12 Overview of Flow Visualization

DANIEL WEISKOPF
University of Stuttgart
GORDON ERLEBACHER
Florida State University

12.1 Introduction

Flow visualization is an important topic in scientific visualization and has been the subject of active research for many years. Typically, data originates from numerical simulations, such as those of computational fluid dynamics, and needs to be analyzed by means of visualization to gain an understanding of the flow. With the rapid increase of computational power for simulations, the demand for more advanced visualization methods has grown. This chapter presents an overview of important and widely used approaches to flow visualization, along with references to more detailed descriptions in the original scientific publications. Although the list of references covers a large body of research, it is by no means meant to be a comprehensive collection of articles in the field.

12.2 Mathematical Description of a Vector Field

We start with a rather abstract definition of a vector field by making use of concepts from differential geometry and the theory of differential equations. For more detailed background information on these topics, we refer to textbooks on differential topology and geometry [45,46,66,81]. Although this mathematical approach might seem quite abstract for many applications, it has the advantage of being a flexible and generic description that is applicable to a wide range of problems.

We first give the definition of a *vector field*. Let M be a smooth m-manifold with boundary, let N be an n-dimensional submanifold with boundary ($N \subset M$), and let $I \subset \mathbb{R}$ be an open interval of real numbers. A map

$$u : N \times I \to TM$$

is a *time-dependent vector field* provided that

$$u(x, t) \in T_x M$$

An element $t \in I$ serves as a description for time; $x \in N$ is a position in space. TM is a tangent bundle—the collection of all tangent vectors, along with the information of the point of tangency. Finally, $T_x M$ is the tangent space associated with x. The vector field maps a position in space and time, (x, t), to a tangent vector located at the same reference point x. For a *tangential time-dependent vector field*, the mapping remains in the tangent bundle TN and therefore does not contain a normal component. That is,

$$u : N \times I \to TN$$

For a nontangential vector field, a related tangential vector field can be computed by projection from $T_x M$ to $T_x N$, i.e., by removing the normal parts from the vectors.

Integral curves are directly related to vector fields. Let $u : N \times I \to TN$ be a continuous (tangential) vector field. Let x_0 be a point in N and $J \subset I$ be an open interval that contains t_0. The C^1 map

$$\xi_{x_0, t_0} : J \to N$$

with

$$\xi_{x_0, t_0}(t_0) = x_0 \quad \text{and} \quad \frac{d\xi_{x_0, t_0}(t)}{dt} = u(\xi_{x_0, t_0}(t), t)$$

is an integral curve for the vector field with initial condition $x = x_0$ at $t = t_0$. The subscripts in the notation of ξ_{x_0, t_0} denote this initial condition. These integral curves are usually referred to as *pathlines*, especially in the context of flow visualization. If u satisfies the Lipschitz condition, the differential equation for ξ_{x_0, t_0} has a unique solution.

In all practical applications of flow visualization, the data is given on a manifold of two or three dimensions. To investigate a vector field on an arbitrary curved surface, the above formalism is necessary and, for example, the issue of describing the surface by charts has to be addressed. Very often, however, N is just a Euclidean space. This allows us to use a simpler form of the tangential vector field given by

$$u : \Omega \times I \longrightarrow \mathbb{R}^n, \qquad (\mathbf{x}, t) \mapsto \mathbf{u}\,(\mathbf{x}, t)$$

The vector field is defined on the n-dimensional Euclidean space $\Omega \subset \mathbb{R}^n$ and depends on time $t \in I$. We use boldface lowercase letters to denote vectors in n dimensions. The reference point x is no longer explicitly attached to the tangent vector. In this modified notation, the integral curve is determined by the ordinary differential equation

$$\frac{d\mathbf{x}_{\text{path}}\,(t;\mathbf{x}_0, t_0)}{dt} = \mathbf{u}\,(\mathbf{x}_{\text{path}}(t;\mathbf{x}_0, t_0), t) \tag{12.1}$$

