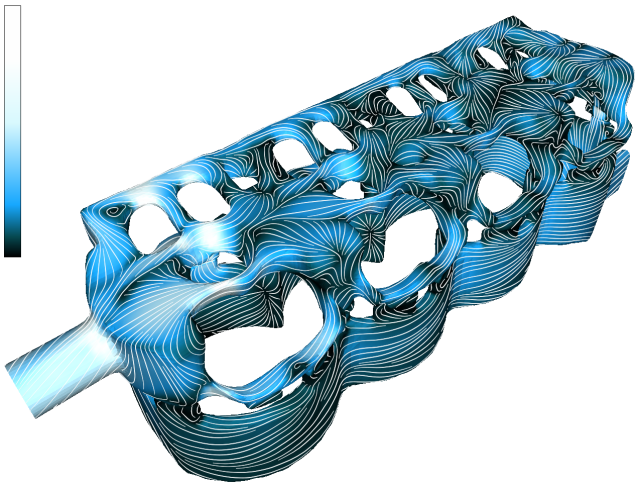

Figure 3: Evenly spaced streamlines on the boundary surface of a cooling jacket flow simulation. Image courtesy of R.S.Laramee et al. [SLCZ09].

Classification			
Constructing Surfaces for Flow Visualization			
Integral: Stream/Path	Integral: Streak/Time	Implicit	Topological
[Hul92]s, [USM96]s, [SBH*01]s, [GTS*04]s, [STWE07]t, [GKT*08]t, [PCY09]s, [SWS09]s, [MLZ09]t, [PS09]s. [YMM10]t. [SRWS10]s.	[vFWTS08a]t, [KGJ09]t, [BFTW09]t, [MLZ10]t, [FBTW10]t,	[vW93]s, [WJE00]s, [Gel01]s.	[TWHS03]s, [WTHS04]s, [TSW*05]t, [BSDW12]t
Rendering Flow on Surfaces for Visualization			
Direct	Geometric	Texture: Static Texture	Texture: Dynamic Texture
[PL08]s, [PGL*12]s.	[LMG97]s, [LMGP97]s, [WH06]s, [SLCZ09]s, [BWF*10]s, [HGH*10]t, [EML*11]s, [ELM*12]s, [ELC*12]s.	[vW91]s, [dLvW95]s, [FC95]t, [MKFI97], [BSH97]s, [SK98]t, [Wei09]t, [PZ10]s.	[LJH03]t, [vW03]t, [LvWJH04]t, [LSH04]s, [LWSH04]s, [WE04]t, [LGD*05]s, [LGSH06]s, [BSWE06]s, [LTWH08]t.

Table 1: This table classifies surface techniques into two main categories; Constructing surfaces for flow visualization and Rendering flow on surfaces for visualization. The table also sub-classifies the surface construction into Integral, Implicit, and Topological, with Integral surfaces further divided between stream/path and streak surfaces. Additional sub-classification of this section into point based, triangle based and quad based primitives are shown with color. The rendering section is sub classified into Direct, Geometric, and Texture based techniques, with the texture based category further subdivided into static texture and dynamic texture based techniques. Additional sub-classification of this section into Parameter Space , and Image Space techniques are displayed with color. An additional suffix is used to represent techniques applied to steady-state (s), and time-dependent (t) vector fields. Each of the entries are ordered chronologically within each subcategory.

between seeding and occlusion of integral surfaces. Seeding too many surfaces, or seeding them in such a way that they occupy the same region of the domain leads to high levels of occlusion. The placement of isosurfaces is a function of the selected isovalue. Choosing optimal isovalues is an analogous problem. See Sections 2.2, 2.3, 3.1 and 3.2 for literature which studies these challenges.

1.4. Classification

The classification in this paper represents the subtopics of flow visualization with surfaces. The classification highlights areas which are more mature and areas which require additional work. Refer to Table 1.

The two main classifications deal with surface construction techniques e.g., the fluid flow is represented by the surface curvature, and surface rendering techniques e.g., fluid flow properties that are visualized on the surface geometry. This classification highlights the difference between applying visualization techniques to surfaces, and the underlying surface construction.

Each of the two classifications are further sub-classified. The surface construction is classified into Integral, Implicit, and Topological techniques, while the visualization of flow on surfaces is divided into Direct, Geometric, and Texture-based techniques. Topological techniques visualize flow topology explicitly using surfaces to do so. Surface construction is also sub-classified into triangle, point, and quad-based meshing techniques, while the visualization of flow on surfaces is sub-classified into parameter space-based, and image space-based techniques.

Another possible choice for the visualization of flow on surfaces is classification into single chart (single parametrization) and a collection of charts (atlas-based parameterization). The image space methods would be treated similar to other simple parameterizations such as C-Space techniques. The sub-categories are further divided into steady flow and time-dependent flow. The papers within each subcategory are ordered chronologically. We note that this survey does not cover flat or planar surfaces or slices like those described by Laramee [Lar03].

1.5. Contributions and Summary

Surface techniques fall into two main categories: construction of surfaces and visualization techniques applied to surfaces. Surface construction techniques are a fairly well researched topic. The majority of techniques are variations and extensions of the original Hultquist method [Hul92]. These techniques are either faster, more accurate, cater to large data domains, or address topology. The self-occluding problems inherent of surfaces are partially addressed by Löffelmann et al. who effectively create holes in the surface [LMGP97], Theisel et al. [TWHS03] use connectors to represent the separating surface, and the general approach of using translucency to alleviate occlusion e.g., see Sections 2 and 3.

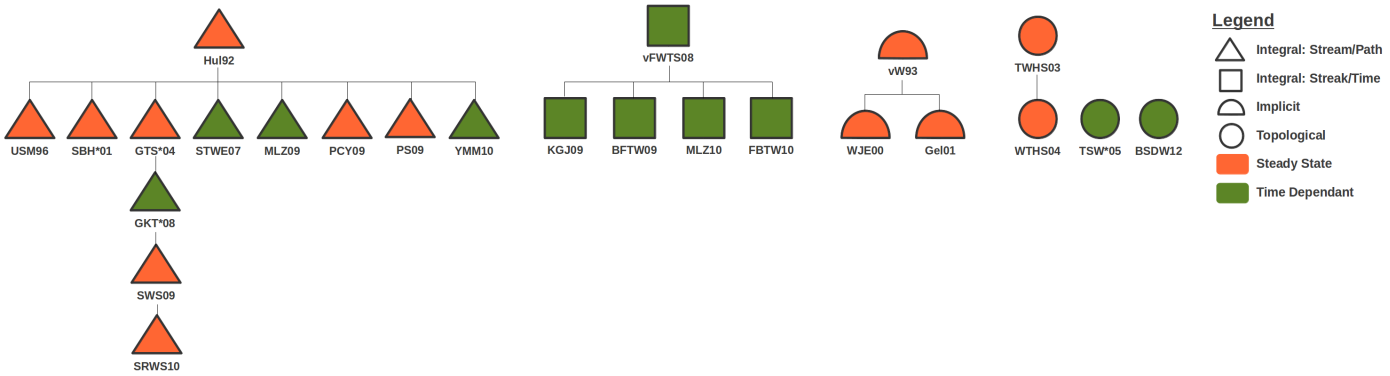

Figure 4: The classifications of construction techniques illustrates a chronological flow of work from author to author. The child-parent relationships indicates the key ideas that are progressed. The flow of work diverges and in some cases converges as new concepts are built on top of previous ideas. The charts also show the originating work, and where key work is continued.

The methods of van Wijk [vW93] and Westermann et al. [WJE00] concentrate on deriving a scalar field from a vector field and then employing isosurface techniques to represent the domain. Although these techniques provide good domain coverage, visual clutter and occlusion can result. The projection of vector information onto a surface by Laramee et al. [LGSH06], improves performance and perception of the flow local to that surface, but the surface occlusion issue remains due to the nature of the image space-based techniques. In addition, isosurface work in the area of unsteady flow data is limited.

A fairly common theme throughout the different surface types is the application of additional visualization techniques to enhance the surfaces. Parameter and image space-based techniques are the main focus of these visualizations as they can provide effective interactive solutions. Given a surface (it can be any type, as long as it is manifold) and a vector field defined on it, the flow behavior can be illustrated with the desired dimension of visual mapping, such as 0D (hedgehogs), 1D (streamlines), or 2D (textures). This enables not only the direct display of the flow data in the Eulerian point of view, or the visualization of the behavior of the selected particles in the Lagrangian point of view, but also the complete (dense) image of the flow behavior over the surfaces. In addition, combined with other conventional visualization techniques, such as color coding and animation, more complete flow information including both vector magnitude and orientation, as well as the time varying characteristics, can be conveyed.

The main benefits and contributions of this survey survey are:

- A review of the latest research developments in flow visualization with surfaces.
- The introduction of a novel classification scheme based on challenges including; construction, rendering, occlusion, and perception. This scheme lends itself to an intuitive grouping of papers that are naturally related to one another.
- The classification highlights both mature areas where many solutions have been provided and unsolved problems.
- A concise overview in the area of flow visualization with surfaces for those who are interested in the topic and wishing to carry out research in this area.

This report is divided into four main sections: First is a review of the surface construction techniques in Section 2. Then a review of flow visualization on surfaces in Section 3. An analysis of the different sub-classifications is conducted in Section 4 with an emphasis on the initial seed or surface placement/generation, perception, visual clutter and occlusion. Finally the survey finishes with conclusions drawn from the analysis, and proposed areas of future work in Section 5.

2. Constructing Surfaces For Flow Visualization

This section studies the different construction techniques surveyed. Figure 4 shows a chronological flow of work from author to author. The child-parent relationships indicate the origin and evolution of key ideas. The work diverges as new concepts are built on top of previous ideas.

We start this section with a discussion about flow data and associated challenges, before moving on to the sub-classification of primitive types used for the surface mesh representations. Following this we study the range of surface construction techniques. These are divided into Integral, Implicit, and Topological surface construction methods.

Steady State, Time Dependent and Large Complex Data

Early work focuses on processing steady-state simulations which represent a static flow field or single snapshots of flow in time. As the ability to process larger amounts of data increases, the research into processing this data follows. Data sampling becomes denser, the size of the simulation domain increases, and multiple time steps are incorporated and processed. The structure of the data can be complex, incorporating a range of associated scalar quantities representing range of additional attributes.

Velocity data is comprised of a set of x, y, z components for each sample point within the data domain. For example a 3D steady-state vector field $\boldsymbol{v}_s(\boldsymbol{p}) \in \mathbb{R}^3$ where $\boldsymbol{v}_s(\boldsymbol{p})$

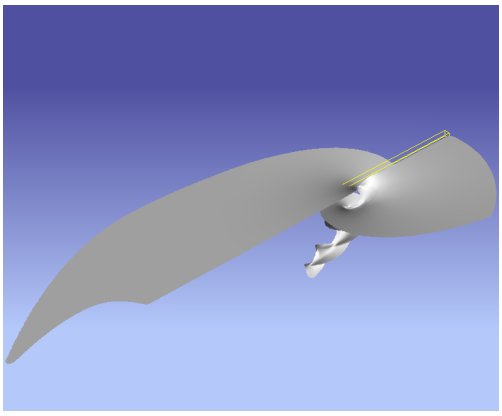

Figure 5: Visualization of the flow field of a tornado with a point-based stream surface. The stream surface is seeded along a straight line in the center of the respective image. Image courtesy of D.Weiskopf et al. [STWE07].

$= \left[\boldsymbol{v}_x(x,y,z) \quad \boldsymbol{v}_y(x,y,z) \quad \boldsymbol{v}_z(x,y,z)\right]$ for $\boldsymbol{p} \in \Omega$, $\boldsymbol{v_s} \in \mathbb{R}^3$ and $\Omega \subset \mathbb{R}^3$, where Ω may be a 3D regular, structured, unstructured or irregular grid. For unsteady flow we have a time-dependent vector field $\boldsymbol{v}_t(\boldsymbol{p}) \in \mathbb{R}^3$ where $\boldsymbol{v}_t(\boldsymbol{p}) = \left[\boldsymbol{v}_x(x,y,z,t) \quad \boldsymbol{v}_y(x,y,z,t) \quad \boldsymbol{v}_z(x,y,z,t)\right]$ for $\boldsymbol{p} \in \Omega$, $\boldsymbol{v_t} \in \mathbb{R}^3$ and $\Omega \subset \mathbb{R}^3$.

Our survey discusses work addressing the challenges of both steady and time-dependent data. In Table 1 we use an 's' suffix to the citation for techniques which process steady state vector fields and a 't' suffix for techniques addressing time-dependent data. The trend of moving from steady-state toward time-dependent data can be observed in the chronological classification of work.

Point vs. Triangle vs. Quad-based Construction Techniques

Examining the sub-classification of surface mesh construction techniques into point-, triangle-, and quad-based methods yields some interesting insights into these approaches. Point-based surfaces are generally simpler and faster to construct as they do not require any mesh construction computation to represent a closed surface. This approach can be limiting regarding rendering options. Rendering the vertices as simple point sprites, disks, or spheres is effective for dense vertex representations, but gaps or inconsistencies can appear when viewing in close proximity to the surface. Lighting can also be a challenge with this technique as the methods available for normal calculation become limited and increase computation. Another approach to rendering is using image-based techniques which don't explicitly require geometric primitives to render closed surfaces.

Schafhitzel et al. [STWE07] present a point based stream and path surface algorithm where the vertices for the surface representation are generated on the GPU. With a dense output, each of the vertices and their normals are used to render a closed surface as small, lit, point sprites, as in Figure 5. A texture-based closed surface rendering can also be achieved using LIC [CL93] performed in image space. With a similar approach, Ferstl et al. [FBTW10] present a streak surface algorithm which has a rendering option using spherical point sprites to represent a closed surface.

To represent a geometric mesh for rendering we must define how the mesh is constructed. Of the two common methods for defining a mesh, one is defined by constructing an array of vertices in the correct order for rendering the primitive. This approach can hold redundant instances of the same vertices for a given mesh, with a larger memory footprint and increased data traversing the graphics bus at the cost of valuable bandwidth. In some simple surface construction implementations however this method can be easier to implement.

A second approach to defining a mesh for rendering primitives is the utilization of an indexing array along with the array representing the vertices of the surface. An additional index array specifies each vertex in the correct order to construct the geometric primitive. In the context of surfaces the result of this approach is a much smaller data array, but the addition of an index array. The index array can be compressed, depending on the quantity of vertices, by using data types requiring less memory such as unsigned characters or unsigned short integers instead of integers. This also significantly reduces the graphics bandwidth.

Another constraint on implementations is the method used for constructing the data and index arrays. The most common mesh primitive used is the triangle. This is the default for surface construction techniques such as isosurfaces as used by van Wijk [vW93] and Westermann et al. [WJE00]. This is a bi-product of early graphics card support for rendering triangles.

The most common approach for meshing integral surfaces with triangles is a greedy minimal tiling strategy as described by Hultquist [Hul92]. Other approaches include; additional processing for streak surfaces where the surface topology changes with time as in the work by Krishnan et al. [KGJ09] as shown in Figure 6, and processing of irregular grids such as tetrahedra as described by Scheuermann et al. [SBH*01].

With advancing front integration techniques, a simple approach to generating a mesh is the direct use of the quad patch represented by the bounding streamlines and timelines. This approach initially requires less computational expense, how-

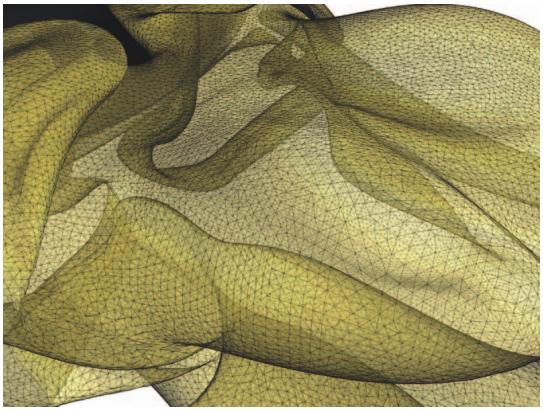

Figure 6: Time surface mesh in the Ellipsoid dataset. Although the surface has undergone strong deformation, the mesh remains in good condition. Image courtesy of C.Garth et al. [KGJ09].

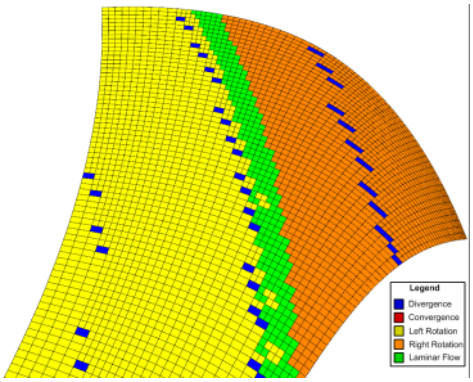

Figure 7: The algorithm by McLoughlin et al. [MLZ09] handles widely diverging flow while maintaining the desired organized advancing front. This surface is colored according to the underlying flow characteristics: left rotation is mapped to yellow, right rotation is mapped to orange, parallel flow is mapped to green, divergence is mapped to blue, convergence is mapped to red. Image courtesy of R.S.Laramee et al. [MLZ09].

ever dealing with issues of sheering quads, t-junctions and additional normal computations (one per quad corner rather than one per triangle) can have a significant impact. Refer to Figure 7 and see McLoughlin et al. [MLZ09, MLZ10] and Schneider et al. [SWS09]. Peikert and Sadlo [PS09] construct their surface geometry in an incremental manner advancing from the initial curve structure attempting to avoiding the issue of quad sheering. The algorithm by van Gelder et al. [Gel01] also lends itself to quad-based meshing due to the connectivity of the curvilinear grids.

2.1. Integral Construction Techniques

In this subsection we present an overview of integral surface construction techniques. This work is subdivided into stream/path, and streak/time surface construction algorithms. The similarity between the stream and path surface construction provides a natural classification for discussion in the next subsection. Following the review of stream/path surfaces we then present a study of streak surface construction. The main challenges addressed by the literature in this subsection are accurate surface construction, performance time, and continuous representations.

2.1.1. Stream/Path Surface Construction

A **streamline** is a curve which is tangent to the velocity field at every point along its length. A Streamline is the trace of a massless particle from an initial location (seed point). Streamlines show the direction fluid flow within a steady-state flow domain. if $\boldsymbol{v}(p)$ is a three dimensional vector field, the streamline through a point x_0 is the solution $I(x_0, t)$ to the differential equation:

$$\frac{d}{dt} I(x_0, t) = \boldsymbol{v}(I(x_0, t))$$

with the initial condition $I(x_0, 0) = x_0$. A **stream surface** is the trace of a one-dimensional seeding curve C through the flow. The resulting surface is everywhere tangent to the local flow. A stream surface S is defined by:

$$S(s, t) := I(C(s), t)$$

S is the union or continuum of integral curves passing through the seeding curve C. $S(s, -)$ coincides with an individual integral curve, and $S(-, t)$ coincides with individual time lines [GKT*08].

Since there is no normal component of the velocity along streamlines and stream surfaces, mass cannot cross their boundary and therefore they are useful for separating distinct regions of similar flow behavior. In practical applications a discretized approximation of the stream surface is constructed by tracing discretized seeding curves through the vector field using integration methods such as the fourth-order Runge Kutta integration scheme.

Hultquist introduces one of the first methods [Hul92] for the generation of stream surface approximations. This technique includes strategies for controlling the density of particles across the advancing front. Points can be added to the advancing front when the sampling rate becomes too sparse and removed when it becomes too dense. Neighboring pairs of streamlines are tiled with triangles to form ribbons. These connected ribbons then form the surface. Ribbons which encounter rapid divergence of flow may be torn/ripped to allow the surface to flow around an obstacle. The separate portions of the surface are then computed independently. Hultquist builds on work by Belie [Bel87] and Kerlick [Ker90] who describe narrow stream ribbon methods, and Schroeder et al. [SVL91] who describe a stream primitive called the stream polygon.

Ueng et al. [USM96] expand the stream surface work to stream ribbons, stream tubes, and streamlines on unstructured grids. The authors build on the stream polygons algorithm by Darmofal and Haimes [DH92], the research by Ma and Smith [MS93], and the steam ribbons of Pagendarm [PW94]. This paper describes extending the techniques to unstructured grids by converting the physical coordinate system to a canonical coordinate system. The main idea is the use of a specialized fourth-order Runge Kutta integrator which requires only one matrix-vector multiplication and one vector-vector addition to calculate the successive streamline vertices. This technique

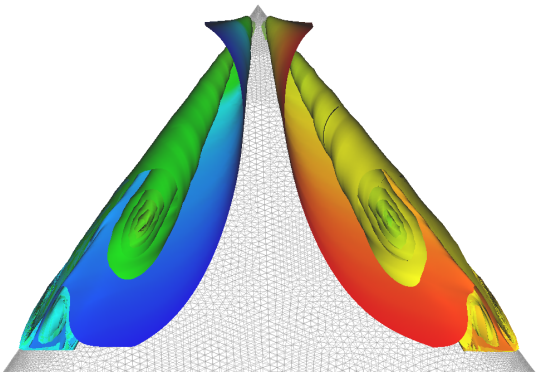

Figure 8: The formation of vortices at the apex of a delta wing illustrated with the use of a stream surface. Image courtesy of C.Garth et al. [GTS*04].

significantly simplifies the construction of the geometric primitives, reducing the computational cost and therefore improving speed.

Scheuermann et al. [SBH*01] present a method of stream surface construction on tetrahedral grids. The technique propagates the surface through the tetrahedral grid, one tetrahedron at a time, calculating on the fly where the surface intersects the tetrahedron. This approach enables the inclusion of topological information from the cells such as singularities. When the surface passes through the tetrahedron, the curve segments end points are traced as streamlines through the next cell. For each point on a streamline, a line is added connecting it to its counterpoint. These are then clipped against the faces of the tetrahedron cell and the result forms the boundary of a polygonal surface within the cell. This method is inherently compatible with multi-resolution grids and handles increased grid resolution in intricate flow regions. This work is a significant improvement over the previous irregular grid work, improving surface construction within complicated areas, typically of more interest in fluid dynamics.

Garth et al. [GTS*04] improve on the Hultquist method and showed how to obtain surfaces with higher accuracy in areas of intricate flow. See Figure 8. The improvements are achieved by employing arc length particle propagation and additional curvature-based front refinement. They also considered visualization options such as color mapping of vector field related variables going beyond straightforward surface rendering. A novel method to determine boundary surfaces of vortex cores and a scheme for phenomenological extraction of vortex core lines using stream surfaces is discussed and its accuracy is compared to one of the most established standard techniques.

Next we review the concept of a path surface. We start by describing a pathline. A **pathline** or particle trace is the trajectory that a massless particle takes in time dependent fluid flow. If $\boldsymbol{v}(x,t)$ is a three-dimensional vector field for x in domain $\Omega \in \mathbb{R}^3$ and t in a time interval $[T_0, T_1]$ the pathline $I(x_0, t_0; t)$ passing through x_0 at time t_0 is the solution to the ordinary differential equation:

$$\frac{d}{dt} I(x_0, t_0; t) = \boldsymbol{v}(I(x_0, t_0; t), t)$$

with the initial condition $I(x_0, t_0; t_0) = x_0$. A **path surface** is the trajectory of a massless curve C in time-dependent fluid flow. A path surface P is defined by:

$$P(s,t) := I(C(s), t_0; t)$$

Where P is the union or continuum of pathlines passing through the seeding curve C at time t_0. $S(s,-)$ coincides with an individual integral curve, and $S(-,t)$ coincides with an individual time lines [GKT*08]. The first example of this is the work by Schafhitzel et al. [STWE07] who introduce a point-based algorithm for stream and path surface construction and rendering.

Schafhitzel et al. combine and build on three specific areas: stream surface computation Hultquist [Hul92], rendering of point-based surfaces Zwicker et al. [ZPKG02], and texture-based flow visualization on surfaces (Weiskopf et al. [WE04]).

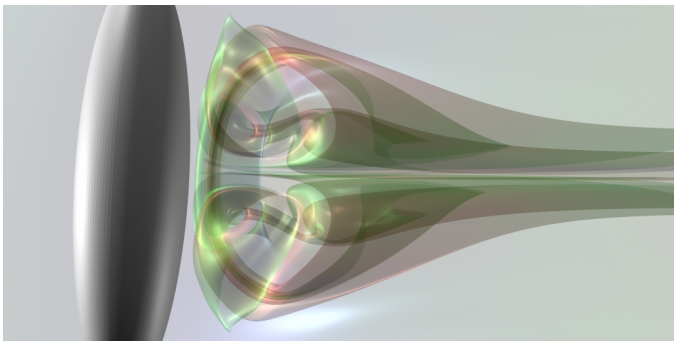

Figure 9: Path surface visualization of vortex shedding from an ellipsoid. The transparent surface consists of 508,169 triangles. Different layers identified by color mapping. Image courtesy of C.Garth et al. [GKT*08]. © IEEE/TVCG.

The stream/path surface generation is modified to run on the GPU in a highly parallel fashion. Seed points are generated and integrated through the vector field. To maintain a roughly even density, integrated points along the advancing front are inserted and removed. The surface rendering method is based on point set surfaces (PSS) and is extended to include stored connectivity information, this enables quick access to neighboring points. The authors' approach to the hybrid object/image space LIC method displays clear line patterns which show a choice of path lines.