We assume that the initial condition $\mathbf{x}_{\text{path}}\,(t_0;\mathbf{x}_0, t_0) = \mathbf{x}_0$ is given at time t_0, i.e., all integral curves are labeled by their initial conditions $(\mathbf{x}_0, t_0)$ and parameterized by t. By construction, the tangent to the pathline at position x and time t is precisely the velocity $\mathbf{u}(\mathbf{x}, t)$. In more general terms, the notation $\mathbf{x}(t;\mathbf{x}_0, t_0)$ is used to describe any curve parameterized by t that contains the point $\mathbf{x}_0$ for $t = t_0$.

12.3 Particle Tracing in Time-Dependent Flow Fields

Integral curves play an important role in visualizing the associated vector field and in understanding the underlying physics of the flow. There exist two important additional types of characteristic curves: *streamlines* and *streaklines*. In steady flows, pathlines, streamlines, and streaklines are identical. When the vector field depends explicitly on time, these curves are distinct from one another.

In a steady flow, a particle follows the streamline, which is a solution to

$$\frac{d\mathbf{x}_{\text{stream}}\,(t;\mathbf{x}_0, t_0)}{dt} = \mathbf{u}\,(\mathbf{x}_{\text{stream}}\,(t;\mathbf{x}_0, t_0))$$

In an unsteady context, we consider the instantaneous vector field at fixed time τ. The particle paths associated with this artificially steady, virtually frozen field are the streamlines governed by

$$\frac{d\mathbf{x}_{\text{stream}}\,(t;\mathbf{x}_0, t_0)}{dt} = \mathbf{u}\,(\mathbf{x}_{\text{stream}}\,(t;\mathbf{x}_0, t_0), \tau)$$

Here, t and t_0 are just parameters along the curve and do not have the meaning of physical time, in contrast to the physical time τ. A third type of curve is produced by dye released into the flow. If dye is released continuously into a flow from a fixed point $\mathbf{x}_0$, it traces out a streakline. For example, smoke emanating from a lit cigarette follows a streakline.

It is instructive to derive integrated equations for pathlines, streamlines, and streaklines. The solution to the ordinary differential equation (Equation 12.1) is obtained by formal integration, which gives the pathline

$$\mathbf{x}_{\text{path}}\,(t\,;\mathbf{x}_0, t_0) = \mathbf{x}_0 + \int_{t_0}^{t} \mathbf{u}\,(\mathbf{x}_{\text{path}}\,(s\,;\mathbf{x}_0, t_0), s)\,ds$$

Similarly, streamlines at time τ can be computed by

$$\mathbf{x}_{\text{stream}}\,(t\,;\mathbf{x}_0, t_0) = \mathbf{x}_0 + \int_{t_0}^{t} \mathbf{u}\,(\mathbf{x}_{\text{stream}}(s\,;\mathbf{x}_0, t_0), \tau)\,ds$$

To obtain the snapshot of a streakline at time t, a set of particles is released from $\mathbf{x}_0$ at times $s \in [t_{\min}, t]$ and the particles' positions are evaluated at time t:

$$\mathbf{x}_{\text{streak}}(s\,;\mathbf{x}_0, t) = \mathbf{x}_{\text{path}}(t\,;\mathbf{x}_0, s)$$

The streakline is parameterized by s, and $t_{\min}$ is the first time that particles are released.

12.4 Classification of Visualization Approaches

There exist many different vector-field visualization techniques, which can be distinguished according to their properties with respect to a number of categories. The following classification should be considered rather a collection of important issues than a complete taxonomy. These issues should be taken into account when choosing a visualization approach.

In one classification scheme, techniques are distinguished by the relationship between a vector field and its associated visual representation. Point-based direct visualization approaches take into account the vector field at a point, and possibly also its neighborhood, to obtain a visual representation. The vector field is directly mapped to graphical primitives in the sense that no sophisticated intermediate processing of data is performed. Another class is based on characteristic curves obtained by particle tracing. The third class thoroughly preprocesses data to identify important features, which then serve as a basis for the actual visualization.