More recently, Garth et al. [GKT*08] replaced the advancing front paradigm by an incremental time line approximation scheme. See Figure 9. This allows them to keep particle integration localized in time. The authors propose a decoupling of the surface geometry and graphical representation, and a curve refinement scheme which is used to approximate time lines, yielding accurate path surfaces in large time-dependent vector fields.

Following recent work by Bachthaler and Weiskopf [BW08] who describe the use of tracing structures perpendicular to the vector field to generate animated LIC patterns orthogonal to the flow direction, and work by Rosanwo et al. [RPH*09] which introduces dual streamline seeding based on streamlines orthogonal to the vector field, Palmerius et al. [PCY09] introduce the concept of perpendicular surfaces. These surfaces are perpendicular to the underlying vector field.

The authors describe the common properties of such surfaces and discuss the issues of non-zero helicity density, and stop conditions. The construction method starts at a predefined seed point propagating outwards in a clockwise spiral fashion where each new point is integrated perpendicular to the flow field for a given distance. The stop criteria are: length limit, accumulated winding angle limit, and maximum orientation error as a result of vector fields with non-zero helicity. Convergence or divergence is characterized by cone shaped surfaces. A combination of both results are saddle shaped surfaces. Vortices distort the surfaces by tearing them apart and producing a fan like pattern. A fast approach for generating the surfaces and stop conditions is also described.

Extending the previous work by Garth et al [GKT*08], Schneider et al. [SWS09] produce a more accurate and

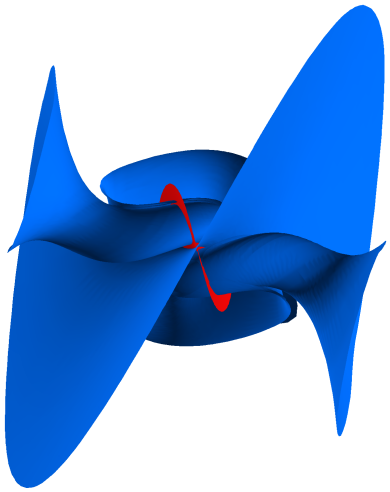

Figure 10: 2D manifolds visualizing the critical point of the Lorenz dynamical system. Image courtesy of R.Peikert et al. [PS09].

smoother timeline interpolation using a fourth-order Hermite interpolation scheme when adjusting the advancing front density. The Hermite interpolation between streamlines requires the covariant derivatives to be calculated with respect to s (streamline) and t (timeline). An additional surface accuracy error criterion is used to dictate when coarsening or refinement takes place. The error based refinement strategy splits a ribbon when the local interpolation error exceeds a given bound. This error is estimated directly by seeding a new short streamline from a position between neighboring streamlines at time $t_n - 1$ integrating it to time t_n.

Alternatively, McLoughlin et al. [MLZ09] describe a simple and fast CPU-based method for creating accurate stream and path surfaces using quad primitives. The authors propose a method which is based on a small set of simple, local operations performed on quad primitives, requiring no global re-meshing strategy. To handle divergent, convergent, and rotational flow, the sampling rate of the advancing front is updated. The quad is either divided (in areas of divergence) or collapsed (in areas of convergence). For curvature, a test of the advancing front rotation is conducted. If true then the advancing front integration step size is reduced by a factor dependent on the amount of rotation.

Using topology for the construction and placement of flow geometry, Peikert and Sadlo present topology relevant methods for constructing seeding curves to produce topologically-based stream surfaces [PS09]. The authors build on work by Garth et al. [GTS*04] expanding the notion of feature visualization applying stream surfaces to a range of singularities and periodic orbits. The discretised offset curves constructed at the topological structures are used to initialize the stream surface propagation. See Figure 10. The authors construct their stream surface from quads which are divided in areas of divergence to maintain a consistent mesh topology. This is achieved by subdividing between neighboring nodes with a cubic interpolant after tracing back a fixed number of steps. The nodes of the mesh retain a number of attributes representing the flow field, which are used for controlling the growth of the surface and texturing.

The work by Yan et al. [YMM10] proposes a number of surface surgery operations during integration to reveal the fractal geometry (thin sheet rotating around and tending to the attractors) of the strange attractors in 3D vector fields which previously were difficult to visualize. Their method consists of three major steps. First a polygonal surface is advected and deformed according to the vector field. This polygonal surface is initialized as some regular shape, such as a torus, which neglects the fractal dimension of the strange attractor. Second, due to the possible high distortion, the polygonal surface may need refinement. In this step, a GPU-based adaptive subdivision (i.e. edge division) of mesh is applied to preserve the necessary features. A mesh decimation on the CPU may also be conducted to reduce the resolution of uninteresting portion of the surface for memory efficiency. Third, a mesh re-tiling is performed to maintain the consistent triangulation of the thin sheet structure when approaching the attractor and to correct the self-intersection artifacts. This method has shown its utility through examples with strange attractors and is expected to apply to other integral surface computations.

Another extension to steam surface algorithms inspired by Peikert and Sadlo [PS09], is the work by Schneider et al. [SRWS10] whose algorithm detects singularities within the flow field and deals with them appropriately, rather than the current methods of continuous refinement or splitting the surface. The authors use a pre-processing step to generate the required topological information. The stream surface algorithm then detects intersection with the separating two dimensional manifold of a saddle point. The resulting surface will either follow a new direction appropriate to the local vector field when encountering a node saddle (See Figure 11) or split when encountering a spiral saddle.

2.1.2. *Streak/Time Surface Construction*

The main challenges addressed by the methods presented here are computational time and maintaining a continuous dynamic surface.

A **streakline** is the line joining a set of massless particles that have all been seeded successively over time at the same spatial location in time dependent flow. Dye steadily injected

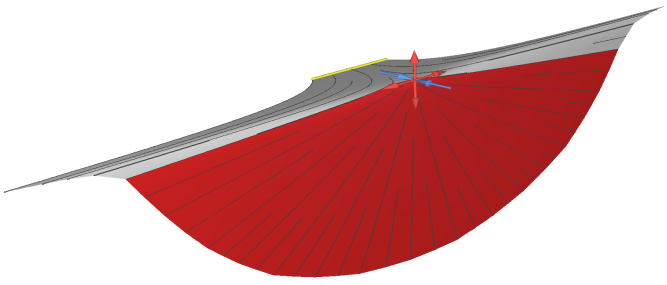

Figure 11: A stream surface in a linear vector field with highly diverging streamlines where the angle criterion (130 degrees) for splitting the surface fails because the surface does not run into the saddle. The red part of the surface would have been left out if the angle criterion were to be used. Image courtesy of D.Schneider et al. [SRWS10].

into the fluid at a fixed point extends along a streakline. If $\boldsymbol{v}$ is a three-dimensional vector field defined over a domain $\Omega \in \mathbb{R}^3$ and time interval $[T_0, T_1]$ to find the streakline $L(x, T_0, T_1; s)$ for particles seeded at x_0, starting at time T_0, as it appears at time T_1, we must solve separately the pathline equation for $I(x_0, t_0; t)$ for each $t_0 \in [T_0, T_1]$, and then let $L(x, T_0, T_1; s) = I(x_0, T_1 - s; s)$, for $s \in [0, T_1 - T_0]$. If seeded at the same location in a steady state flow field streamlines, pathlines and streaklines are identical.

A **streak surface** is the smooth union of streaklines from seeding locations along a continuous curve C. A streak surface K is the union of all particles emanating continuously from a parameterized curve $C(u)$ over time interval $[t_0, t_1]$ and moving with the flow from the time of seeding t. In terms of individual streaklines it can be described as:

$$K(u, T_0, T_1; t) := L(C(u), T_0, T_1; t)$$

Introducing the first streak surface approximation, Von Funck et al. [vFWTS08a] represent smoke structures as a triangular mesh of fixed topology, connectivity and resolution. The transparency of each triangle is represented by α where $\alpha = \alpha_{density}\ \alpha_{shape}\ \alpha_{curvature}\ \alpha_{fade}$. The density component $\alpha_{density}$ is a representation of the smoke optical model by considering the triangle primitive to be a small prism filled with smoke. The shape component α_{shape} of the triangle is defined as the ratio of its shortest edge to the radius of its circumcircle and represents its distortion. The curvature $\alpha_{curvature}$ is represented by the local mean surface curvature. The fade α_{fade} is defined as an increase in transparency over time. To effectively render the transparency at real-time frame rates a depth peeling algorithm is utilized. The authors demonstrate modifications to the algorithm to mimic smoke nozzles and wool tufts. This technique is the first step in generating streak surfaces, and addresses the occlusion issues associated with complex flow structures represented by surfaces.

Focusing on performance of large, time-varying vector fields, Krishnan et al. [KGJ09] propose a method for time and streak surface generation. Their approach enables parallelization by decoupling the surface advection and surface refinement. The authors build on work by Von Funck [vFWTS08a] with extensions to time surfaces while parallelising the pipeline and improving the meshing scheme using techniques described by Bridson [Bri03]. This paper describes the algorithm with respect to time surfaces and then explores the extension to streak surfaces. A time surface is generated in two steps. First each point of the initial surface mesh is advected over the next time interval. The mesh is then passed to the adaptation phase where three basic operations, edge split, edge flip, and edge collapse are applied to refine the surface. To prevent irreparable changes to the mesh, the integration time step is automatically chosen requiring that no vertex moves further from its current position than a predetermined function of maximum velocity. Streak surface evolution is refined using a similar approach accounting for the new particles seeded continuously from the seeding curve. The surface visualization uses a combination of texture mapping, lighting effects, and depth peeling for transparency. The use of these effects helps with occlusion and depth perception.

Continuing in the same theme, Burger et al. [BFTW09] describe two methods of streak surface construction for the visualization of unsteady flow. Building on work by von Funck et al. [vFWTS08a] the authors present the first real-time approach for adaptive streak surface integration and high quality rendering. See Figure 12. The first approach computes a quad-based surface where each quad patch is independent of all others. This independence enables parallel processing and rendering of each patch on the GPU. The refinement of these patches is performed independently and is based on an area criterion. If the criterion threshold is met the quad patch is split along the longest edge and its opposite edge, forming two new independent patches. The patches are rendered directly from the vertex buffer using a two-pass approach to fill gaps left by the refinement process.

The second approach computes a point-based interconnecting triangular mesh which is modified during the refinement process. Each advected timeline is stored in its own vertex buffer in order, and refined every time step. The refinement process is completed in three passes: time line refinement, connectivity update, streak line refinement. When inserting points the location is determined by fitting a cubic polynomial and bisecting it equally between the two diverging points. The connectivity of the mesh is then updated by searching the previous and next timelines for any given point's nearest neighbor. The third pass computes the maximum euclidean distance between neighboring timelines. The complete time line is then added or removed. The mesh is then rendered after a final pass computes the mesh triangulation.

Extending their work on quad-based stream and path surfaces, McLoughlin et al. [MLZ10] present a novel streak surface algorithm using quad primitives. The refinement of the surface is achieved by performing local operations on a quad-by-quad basis. Quads may be split or merged to maintain sufficient sampling in regions of divergence and convergence. Shear flow is handled by updating the topology of the mesh to maintain fairly regular quads. This method is designed for and implemented on the CPU and generally achieves interactive frame rates.

Following the collection of works which define methods for

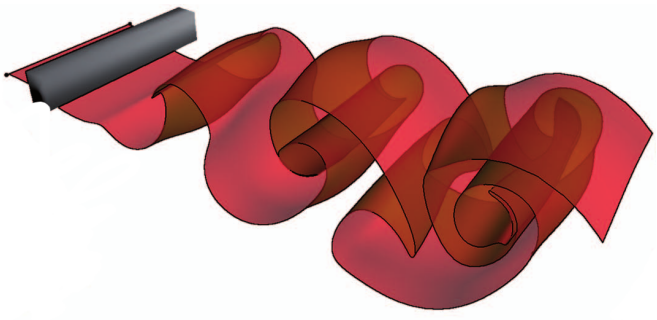

Figure 12: The visualization of a transparent streak surface rendered using depth peeling and generated on the GPU. Image courtesy of H.Theisel et al. [BFTW09]. © IEEE/TVCG.

Figure 13: Particle based surface visualization. Red particles correspond to points on the separating surface. Green particles serve as context information. They correspond to points on time surfaces, which are released from the planar probe at a fixed frequency. Image courtesy of H.Theisel et al. [FBTW10]. © IEEE/TVCG.

streak surface construction, Ferstl et al. [FBTW10] introduce real-time construction and rendering of surfaces, which represent Lagrangian Coherent Structures (LCS), in conjunction with the rendering of the streak surface particles. See Figure 13. This technique interactively displays the separation surfaces leading to new possibilities of studying complex flow phenomena. The user can interactively change the seeding parameters, and visually display the separation surfaces, resulting in a visually guided exploration of separation surfaces in 3D time-dependent vector fields.

LCS are computed by extracting the ridges in the finite time Lyapunov exponent (FTLE) field. The paper builds on the work by Sadlo and Peikert [SP07] who describe a filtered ridge extraction technique based on adaptive mesh refinement. The method enables a substantial speed-up by avoiding the seeding of trajectories in regions where no ridges are present or do not satisfy the prescribed filter criteria such as a minimum finite Lyapunov exponent.

The finite-time Lyapunov exponent (FTLE) [Hal01] quantifies the local of separation behavior of the flow. It is used to measure the rate of separation of infinitesimally close flow trajectories. The pathline solution $x = I(x_0, t_0; t)$ of Section 2.1.1 for fixed times t_0 and t can be considered as a flow map from a position x_0 to the pathline position x where it is advected by the flow at time t. Using the flow map, the *Cauchy-Green deformation tensor field*, $\boldsymbol{C}^t_{t_0}$ is obtained by left-multiplying the Jacobian matrix of the flow map with its transpose [Mas99]:

$$\boldsymbol{C}^t_{t_0}(\boldsymbol{x}) = \left[\frac{\partial(\boldsymbol{x}_0, t_0; t)}{\partial(\boldsymbol{x}_0)}\right]^T \left[\frac{\partial(\boldsymbol{x}_0, t_0; t)}{\partial(\boldsymbol{x}_0)}\right]$$

From this, the FTLE is computed by:

$$\boldsymbol{FTLE}^t_{t_0}(\boldsymbol{x}) = \frac{1}{t - t_0} \ln \sqrt{\lambda_{max}(\boldsymbol{C}^t_{t_0}(\boldsymbol{x}))}$$

where $\lambda_{max}(\boldsymbol{M})$ is the maximum eigenvalue of matrix $\boldsymbol{M}$ [Hal01].

FTLE requires the choice of a temporal window, the effect of a change in the time-window length has not been studied sufficiently [PPF*10]. A common use of FTLE is to extract *Lagrangian Coherent Structures* (LCS). LCS are extracted from an FTLE field by ridge extraction [SP07].

2.2. Implicit Construction Techniques

This section reviews a set of surface construction techniques which are described by solving some function of the underlying flow field. The motivation for this type of technique include avoiding compound error associated with integration schemes and meshing challenges resulting from convergent, divergent and shear flow. The work in this area is limited to stream surface representations in steady-state velocity data.

van Wijk [vW93] introduces a method for the global representation of the stream surfaces as implicit surfaces $f(x) = C$. Once f is defined at the boundary, the definition is then extended to the interior of the domain by specifying that it is constant on streamlines. C can be varied to efficiently generate a family of stream surfaces. The originating curves are defined at the boundary by the value of f. This method greatly differs from the advancing front methods introduced by Hultquist [Hul92]. Two methods are presented to derive f; The first is based on solving the convection equations, and the second is based on backward tracing trajectories from grid points. The 3D stream function defines a scalar field from which traditional isosurface extraction techniques are then used to create the stream surfaces.

Taking this concept a step further, Westermann et al. [WJE00] present a technique for converting a vector field to a scalar level set representation. See Figure 14. An analysis of the subsequent distorted level-set representation of time surfaces is conducted before combining geometrical and topological considerations to derive a multiscale representation. This is implemented with the automatic placement of a sparse set of graphical primitives, depicting homogeneous streams within the fields. The final step is to visualize the scalar field with isosurfaces. The advantage of this technique is full domain coverage as the van Wijk [vW93] method constructs surfaces only where intersections with the boundaries occur.

With a different approach van Gelder. [Gel01] introduce a semi-global method which does not suffer from the compound error from integral surface generation or computational overhead and error seen in the global approaches. Stream surfaces are constructed on 3D curvilinear grids which satisfy the con-

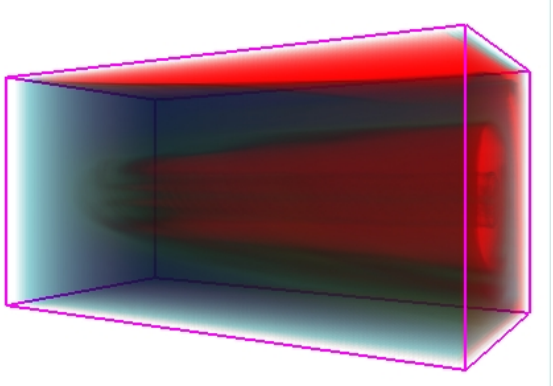

Figure 14: Dense flow fields are first converted into a scalar field, and then displayed and analyzed by means of level-sets in this field. Image courtesy of R.Westermann et al. [WJE00].

straints of a region expressed as integrals, instead of solving a local ordinary differential equation. The constraints are expressed as a series of solvable quadratic minimization problems. The solution exploits the fact that the matrix of each quadratic form is tridiagonal and symmetric. The author describes the transformation of the curvilinear grid into parameter space to simplify the stream surface construction problem.

2.3. Topological Surface Construction Techniques

The challenge of topology based methods is to separate or segment the flow into areas of similar behavior. As part of this process singularities and separatrices are extracted from the flow field. In steady-state flow the separatrices are stream surfaces. The topological structures can also useful for supporting other flow visualization methods, and is the inspiration for techniques is this section.

Theisel et al. [TWHS03] present an approach for constructing saddle connectors in place of separating stream surfaces is a significant effort to help alleviate occlusion. A **saddle connector** is a streamline which connects two saddle points. Building on work by Theisel and Seidel [TS03] the authors apply saddle connectors in three dimensions. This work is extended by Weinkauf et al. [WTHS04] who introduce the concept of separating surfaces originating from boundary switch curves. A **boundary switch curve** is a curve generated at the domain boundary where inflow changes to outflow or vice versa e.g., flow is parallel with the boundary surface. This is achieved by joining saddle points to boundary switch curves, or between each other, using a type of streamline called boundary switch connectors. The idea of using streamline connectors in place of separating surfaces reduces visual clutter as can be seen in Figure 15.

Inspired by Theisel and Seidel's work on tracking features in 2D [TS03], Theisel et al. [TSW*05] introduce a method for visualizing the propagation of vortex core lines over time. The contextual surfaces are shown emanating from the vortex core lines in Figure 16. Two 4D vector fields are computed which

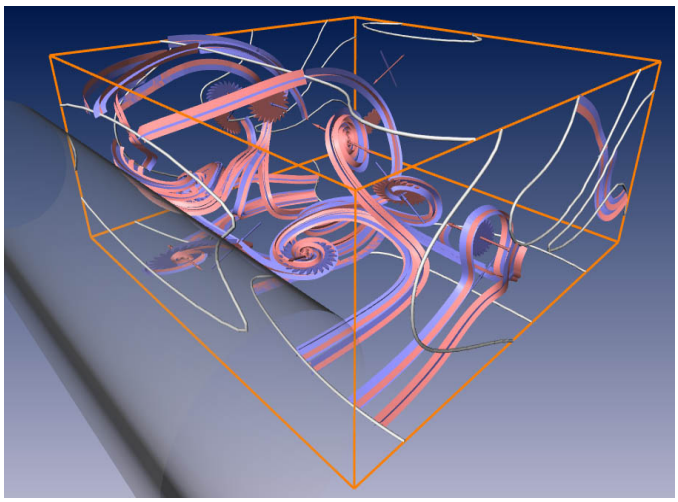

Figure 15: Topological skeleton showing saddle connectors, singularities and boundary switch connectors. Image courtesy of H.Theisel et al. [WTHS04].

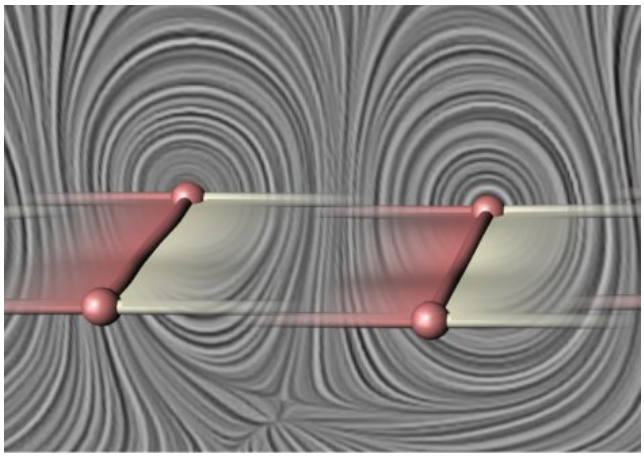

Figure 16: Time surfaces shown as contextual information emanating from the visualized vortex core lines. Image courtesy of H.Theisel et al. [TSW*05].

act as feature flow fields such that their integration surfaces (e.g. stream surfaces) provide the vortex core structures. The feature flow field is equivalent to the parallel vector (PV) approaches by Peikert and Roth [PR99]. In addition, this work describes a method to extract and classify local bifurcations of vortex core lines in space-time through the tracking and analysis of PV lines in the feature flow field.

Avoiding the integration of hyperbolic trajectories by replacing them with intersections of LCS while utilizing LIC to reveal the tangential dynamics, Bachthaler et al. [BSDW12] stack 2D vector fields according to time to generate a 3D space-time vector field. The LCS ridge structures are computed from an FTLE scalar field generated in three dimensions. The hyperbolic trajectories are mapped to saturation. Visualizing the LCS dynamics the authors apply the method described by Weiskopf et al. [WE04]. To address the problem of occlusion in the space-time visualization of the LCS, the paper describes restricting the visualization to bands around the LCS intersection curves. The authors adopt the concept of hyperbolic trajectories and space-time streak manifolds.

3. Rendering Flow on Surfaces for Visualization

This section presents a survey of techniques that enhance the rendering and visualization of surfaces used for flow visualization. Figure 17 show a chronological flow of techniques demonstrating the child-parent relationships and key ideas that are progressed. The charts also show the originating work, and where key work is continued.

We start this section with a discussion about the conceptual differences of Parameter Space and Image Space techniques. We then examine the rendering techniques. The techniques in this section are classified into Direct, Geometric and Texture-based. The texture-based subsection is further divided into static and dynamic type textures.

Parameter Space and Image Space Techniques

One approach to applying texture properties on surfaces is via the use of a parameterization. Applying textures to surfaces

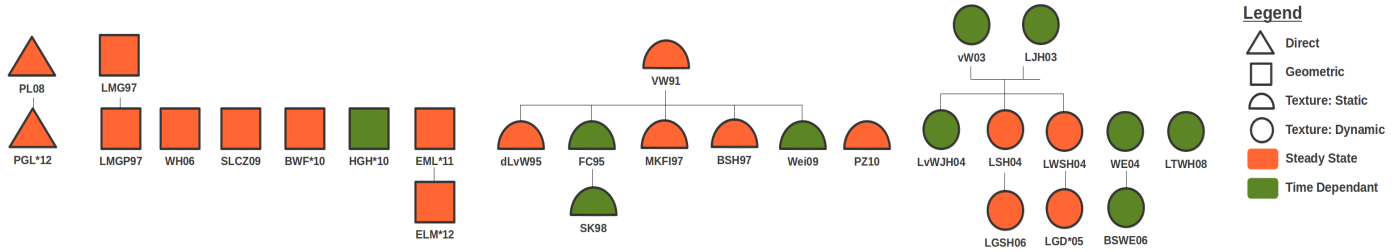

Figure 17: The classifications of rendering techniques show a chronological flow of work from author to author. The child-parent relationships indicates the key ideas that are progressed. The flow of work diverges and in some cases converge as new concepts are built on top of previous ideas. The charts also show the originating work, and where key work is continued.

becomes particularly suitable when the whole surface can be parameterized globally in two dimensions as shown by Forssell and Cohen [FC95]. The drawbacks with this approach include challenges such as distorted textures as a result of the mapping between object space and parameter space. A global parameterization for many types of surface is not available such as isosurfaces generated from marching cubes algorithms.

A more recent approach is the use of image space techniques to accelerate computation. The general approach is to project the surface geometry to image space and then apply a series of image space techniques. These techniques can range from advecting dense noise textures [LJH03] to rendering attributes from the underlying data to the surface such as streamlines [SLCZ09], or illustrating various perceptual attributes of the surface such as silhouette edge highlighting [HGH*10].

3.1. Direct Rendering Techniques

Direct visualization techniques are the most primitive methods of flow visualization. Typical examples involve placing an arrow glyph at each sample point in the domain to represent the vector data or mapping to some scalar attribute of the local vector field. Direct techniques are simpler to implement and enable direct investigation of the flow field. However, these techniques may suffer from visual complexity and imagery that lacks in visual coherency. They also suffer from serious occlusion problems when applied to 3D data sets. This idea provides motivation for the work classified in this section.