Another type of property is the density of representation: the domain can be sparsely or densely covered by visualization objects. Density is particularly useful for subclassing particle-tracing approaches. Related to density is the distinction between local and global methods. A global technique essentially shows the complete flow, whereas important features of the flow could possibly be missed by a local technique.

The choice of a visualization method is also influenced by the structure of the data. The dimensionality of the manifold on which the vector field is defined plays an important role. For example, strategies that work well in 2D might be much less useful in 3D because of perception issues. The recognition of orientation and spatial position of graphical primitives is more difficult, and important primitives could be hidden by others. Dimensionality also affects performance; a 3D technique has to process substantially more data. If visualization is restricted to slices or more general hypersurfaces of a 3D flow, the projection of vectors onto the tangent spaces of the hypersurfaces has to be considered. Moreover, a distinction has to be made between time-dependent and time-independent data. A steady flow is usually much less demanding since frame-to-frame coherence is easy to achieve and streamlines, streaklines, and pathlines are identical. Finally, the type of grid has to be taken into account. Data can be provided, for example, on uniform, rectilinear, curvilinear, or unstructured grids. The grid type affects the visualization algorithms with respect to mainly data storage and access mechanisms or interpolation schemes. Ideally, the final visual representation does not depend on the underlying grid.

It should be noted that there is no "ideal" technique that is best for all visualization tasks. Therefore, it is often useful to combine different approaches for an effective overall visualization. Nevertheless, we focus on describing the methods individually. The techniques are roughly ordered according to their classification: point-based direct methods, sparse representations for particle-tracing techniques, dense representations based on particle tracing, and feature-based approaches. The other types of properties are discussed along with the descriptions of the individual methods.

This chapter is meant to provide an overview of flow visualization. For more detailed descriptions, we refer to [18,40,82,88,99] and to the other flow visualization chapters of this book.

12.5 Point-Based Direct Flow Visualization

The traditional technique of arrow plots is a well-known example for direct flow visualization based on glyphs. Small arrows are drawn at discrete grid-points, showing the direction of the flow and serving as local probes for the velocity field (Fig. 12.1a). In the closely related hedgehog approach, the flow is visualized by