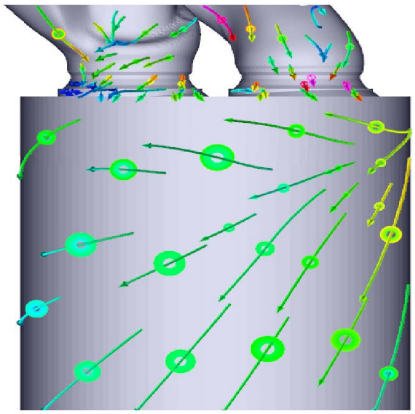

Figure 18: The combination of velocity-range glyphs (Disk glyph) and streamlet tubes (Arrow glyph) is applied to provide both detailed (the range glyph) and summary (the streamlet) information of the vector field direction. Image courtesy of R.S.Laramee et al. [**?**].

Extending the direct visualization paradigm, combining it with clustering techniques and utilizing image space methods, Peng and Laramee introduce a glyph placement technique performed in image-space which visualizes boundary flow [PL08]. This concept builds on work by Laramee et al. [LvWJH04] by projecting the visible vector field from object-space to image-space, then generating evenly-spaced glyphs on a regular grid. The glyphs are a clustered approximation of the underlying vector field mesh resolution. Calculated on the fly, this algorithm is efficient and can handle large unstructured, adaptive resolution meshes.

Following their previous image space work [PL08], Peng et al. [PGL*12] present a novel, robust, automatic vector field clustering algorithm that produces intuitive images of vector fields on large, unstructured, adaptive resolution boundary meshes from CFD. Their bottom-up, hierarchical approach is the first to combine the properties of the underlying vector field and mesh into a unified error-driven representation. See Figure 18. Clusters are generated automatically, no surface parameterization is required, and large meshes are processed efficiently. Users can interactively control the level of detail by adjusting a range of clustering distance measures. This work also introduces novel visualizations of clusters inspired by statistical methods.

3.2. Geometric Illustration Techniques

In this section we study a range of geometric illustration techniques. The purpose of illustrative techniques is to enhance the often complex surface representations commonly found in flow data. A general solution to this problem is to use transparency. With surfaces we have additional options. Surface primitives have well defined normals thus they can offer perceptual advantages including: lighting and shading which provide intuitive depth cues, the ability to texture map, image space techniques such as silhouette edge highlighting and the placement of additional geometry on the surface.

Löffelmann et al. [LMGP97] introduce methods for placing arrow images on a stream surface using a regular tiling, and offsetting portions of the surface, leaving arrow shaped holes which alleviate occlusion. The authors are inspired by

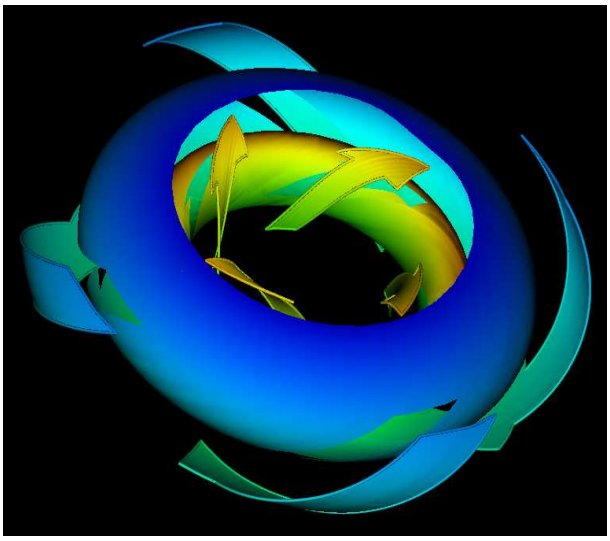

Figure 19: Stream arrows offset from a stream surface. The portion offset leaves holes, reducing occlusion by the surface. Image courtesy of H.Hauser et al. [LMG97].

Shaw's [AS92] artistic approach to the use of stream surfaces to help with occlusion within the dynamical systems. The authors extend this work as their method provides unsatisfactory results in areas of divergence and convergence [LMG97]. The arrows could become too small or too large to provide quality stream surface visualizations. The proposed novel enhancement is a hierarchical approach which stores a stack of scaled stream arrows as textures applying the suitably scaled arrow to the surfaces maintaining consistent proportionality. See Figure 19.

Weiskopf and Hauser [WH06] explore a GPU based shading method for the evaluation and visualization of surface shapes. Cycle shading and hatched cycle shading methods extend the idea of natural surface highlights in a regular repeating pattern. This technique can be used where Phong illumination is appropriate as curvature or mesh connectivity information is not required. To reduce depth induced aliasing an approach similar to Mip Mapping is introduced. The effectiveness of cycle shading for the assessment of surface quality is demonstrated by a user study.

Image-based streamline seeding on surfaces by Spencer et al. [SLCZ09] utilize the perceptual properties of surfaces by displaying the local vector field as as evenly-spaced streamlines [JL97], while maintaining lighting, shading and other useful depth cues. This image-based approach is efficient, handles complex data formats, and is fast. The evenly spaced streamlines are produced by scanning the image, checking the z depth of the fragments (where the fragments z depth is obtained from the depth buffer), and initializing the seeding where the fragment z depth is non zero (i.e., the fragment is part of some object in the field of view) and no other streamline is closer than the user specified minimum distance.

The illustration of stream surfaces is explored by Born et al. [BWF*10] who are inspired by traditional flow illustrations drawn by Dallmann, Abraham and Shaw in the early 1980's. The authors describe techniques for silhouettes and feature lines, halftoning, illustrative surface streamlines, cuts, and slabs to visually describe the surface shape. User interactive exploration with these techniques allows insight into the inner flow structures of the data. Implemented on the GPU, this work requires no pre-processing for the final visualizations.

Further exploration of illustration techniques is performed by Hummel et al. [HGH*10] who present a novel application of non-photorealistic rendering methods to the visualization of integral surfaces. See Figure 20. The paper examines how transparency and texturing techniques can be applied to surface geometry to enhancing the users perception of shape and direction. They describe angle, normal variation, window and silhouette transparencies with adaptive pattern texturing. The authors present this work in a combined rendering pipeline implemented on the GPU. This maintains interactivity and removes the need for expensive surface processing to generate visualizations.

Defining seeding curves at the domain boundaries from isolines generated from a scalar field, Edmunds et al. [EML*11] automatically seed and generate stream surfaces integrated through the 3D flow. The scalar field is derived from the exit trajectory of the flow from the domain capturing the topological constructs of the flow at the boundary. See Figure 21. This work also discusses strategies for resolving occlusion resulting from seeding multiple surfaces.

Using a vector field clustering strategy combined with a derived curvature field to define stream surface seeding location and seeding curve contour, Edmunds et al. [ELM*12] use a range of illustrative techniques to enhance the visualization. The illustrations, as with surface placement, specifically target providing the required visual information for the engineer to accurately interpret the flow characteristics. Figure 22 demonstrates the focus and context approach to the surface placement and visualizes the applied illustrative techniques.

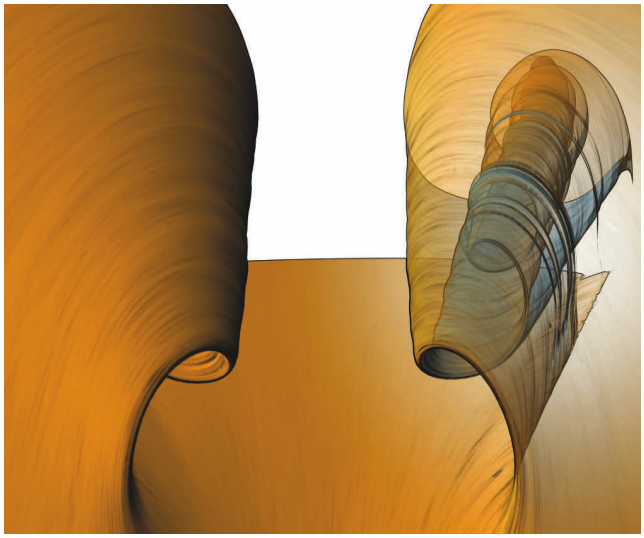

Figure 20: Illustrative techniques applied to a stream surface. This technique shows windowed transparency and two sided surface coloring. Image courtesy of C.Garth et al. [HGH*10].

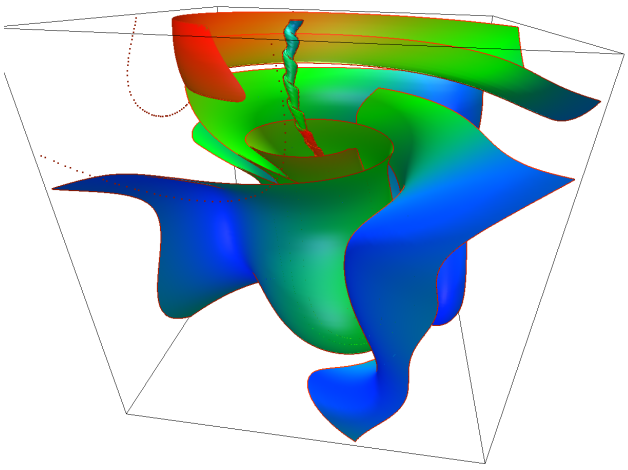

Figure 21: A set of streamsurfaces seeded automatically on a tornado simulation. The image shows surfaces with edge highlighting improving the perception and allows insight into the behavior of the inner flow structures. Image courtesy of M.Edmunds et al. [EML*11].

3.3. Texture-Based Techniques

In this subsection we present an overview of texture-based visualization algorithms. This work is subdivided into Static Texture and Dynamic Texture techniques. Dense texture-based techniques exploit textures to display a representation of the flow and avoid the seeding challenges. The general approach uses a texture with a filtered noise pattern which is smeared and stretched according to the local velocity field. Texture-based approaches provide dense visualization results, show intricate detail, and capture the characteristics of the flow even in complex areas of flow such as vortices, sources, and sinks.

With standard spot noise, a texture [vW91] can be characterized by a scalar function $f(x)$. A spot noise texture is defined as:

$$f(x) = \sum a_i h(x - x_i)$$

in which $h(x)$ is called the spot function. It is a function everywhere zero except for an area that is small compared to the texture size. a_i is a random scaling factor with a zero mean, x_i is a random position. In non-mathematical terms: spots of random intensity are drawn and blended together on random positions on a plane.

3.3.1. Static Texture

The concept of the static texture in the context of dense texture-based methods refers to the non-animation of the flow in the final visualization. Although the images are static in nature the algorithms can be applied to both steady-state or time-dependent flow data. One of the early examples of this classification of techniques is presented by de Leeuw and van Wijk, who first use the basic textured spot noise principle for visualizing vector fields on surfaces and steady flow [dLvW95]. The authors build on work by van Wijk [vW91] extending it in four main areas; using a parameterized stream surface to deform the spot polygon adapting it to the shape of the local velocity field, using a negative Gaussian high-pass filter to remove the low frequency components of the spot noise textures, using the graphics hardware to conduct a series of transformations of the matrix stack, normally used for the viewing pipeline, for improved interactive animation, and synthesizing spot noise on highly irregular grids by pre-distorting the spots in texture space.

Extending the original LIC method [CL93] to the visualization of flows on curvilinear surfaces, Forssell and Cohen [FC95] introduce a technique to visualize vector magnitude (velocity) as variable speed flow animation. In order to extend LIC to a parameterized surface the surface geometry and related vector field information is mapped to parameter space. The LIC image in parameter space is computed and mapped back onto the physical surface through an inverse mapping. The new visualization takes into account the vector magnitudes by varying the speed of the phase shift in the animation, and handles unsteady flow by conducting convolution along the pathlines instead of streamlines. However, the direct convolution following the pathlines of the particles can lead to visual ambiguity since the forward and backward pathlines through a pixel need not match.

Mao et al. [MKFI97] extend the original LIC method by applying it to surfaces represented by arbitrary grids in 3D. Former LIC methods targeted at surfaces were restricted to structured grids [For94], [FC95], [SJM96]. Also, mapping a computed 2D LIC texture to a curvilinear grid may introduce distortions in the texture. The authors propose solutions to overcome these limitations. The principle behind their algorithm relies on solid texturing [Pea85]. The convolution of a 3D white noise image, with filter kernels defined along the local streamlines, is

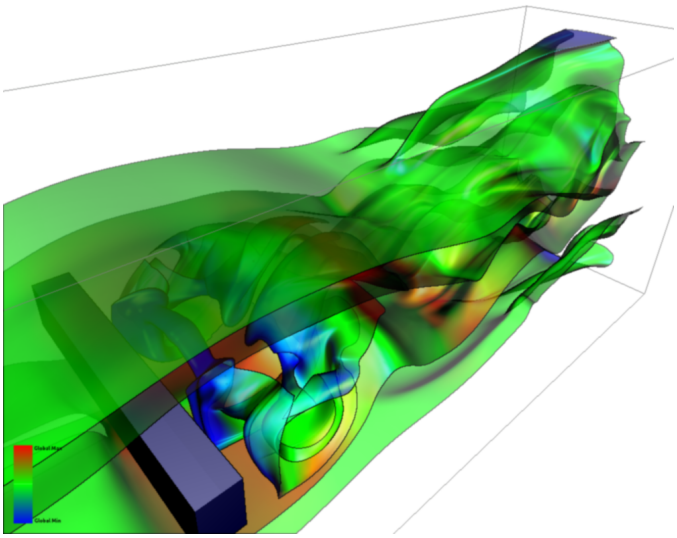

Figure 22: This visualization demonstrates the feature-centered approach to stream surface placement. The vortex shedding is represented by the stream surfaces along with the contextual information of the complete flow field. Image courtesy of M.Edmunds et al. [ELM*12]. This is a direct numerical Navier Stokes simulation by Simone Camarri and Maria-Vittoria Salvetti (University of Pisa), Marcelo Buffoni (Politecnico of Torino), and Angelo Iollo (University of Bordeaux I) [CSBI05] which is publicly available [Int]. We use a uniformly resampled version which has been provided by Tino Weinkauf and used in von Funck et al. for smoke visualizations [vFWTS08b].

performed only at visible ray-surface intersections.