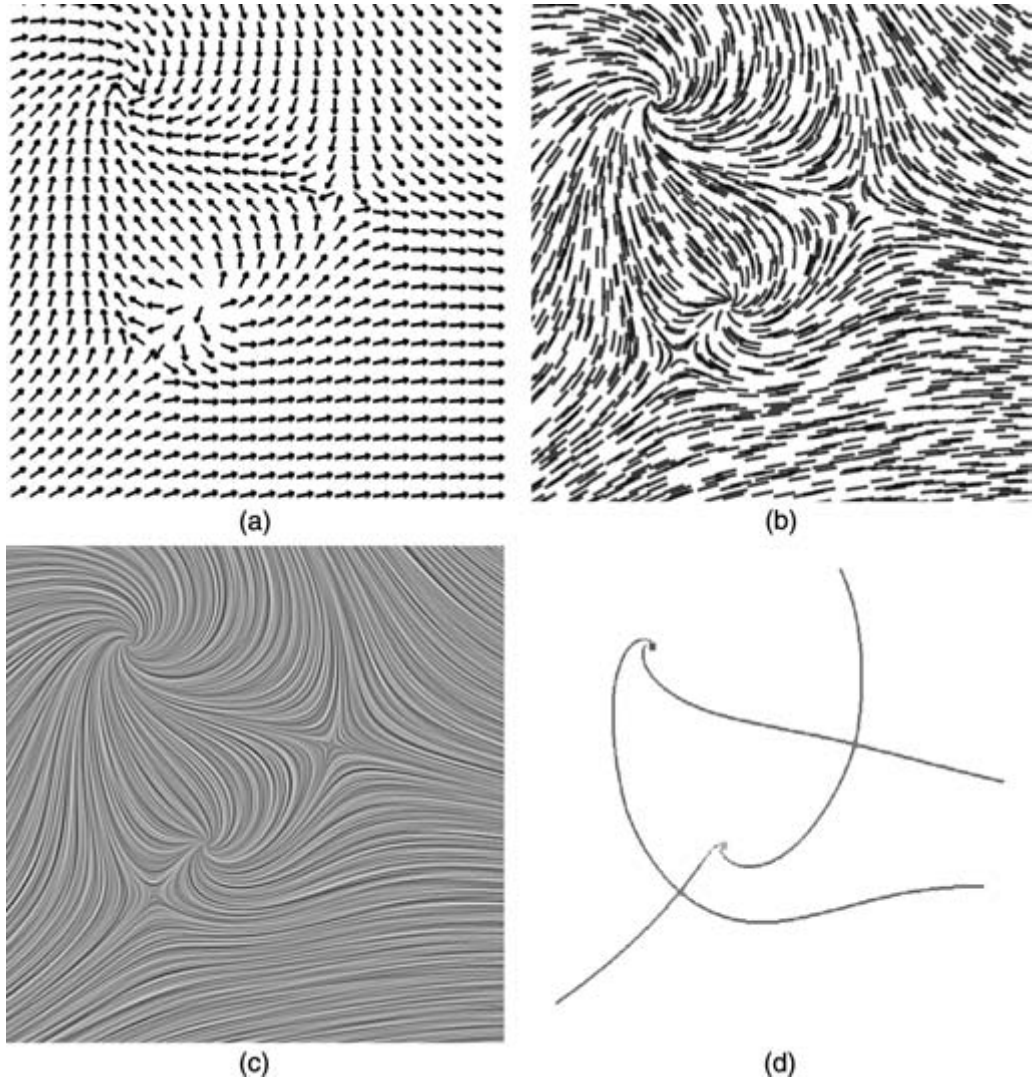

Figure 12.1 Comparison of visualization techniques applied to the same 2D flow: (a) arrow plot, (b) streamlets, (c) line integral convolution (LIC), and (d) topology-based. Image 12.1d courtesy of Gerik Scheuermann.

directed line segments whose lengths represent the magnitude of the velocity. To avoid possible distracting patterns for a uniform sampling by arrows or hedgehogs, randomness can be introduced in their positions [29]. Arrow plots can be directly applied to time-dependent vector fields by letting the arrows adapt to the velocity field for the current time. For 3D representations, the following facts have to be considered: the position and orientation of an arrow is more difficult to understand due to the projection onto the 2D image plane, and an arrow might occlude other arrows in the background. The problem of clutter can be addressed by highlighting arrows with orientations in a range specified by the user [9], or by selectively seeding

the arrows. Illumination and shadows serve to improve spatial perception; for example, shadowing can be applied to hedgehog visualizations on 2D slices of a 3D flow [65].

More complex glyphs [26] can be used to provide additional information on the flow at a point of the flow (Fig. 12.2). In addition to the actual velocity, information on the Jacobian of the velocity field is revealed. The Jacobian is presented in an intuitive way by decomposing the Jacobian matrix into meaningful components and mapping them to icons based on easily understandable metaphors. Typical data encoded into glyphs comprise velocity, acceleration, curvature, local rotation, shear, and convergence. Glyphs can also be used to represent information on the uncertainty of the vector-field data [134]. Glyph-based uncertainty visualization is also covered by Lodha et al. [69] and Pang et al. [86], who additionally discuss uncertainty representations for other visualization styles.