Battke et al. [BSH97] introduce a fast LIC technique for surfaces in 3D space. Instead of using 2D global parameterization, which is limited to curvilinear surfaces, the authors propose a 3D local parameterization scheme. In this way multiple interconnected surfaces can be handled. An initial tessellation of the surface is conducted and local euclidean texture coordinates are defined for each triangle. LIC textures are then computed by projecting a locally scaled vector field onto each planar triangle. To ensure a smooth transition across the tessellated surface, streamlines are followed across neighboring triangles, which results in a smooth textured surface. The textures are packed efficiently into texture memory by arranging similar patches in rows and then proceeding using a greedy algorithm. The result is a smooth interactive visualization of LIC on surfaces in 3D flow.

Shen et al. [SK98] further improve [FC95] by eliminating the artifacts caused by the ambiguity of backward and forward convolution, directly following pathlines of the flow. This enables the visualization of unsteady flow on surfaces. In order to achieve this a time accurate scattering scheme, compared to the gathering scheme in the original LIC, is used to model the texture advection. More specifically, every pixel in the image space *scatters* its color value to its neighbors following the pathline of the particle at the center of the pixel in a small time step. The resulting color for each pixel is the weighted sum of all the contributions from its neighbors by considering their ages. In order to maintain the temporal coherence, this method uses the previous result as input for the next iteration. To improve the performance a parallel framework is also discussed.

Weiskopf [Wei09] introduces iterative twofold convolution as an efficient high-quality two-stage filtering method for dense texture-based vector field visualization. The first stage applies a Lagrangian-particle-tracing-based user-specified compact filter kernel. The second stage applies iterative alpha blending for large-scale exponential filtering. A discussion of sampling rates demonstrates this order of convolution operations facilitates large integration step sizes. Twofold convolution can be applied to steady and unsteady vector fields, dye and noise advection, and surfaces. This work has the potential to be incorporated in existing GPU-based 3D vector field visualization methods.

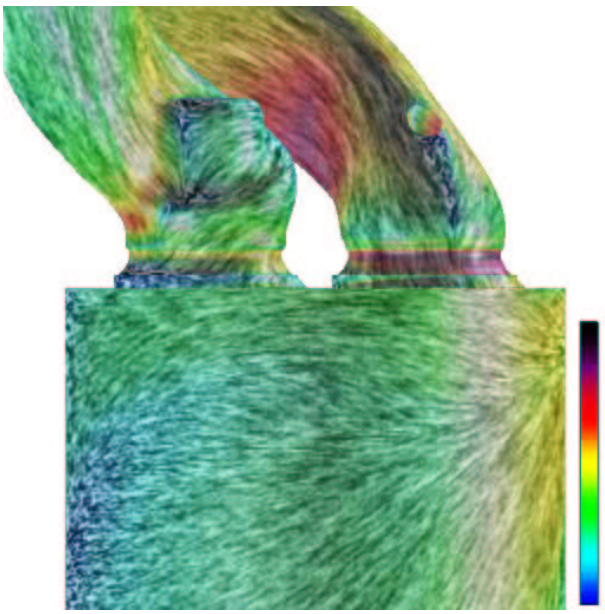

Figure 23: A side view of the surface of a 221K polygonal intake port mesh. The visualization shows ISA applied to the flow simulation data. Color is mapped to velocity. Image courtesy of R.S.Laramee et al. [LvWJH04].

Advancing techniques which illustrate flow on surfaces, Palacios et al. [PZ10] present an algorithm for the visualization of N way rotationally symmetric fields (N-RoSy) on 2D planes and surfaces. The basic idea is to decompose the N-RoSy field into multiple distinct vector fields. Together these decomposed vector fields capture all N directions at each point. The LIC method is then adopted to compute a flow image for each vector field. These flow images are blended to obtain the final result. A simple probability model based on the correction of normally distributed random variables is applied to compensate for the loss of contrast caused by the blending of images. This algorithm is easily extended to surfaces with an additional transformation to combat the artifacts caused by the perspective difference of the N-direction vectors on the tangent plane and the view plane, respectively.

3.3.2. Dynamic Texture

Dynamic textures in the context of dense texture-based methods refers to the animation of the flow in the final visualization. The technique generally known as texture advection is applied to steady state and time dependent flow data. The main challenges of this literature addressed here are computational time and visualization of unsteady flow.

Laramee et al. [LJH03] and van Wijk [vW03] present new methods for the synthesis of dense textures on surfaces, bringing the concept into the realms of interactivity. These techniques follow previous work in this area in which van Wijk presents a method for producing animated textures for the visualization of 2D vector fields [vW02] along with methods by Jobard et al. [JEH01]. This technique uses a different order of visualization operations than traditional work. The surface geometry and related vector field information are projected to image space where the texturing is then applied.

Following this work, Laramee et al. [LvWJH04] perform a comparative analysis of two techniques: Image Space Advection (ISA) [LJH03], and Image Based Flow Visualization for Surfaces (IBFVS) [vW03]. The authors discuss the best application of each technique, explaining that IBFVS is a good choice where the pixel to polygon ratio in image space is high, and the ISA technique is better for larger meshes in which many polygons cover a single pixel or where there are many occluded polygons, see Figure 23.

Laramee et al. [LSH04] then extend current texture advection techniques to present a novel hybrid method in which a dense texture-based flow representation is applied directly to isosurfaces. The authors build on work by van Wijk et al. [vW03] and

Laramee et al. [LJH03] whose methods are suitable for the visualization of unsteady flow on surfaces. This paper addresses issues associated with applying texture advection to isosurfaces where there may be a component of the velocity that is normal to the surface, the perceptual challenges such as occlusion, and issues related to re-sampling of the 3D vector field to 2D image-space. The authors use a normal mask to dim areas of flow which have strong cross flow components at the isosurface, and use a clipping plane to remove sub sets of the geometry to help reduce occlusion.

The proposed texture advection technique by Weiskopf and Ertl [WE04] depends on Lagrangian particle tracing, which is simultaneously computed in object space and in image space. This approach builds on previous texture advection work, introducing frame-to-frame coherence when the camera position is changed. The input noise is modeled as a 3D texture which is scaled appropriately in object space. The authors propose different color schemes to improve the visualization of shape and flow. This technique is implemented on the GPU and supports interactive visualization.

Studying the application of image-based techniques, Laramee et al. [LWSH04] visualize the flow characteristics of the cycle of an engine cylinder, focusing on swirl and tumble. A variety of techniques are demonstrated including iso-surfaces enhanced with an image-based technique. As another application of visualization of automotive CFD simulations, Laramee et al. [LGD*05] analyze fluid flow within an engine cooling jacket. Again, a variety of visualization methods are demonstrated including surface-based methods such as stream surfaces and isosurfaces.

Presenting a new extension to previous work by Laramee et al. [LSH04] and Garth et al. [GTS*04], Laramee et al. [LGSH06] propose a hybrid method where a dense texture-based flow representation is applied directly to stream surfaces. This conveys features of the flow that otherwise would not be seen using stream surfaces alone. Texture advection applied to stream surfaces avoids issues inherent with isosurfaces such as flow normal to the surface.

Modifying the texture advection approach by [WE04] to run on a highly parallelized GPU cluster, Bachthaler et al. [BSWE06] introduce an algorithm which is scalable according to the number of the cluster nodes. The authors employ a sort-first strategy with image space decomposition for LIC workload distribution. A sort-last approach with an object-space partitioning of the vector field increases the amount of available GPU memory. This work addresses the challenges of memory accesses locality caused by particle tracing, dynamic load balancing to support view changes, and combining image-space and object-space decomposition. For future work, parallel rendering could be extended to the projection of the surface geometry itself in order to visualize extremely large surface meshes.

Li et al. [LTWH08] propose a global texture advection and synthesis method for flow visualization on surfaces. It solves the problem of inconsistent texture correspondence between visible and invisible parts of surfaces in image-based approaches. In order to achieve such global continuous texturing of flow, surfaces are firstly segmented into patches and parameterized. These patches are then overlapped by a small extent with adjacent ones and packed into a uniform texture space. Next, a 2D dense texture based flow visualization technique [LHD*04] such as Graphics Processing Unit Line Integral Convolution (GPULIC) or Unsteady Flow Advection-Convolution (UFAC) is employed to synthesize the flow texture in texture space. The overlapping patches guarantee texture continuity across their discontinuous borders. The synthetic flow texture is mapped to the surface, compositing the overlapping regions to avoid artifacts in texture patterns caused by inconsistent partial particle traces.

4. Discussion and Future Challenges

No individual technique provides optimal results for all problems or phenomena, and many of the techniques overlap in their approach. The best technique depends on several factors such as the purpose of the visualisation; presentation, detailed analysis, or exploration, and the interest of the analyst or engineer studying the data. This survey provides a study of a variety of techniques and approaches to cater for most eventualities when studying 3D vector field simulations. However, there are some topics of study which could potentially benefit from further examination, experiment, and verification. We summarise and discuss these challenges. Some of these challenges are more general, while others are specific to this survey.

- Large, Unstructured, Time-Dependent CFD Data.
- Visualizing Error and Uncertainty.
- Perceptual Challenges.
- Information Content.
- Implicit/Topological/Direct Techniques for Unsteady Flow.
- Interactive Construction and Rendering of Time surfaces.
- Human Centered Evaluation of Flow Visualization Techniques.

Large, Unstructured, Time-Dependent Grids:. A significant effort has gone into the study of processing large, unstructured data [USM96] [SBH*01] while more recently the focus has shifted towards studying highly parallel GPU based implementations along with utilization of scalable GPU based technologies [BSWE06]. With the introduction of advanced GPU technologies a re-examination of techniques with a focus parallelization has occurred with the introduction of new challenges. With modifications required to particle tracing methods [STWE07], restrictions regarding triangulation techniques [BFTW09], and scalability are all important topics for further parallel/GPU based research.

Visualizing Error and Uncertainty:. The numerical integration of particle trajectories sampled using a piecewise interpolation of the underlying data introduces an accumulative error. This error is often overlooked or misunderstood [Hul92] described in section 1.3. Another source of error is the sampling density of the advancing front. If an insufficient density is maintained then the surface may fold or become excessively coarse producing inaccurate surface representations. The difficulty in determining the correct splitting conditions, allowing the surface trajectory to split around some boundary or critical point, is also a potential source of inaccurate representations of the flow. Distortion of the meshing techniques as described by McLoughlin at al. [MLZ10], [MLZ09], regarding distortion of non-planar quads, distortion of triangles in highly diverging flow regions or areas of high shear strain between adjacent time lines according to Berger et al. [BFTW09], and the visual representation of the point strategy by Schafhitzel et al. [STWE07], are areas of future work which should be examined. The current visualizations computed with these error-prone methods could lead to misleading information. Therefore, a possible future work path is to develop proper techniques to visualize these different types of uncertainty.

Perceptual Challenges:. Rendering too many surfaces causes perceptual problems such as occlusion and visual complexity. Garth et al. [GTS*04]showed the placement of stream surfaces is important in the reduction of visual clutter. See section 2 for more on this. Effective and efficient placement strategies not only for static, but for dynamic data are areas of work which require significant study. The topological constructs available in steady flow can easily disappear in dynamic flow. Therefore, a consideration of the time period would likely be necessary. The work on topology aware seeding is an excellent starting point along this direction [PS09, SRWS10], and an initial study of stream surface placement by Edmunds et al. [EML*11] and [ELC*12] attempts to seed surfaces at the domain boundary based on characteristics of the flow exiting the domain. The results show limited success as the technique only works where flow exits the boundary. Further work studying the placement of surfaces to best represent the characteristics of the flow is the feature centered approach by Edmunds et al. [ELM*12]. One possible next goal for this area of research to develop a knowledge-assisted seeding strategy for better extracting more informative integral surfaces.

Born et al. [BWF*10] and Hummel et al. [HGH*10] extend the use of image space and GPU technologies to significantly enhance the geometric structure of surfaces, while representing attributes of the vector field to improve the perception of flow characteristics. Extending these algorithms to dynamic surfaces such as streak and time surfaces is an area of work yet to be studied in depth. Visualizing flow characteristics on the constantly changing surface geometry, which is not necessarily tangential with the underlying vector field, is one such challenge.

Information Content:. This future work direction and those that follow are more specific to this survey. Mapping glyphs to surfaces to allow users to annotate some feature of the surface/flow, or allowing the user to switch between different visual or attribute cues are interesting directions of future work [BWF*10]. Palacios et al. [PZ10] wish to investigate efficient contrast adjustment when the input images are not gray-scale and have different hues. One example of this is to visualize both the major and minor eigenvector fields of a second-order tensor. Also of interest to the authors are new decomposition strategies that will lead to fewer images to blend, thus increasing the interactivity.

A largely unexplored area of further research is the effective visualization of multivariate data attributes normally associated with engineering simulations. This includes not only the visualization of the attributes themselves, but also the effective placement of the surfaces to best represent the information of interest.

Implicit/Topological/Direct Techniques for Unsteady Flow:. The work by van Wijk [vW93], Westermann et al. [WJE00], and van Gelder [Gel01] focuses on implicitly visualizing some scalar function of the underlying flow field. Section 2.2 provides more detail. This approach avoids the numerical integration error, refinement schemes, and user defined thresholds and parameters. However, these techniques do raise new challenges such as higher dimensionality e.g., temporal data, and resolving areas of highly turbulent flow are problems which still remain unsolved.

The work by Theisel et al. [TWHS03] and Weinkauf et al. [WTHS04] use topological constructs to generate visualizations of separatrices, and singularities within the domain. These techniques are applied only to steady-state flow data. Theisel et al. [TSW*05] make the step into temporal data. Ferstl et al. [FBTW10] describe fuzzy ridge structures undergoing frequent topology changes with the FTLE in turbulent areas of flow, which may cause visual clutter, providing scope for future work.

The direct methods by Peng et al. [PL08] [PGL*12] deal with unstructured meshes, with challenges arising from both the re-sampling performance time and perceptual issues. Future work includes the investigation of different measures for the derivation of mesh resolution, and there is great scope to extend this work to temporal data.

Interactive Construction and Rendering of Time surfaces:. There has been little work studying the challenge of time surfaces in time-dependent flow with the exception of Krishnan et al. [KGJ09] (Section 2.1.2). This work handles the test cases well, maintaining a well-formed mesh. A study of extending this work to scalable parallel environments, demonstrated on a wider range of flow types e.g., highly rotational flow, could be conducted with a focus on representation error compared to the ground truth case.

Human Centered Evaluation of Flow Visualization Techniques:. Laidlaw et al. [LKJ*05] conducted an extensive user study of two-dimensional visualization techniques discussing their relative merits for visualizing particular characteristics of the

flow. Further two-dimensional user studies have been conducted more recently by Liu et al. [LCS*12].These works would not necessarily translate directly to three dimensions.

One of the key areas of future work which generally has received little attention is a thorough study of three-dimensional visualization techniques, outlining which technique is best under a given circumstance for providing the required visual information. The work by Forsberg et al. [FCL09] is a step in the right direction performing a user study of three-dimensional flow visualization examining line and tube representations of integral curves with both monoscopic and stereoscopic viewing.

This type of study performed for surface-based flow visualization examining the effective construction, placement, and rendering of surfaces to best handle, and effectively represent characteristics of a given 3D flow field is an important possible future work direction.

5. Conclusions and Summary

Despite the great amount of progress that has been made in the field of surface-based flow visualization over the last two decades, a number of challenges remain. Challenges such as surface placement, speed of computation, perception, and evaluation remain key topics for further research.

We have introduced a novel classification scheme based on challenges including construction, rendering, and data dimensionality. This scheme lends itself to an intuitive grouping of papers that are naturally related to each other. Our classification highlights both unsolved problems and mature areas where many solutions have been provided.

The result is an up-to-date overview of the current state of the art in this rapidly evolving branch of research.

6. Acknowledgments

The Authors would like to thank the Department of Computer Science at Swansea University, UK, and the Department of Computer Science at the University of Utah, US, and the Department of Computer Science at the University of California, Davis, US, and the Department of Computer Science at Oregon State University, US, and the Center for Coastal and Ocean Mapping, University of New Hampshire, US. Guoning Chen was supported by DOE SciDAC VACET. The Authors would also like to thank Phillip James and James Walker at Swansea University, UK, for there proof reading efforts.

References

[ANS10] ANSYS UK: Fluent Engineering Simulation. http://www.fluent.co.uk/, 2010. Accessed: 08th August 2010.

[AS92] Abraham R. H., Shaw C. D.: *Dynamics - the Geometry of Behavior*. Addison-Wesley, 1992.

[Bel87] Belie R. G.: Some Advances in Digital Flow Visualization. In *AIAA Aerospace Sciences Conference* (Reno, NV, January 1987), AIAA, pp. 87–1179.

[BFTW09] Buerger K., Ferstl F., Theisel H., Westermann R.: Interactive Streak Surface Visualization on the GPU. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1259–1266.

[Bri03] Bridson R. E.: *Computational Aspects of Dynamic Surfaces*. PhD thesis, Stanford, CA, USA, 2003.

[BSDW12] Bachthaler S., Sadlo F., Dachsbacher C., Weiskopf D.: Space-Time Visualization of Dynamics in Lagrangian Coherent Structures of Time-Dependent 2D Vector Fields. *International Conference on Information Visualization Theory and Applications* (2012), 573–583.

[BSH97] Battke H., Stalling D., Hege H.: Fast Line Integral Convolution for Arbitrary Surfaces in 3D. In *Visualization and Mathematics* (1997), Springer-Verlag, pp. 181–195.

[BSWE06] Bachthaler S., Strengert M., Weiskopf D., Ertl T.: Parallel texture-based vector field visualization on curved surfaces using GPU cluster computers. In *Eurographics Symposium on Parallel Graphics and Visualization (EGPGV06) , pages 75-82. Eurographics Association, 2006* (2006), Universitt Stuttgart.

[BW08] Bachthaler S., Weiskopf D.: Animation of Orthogonal Texture Patterns for Vector Field Visualization. *IEEE Transactions on Visualization and Computer Graphics 14*, 4 (July 2008), 741–755.

[BWF*10] Born S., Wiebel A., Friedrich J., Scheuermann G., Bartz D.: Illustrative Stream Surfaces. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (2010), 1329–1338.

[CL93] Cabral B., Leedom L. C.: Imaging Vector Fields Using Line Integral Convolution. In *Poceedings of ACM SIGGRAPH 1993* (1993), Annual Conference Series, pp. 263–272.

[CSBI05] Camarri S., Salvetti M.-V., Buffoni M., Iollo A.: Simulation of the three-dimensional flow around a square cylinder between parallel walls at moderate Reynolds numbers. In *XVII Congresso di Meccanica Teorica ed Applicata* (2005).

[DH92] Darmofal D., Haimes R.: *Visualization of 3-D Vector Fields: Variations on a Stream*. Paper 92-0074, AIAA, 1992.

[dLvW95] de Leeuw W., van Wijk J. J.: Enhanced Spot Noise for Vector Field Visualization. In *Proceedings IEEE Visualization '95* (Oct. 1995), IEEE Computer Society, pp. 233–239.

[ELC*12] Edmunds M., Laramee R., Chen G., Zhang E., Max N.: Advanced, Automatic Stream Surface Seeding and Filtering. *Theory and Practice of Computer Graphics (TPCG 2012)* (2012), forthcoming.

[ELM*12] Edmunds M., Laramee R., Malki R., Masters I., Croft T., Chen G., Zhang E.: Automatic Stream Surface Seeding: A Feature Centered Approach. *Computer Graphics Forum (Eurographics 2012) 31*, 3 (2012), forthcoming.

[EML*11] Edmunds M., McLoughlin T., Laramee R. S., Chen G., Zhang E., Max N.: Automatic Stream Surfaces Seeding. In *EUROGRAPHICS 2011 Short Papers* (Llandudno, Wales, UK, April 11–15 2011), pp. 53–56.

[FBTW10] Ferstl F., Burger K., Theisel H., Westermann R.: Interactive Separating Streak Surfaces. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (November-December 2010), 1569–1577.

[FC95] Forssell L. K., Cohen S. D.: Using Line Integral Convolution for Flow Visualization: Curvilinear Grids, Variable-Speed Animation, and Unsteady Flows. *IEEE Transactions on Visualization and Computer Graphics 1*, 2 (June 1995), 133–141.

[FCL09] Forsberg A. S., Chen J., Laidlaw D. H.: Comparing 3D Vector Field Visualization Methods: A User Study. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1219–1226.

[For94] Forssell L. K.: Visualizing Flow over Curvilinear Grid Surfaces Using Line Integral Convolution. In *Proceedings IEEE Visualization '94* (Oct. 1994), IEEE Computer Society, pp. 240–247.

[Gel01] Gelder A. V.: Stream Surface Generation for Fluid Flow Solutions on Curvilinear Grids. In *Proceedings of the Joint Eurographics - IEEE TCVG Symposium on Visualizatation (VisSym-01)* (May 28–30 2001), Ebert D., Favre J. M., Peikert R., (Eds.), Springer-Verlag, pp. 95–106.

[GKT*08] Garth C., Krishnan H., Tricoche X., Tricoche T., Joy K. I.: Generation of Accurate Integral Surfaces in Time-Dependent Vector Fields. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1404–1411.

[GOV10] GOV: Met Office. http://www.metoffice.gov.uk/, 2010. Accessed: 08th August 2010.

[GTS*04] Garth C., Tricoche X., Salzbrunn T., Bobach T., Scheuermann G.: Surface Techniques for Vortex Visualization. In *Joint Eurographics - IEEE TCVG Symposium on Visualization* (2004), Deussen O., Hansen C., Keim D., Saupe D., (Eds.), Eurographics Association, pp. 155–164.

[Hal01] Haller G.: Distinguished Material Surfaces and Coherent Structures in Three Dimensional Fluid Flows. *Physica D 149* (2001), 248–277.

[HGH*10] Hummel M., Garth C., Hamann B., Hagen H., Joy K.: IRIS: Illustrative Rendering for Integral Surfaces. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (2010), 1319–1328.

[Hul92] Hultquist J. P. M.: Constructing Stream Surfaces in Steady 3D Vector Fields. In *Proceedings IEEE Visualization '92* (1992), pp. 171–178.

[IEE10] IEEE VisWeek: 2008 IEEE Visualization Design Contest. http://viscontest.sdsc.edu/2008/, 2010. Accessed: 08th August 2010.

[Int] International CFD Database, http://cfd.cineca.it/.

[JEH01] Jobard B., Erlebacher G., Hussaini M. Y.: Lagrangian-Eulerian Advection for Unsteady Flow Visualization. In *Proceedings IEEE Visualization '01* (October 2001), IEEE Computer Society, pp. 53–60.

[JL97] Jobard B., Lefer W.: Creating Evenly–Spaced Streamlines of Arbitrary Density. In *Proceedings of the Eurographics Workshop on Visualization in Scientific Computing '97* (1997), vol. 7, pp. 45–55.

[Ker90] Kerlic D. G.: Moving Iconic Objects in Scientific Visualization '91. In *Proceedings of Visualization* (October 1990), pp. 124–129.

[KGJ09] Krishnan H., Garth C., Joy K.: Time and Streak Surfaces for Flow Visualization in Large Time-Varying Data Sets. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1267–1274.

[Lar03] Laramee R. S.: FIRST: A Flexible and Interactive Resampling Tool for CFD Simulation Data. *Computers & Graphics 27*, 6 (2003), 905–916.

[LCS*12] Liu Z., Cai S., Swan II J. E., Moorhead II R. J., Martin J. P., Jankun-Kelly T. J.: A 2D Flow Visualization User Study Using Explicit Flow Synthesis and Implicit Task Design. *IEEE Transactions on Visualization and Computer Graphics 18*, 5 (2012), 783–796.

[LEG*08] Laramee R. S., Erlebacher G., Garth C., Theisel H., Tricoche X., Weinkauf T., Weiskopf D.: Applications of Texture-Based Flow Visualization. *Engineering Applications of Computational Fluid Mechanics (EACFM) 2*, 3 (Sept. 2008), 264–274.

[LGD*05] Laramee R. S., Garth C., Doleisch H., Schneider J., Hauser H., Hagen H.: Visual Analysis and Exploration of Fluid Flow in a Cooling Jacket. In *Proceedings IEEE Visualization 2005* (2005), pp. 623–630.

[LGSH06] Laramee R. S., Garth C., Schneider J., Hauser H.: Texture-Advection on Stream Surfaces: A Novel Hybrid Visualization Applied to CFD Results. In *Data Visualization, The Joint Eurographics-IEEE VGTC Symposium on Visualization (EuroVis 2006)* (2006), Eurographics Association, pp. 155–162,368.

[LHD*04] Laramee R. S., Hauser H., Doleisch H., Post F. H., Vrolijk B., Weiskopf D.: The State of the Art in Flow Visualization: Dense and Texture-Based Techniques. *Computer Graphics Forum 23*, 2 (June 2004), 203–221.

[LJH03] Laramee R., Jobard B., Hauser H.: Image Space Based Visualization of Unsteady Flow on Surfaces. In *Proceedings IEEE Visualization '03* (2003), IEEE Computer Society, pp. 131–138.

[LKJ*05] Laidlaw D., Kirby R., Jackson C., Davidson J., Miller T., da Silva M., Warren W., Tarr M.: Comparing 2D Vector Field Visualization Methods: A User Study. *IEEE Transactions on Visualization and Computer Graphics 11*, 1 (2005), 59–70.

[LMG97] Loffelmann H., Mroz L., Groller E.: *Hierarchical Streamarrows for the Visualization of Dynamical Systems*. Technical report, Institute of Computer Graphics, Vienna University of Technology, 1997.

[LMGP97] Loffelmann H., Mroz L., Groller E., Purgathofer W.: Stream Arrows: Enhancing the Use of Streamsurfaces for the Visualization of Dynamical Systems. *The Visual Computer 13* (1997), 359–369.

[LSH04] Laramee R. S., Schneider J., Hauser H.: Texture-Based Flow Visualization on Isosurfaces from Computational Fluid Dynamics. In *Data Visualization, The Joint Eurographics-IEEE TVCG Symposium on Visualization (VisSym '04)* (2004), Eurographics Association, pp. 85–90,342.

[LTWH08] Li G.-S., Tricoche X., Weiskopf D., Hansen C. D.: Flow Charts: Visualization of Vector Fields on Arbitrary Surfaces. *IEEE Transactions on Visualization and Computer Graphics 14*, 5 (2008), 1067–1080.

[LvWJH04] Laramee R., van Wijk J. J., Jobard B., Hauser H.: ISA and IBFVS: Image Space Based Visualization of Flow on Surfaces. *IEEE Transactions on Visualization and Computer Graphics 10*, 6 (Nov. 2004), 637–648.

[LWSH04] Laramee R., Weiskopf D., Schneider J., Hauser H.: Investigating Swirl and Tumble Flow with a Comparison of Visualization Techniques. In *Proceedings IEEE Visualization 2004* (2004), pp. 51–58.

[Mas99] Mase G. T.: *Continuum Mechanics for Engineers*. CRC Press, 1999.

[MKFI97] Mao X., Kikukawa M., Fujita N., Imamiya A.: Line Integral Convolution for 3D Surfaces. In *Visualization in Scientific Computing '97. Proceedings of the Eurographics Workshop* (1997), Eurographics, pp. 57–70.

[MLZ09] McLoughlin T., Laramee R. S., Zhang E.: Easy Integral Surfaces: A Fast, Quad-Based Stream and Path Surface Algorithm. In *Proceedings of Computer Graphics International (CGI '09)* (May 2009), Computer Graphics Society, Springer, pp. 67–76.

[MLZ10] McLoughlin T., Laramee R. S., Zhang E.: Constructing Streak Surfaces in 3D Unsteady Vector Fields. In *Proceedings Spring Conference on Computer Graphics* (Dec 2010), Hauser H., (Ed.), pp. 25–32.

[MS93] Ma K.-L., Smith P. J.: Cloud Tracing in Convection-Diffusion Systems. In *VIS '93: Proceedings of the 4th conference on Visualization '93* (Washington, DC, USA, 1993), IEEE Computer Society, pp. 253–260.

[PCY09] Palmerius K. L., Cooper M., Ynnerman A.: Flow Field Visualization Using Vector Field Perpendicular Surfaces. In *Spring Conference on Computer Graphics* (2009).

[Pea85] Peachey D. R.: Solid Texturing of Complex Surfaces. *Computer Graphics (Proceedings of ACM SIGGRAPH 85) 19*, 3 (1985), 279–286.

[PGL*12] Peng Z., Grundy E., Laramee R. S., Chen G., Croft N.: Mesh-Driven Vector Field Clustering and Visualization: An Image-Based Approach. *IEEE Transactions on Visualization and Computer Graphics* (2012), Forthcoming.

[PL08] Peng Z., Laramee R. S.: Vector Glyphs for Surfaces: A Fast and Simple Glyph Placement Algorithm for Adaptive Resolution Meshes. In *Proceedings of Vision, Modeling, and Visualization (VMV) 2008* (2008), pp. 61–70.

[PPF*10] Pobitzer A., Peikert R., Fuchs R., Schindler B., Kuhn A., Theisel H., Matkovic K., Hauser H.: On The Way Towards Topology-Based Visualization of Unsteady Flow the State Of The Art. *EuroGraphics 2010 State of the Art Reports* (2010), 137–154.

[PR99] Peikert R., Roth M.: The Parallel Vectors Operator - A Vector Field Visualization Primitive. In *Proceedings of IEEE Visualization '99* (1999), IEEE Computer Society, pp. 263–270.

[PS09] Peikert R., Sadlo F.: Topologically Relevant Stream Surfaces for Flow Visualization. In *Proc. Spring Conference on Computer Graphics* (April 2009), Hauser H., (Ed.), pp. 43–50.

[PVH*03] Post F. H., Vrolijk B., Hauser H., Laramee R. S., Doleisch H.:

The State of the Art in Flow Visualization: Feature Extraction and Tracking. *Computer Graphics Forum 22*, 4 (Dec. 2003), 775–792.

[PW94] Pagendarm H.-G., Walter B.: Feature Detection from Vector Quantities in a Numerically Simulated Hypersonic Flow Field in combination with Experimental Flow Visualization. In *VIS '94: Proceedings of the conference on Visualization '94* (1994), IEEE Computer Society Press, pp. 117–123.

[PZ10] Palacios J., Zhang E.: Interactive Visualization of Rotational Symmetry Fields on Surfaces. *IEEE Transactions on Visualization and Computer Graphics* (2010).

[RP96] Roth M., Peikert R.: Flow Visualization for Turbomachinery Design. In *Proceedings IEEE Visualization '96* (Oct. 1996), pp. 381–384.

[RPH*09] Rosanwo O., Petz C., Hotz I., Prohaska S., Hege H.-C.: Dual Streamline Seeding. In *Proceedings of IEEE Pacific Visualization Symposium '09* (2009).

[SBH*01] Scheuermann G., Bobach T., Hagen H., Mahrous K., Hamann B., Joy K. I., Kollmann W.: A Tetrahedral-Based Stream Surface Algorithm. In *Proceedings IEEE Visualization '01* (Oct. 2001), pp. 151–157.

[SJM96] Shen H.-W., Johnson C. R., Ma K.-L.: Visualizing Vector Fields Using Line Integral Convolution and Dye Advection. In *1996 Volume Visualization Symposium* (Oct. 1996), IEEE, pp. 63–70.

[SK98] Shen H.-W., Kao D.: A New Line Integral Convolution Algorithm for Visualizing Time-Varying Flow Fields. *IEEE Transactions on Visualization and Computer Graphics 4*, 2 (Apr. – June 1998), 98–108.

[SLCZ09] Spencer B., Laramee R. S., Chen G., Zhang E.: Evenly-Spaced Streamlines for Surfaces: An Image-Based Approach. *Computer Graphics Forum 28*, 6 (2009), 1618–1631.

[SP07] Sadlo F., Peikert R.: Efficient Visualization of Lagrangian Coherent Structures by Filtered AMR Ridge Extraction. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1456–1463.

[SRWS10] Schneider D., Reich W., Wiebel A., Scheuermann G.: Topology Aware Stream Surfaces. In *Eurographics/IEEE Symposium on Visualization* (Bordeaux, France, June 9–11 2010), vol. 29, pp. 1153–1161.

[STWE07] Schafhitzel T., Tejada E., Weiskopf D., Ertl T.: Point-Based Stream Surfaces and Path Surfaces. In *GI '07: Proceedings of Graphics Interface 2007* (2007), ACM, pp. 289–296.

[SVL91] Schroeder W., Volpe C. R., Lorensen W. E.: The Stream Polygon: A Technique for 3D Vector Field Visualization. In *Proceedings IEEE Visualization '91* (1991), pp. 126–132.

[SWS09] Schneider D., Wiebel A., Scheuermann G.: Smooth Stream Surfaces of Fourth Order Precision. *Computer Graphics Forum 28*, 3 (2009).

[Tan10] Tannoy: Tannoy. http://www.tannoy.com/, 2010. Accessed: 29th September 2010.

[TS03] Theisel H., Seidel H.-P.: Feature Flow Fields. In *Proceedings of the Joint Eurographics - IEEE TCVG Symposium on Visualization (VisSym 03)* (2003), pp. 141–148.

[TSW*05] Theisel H., Shaner J., Weinkauf T., Hege H.-C., Seidel H.-P.: Extraction of Parallel Vector Surfaces in 3D Time-Dependent Fields and Application to Vortex Core Line Tracking. In *Proceedings IEEE Visualization 2005* (2005), pp. 631–638.

[TWHS03] Theisel H., Weinkauf T., Hege H.-C., Seidel H.-P.: Saddle Connectors–An Approach to Visualizing the Topological Skeleton of Complex 3D Vector Fields. In *Proceedings IEEE Visualization '03* (2003), pp. 225–232.

[USM96] Ueng S. K., Sikorski C., Ma K. L.: Efficient Streamline, Streamribbon, and Streamtube Constructions on Unstructured Grids. *IEEE Transactions on Visualization and Computer Graphics 2*, 2 (June 1996), 100–110.

[vFWTS08a] von Funck W., Weinkauf T., Theisel H., Seidel H.-P.: Smoke Surfaces: An Interactive Flow Visualization Technique Inspired by Real-World Flow Experiments. *IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE Visualization) 14*, 6 (2008), 1396–1403.

[vFWTS08b] von Funck W., Weinkauf T., Theisel H., Seidel H.-P.: Smoke surfaces: An interactive flow visualization technique inspired by real-world flow experiments. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization 2008) 14*, 6 (November - December 2008), 1396–1403.

[vW91] van Wijk J. J.: Spot noise-Texture Synthesis for Data Visualization. In *Computer Graphics (Proceedings of ACM SIGGRAPH 91)* (1991), Sederberg T. W., (Ed.), vol. 25, pp. 309–318.

[vW93] van Wijk J. J.: Implicit Stream Surfaces. In *Proceedings of the Visualization '93 Conference* (Oct. 1993), IEEE Computer Society, pp. 245–252.

[vW02] van Wijk J. J.: Image Based Flow Visualization. *ACM Transactions on Graphics 21*, 3 (2002), 745–754.

[vW03] van Wijk J. J.: Image Based Flow Visualization for Curved Surfaces. In *Proceedings IEEE Visualization '03* (2003), IEEE Computer Society, pp. 123–130.

[WE04] Weiskopf D., Ertl T.: A Hybrid Physical/Device-Space Approach for Spatio-Temporally Coherent Interactive Texture Advection on Curved Surfaces. In *Proceedings of Graphics Interface* (2004), pp. 263–270.

[Wei09] Weiskopf D.: *Iterative twofold line integral convolution for texture-based vector field visualization* . Springer, 2009, pp. 191–211.

[WH06] Weiskopf D., Hauser H.: Cycle shading for the assessment and visualization of shape in one and two codimensions. In *Proceedings of Graphics Interface 2006* (Toronto, Ont., Canada, Canada, 2006), GI '06, Canadian Information Processing Society, pp. 219–226.

[WJE00] Westermann R., Johnson C., Ertl T.: A Level-Set Method for Flow Visualization. In *VIS '00: Proceedings of the conference on Visualization '00* (2000), IEEE Computer Society Press, pp. 147–154.

[WTHS04] Weinkauf T., Theisel H., Hege H. C., Seidel H.-P.: Boundary Switch Connectors for Topological Visualization of Complex 3D Vector Fields. In *Proceedings of the Joint Eurographics - IEEE TCVG Symposium on Visualization (VisSym 04)* (2004), pp. 183–192.

[YMM10] Yan S., Max N., Ma K. L.: Polygonal Surface Advection applied to Strange Attractors. In *Pacific Graphics 2010* (2010), vol. 29.

[ZPKG02] Zwicker M., Pauly M., Knoll O., Gross M.: Pointshop 3D: An Interactive System for Point-Based Surface Editing. *ACM Transactions on Graphics 21*, 3 (July 2002), 322–329.