Another strategy is to map flow properties to a single value and apply techniques known from the visualization of scalar data. Typically, the magnitude of the velocity or one of the velocity components is used. For 2D flow visualization, a mapping to color or to iso-lines (contour lines) is often applied. Volume-visualization techniques have to be employed in the case of 3D data. Direct volume rendering, which avoids occlusion problems by selective use of semitransparency, can be applied to single-component data derived from vector fields [30,120]; recent developments are specifically designed for time-dependent data [16,36,85].

12.6 Sparse Representations for Particle-Tracing Techniques

Another class of visualization methods is based on the characteristic lines obtained by particle tracing. Among these are the aforementioned pathlines, streamlines, and streaklines. In addition, *time lines*, constructed from particles released at the same time from different points along a curve, can be used. All these lines are quite intuitive because they represent some kind of transport along the flow. In this section, we

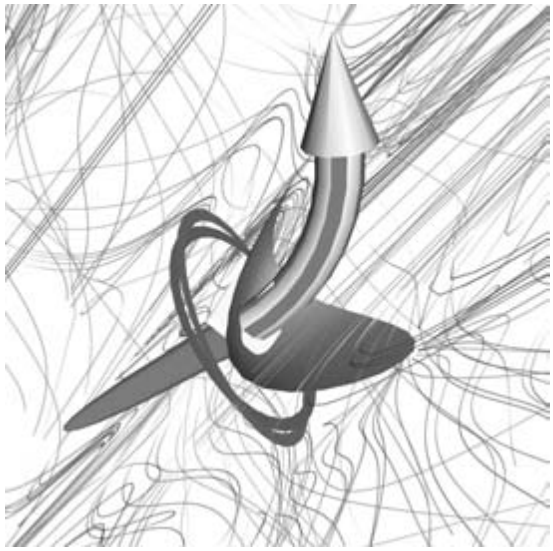

Figure 12.2 Glyph-based 3D flow visualization, combined with illuminated streamlines. (See also color insert.)

discuss sparse representations, in which the spatial domain is not densely covered.

A traditional particle-based approach computes characteristic curves and draws them as thin lines. Since many researchers handle time-independent vector fields, the notion of streamlines is used frequently. The visualization concepts can often be generalized to pathlines, streaklines, or time lines, even if not explicitly mentioned. Streamlines just serve as a role model for the other characteristic lines. Particles traced for a very short time generate short streamlines or *streamlets*.

Streamlines and streamlets can be used in 2D space, on 2D hypersurfaces of an underlying 3D flow, and for 3D flows. Hypersurfaces typically are sectional slices through the volume, or they are curved surfaces such as boundaries or other characteristic surfaces. It is important to note that the use of particle traces for vector fields projected onto slices may be misleading, even within a steady flow: a streamline on a slice may depict a closed loop even though no particle would ever traverse the loop. The problem is caused by the fact that flow components orthogonal to the slice are neglected during flow integration. For 3D flows, perceptual problems might arise due to distortions resulting from the projection onto the image plane. Moreover, issues of occlusion and clutter have to be considered. An appropriate solution is to find selective seed positions for particle traces that still show the important features of the flow, but do not overcrowd the volume; for example, a thread of streamlets along characteristic structures of 3D flow [71] can be used. The method of illuminated streamlines [136], based on illumination in diverse codimensions [1], improves the perception of those lines, and it also increases depth information and addresses the problem of occlusion by making the streamlines partially transparent. An example is shown in Fig. 12.2.

In 2D, particle traces are usually represented by thin lines, although the width of a line is sometimes modified to represent further information. Fig. 12.1b shows an example with a collection of streamlets. In 3D applications, however, the additional spatial dimension allows more information to be encoded into the graphical representation through the use of geometric objects of finite extent perpendicular to the particle trace. Examples of such an extension of streamlines in 3D are *streamribbons* and *streamtubes*. A streamribbon is the area swept out by a deformable line segment along a streamline. The strip-like shape of a streamribbon displays the rotational behavior of a 3D flow. Fig. 12.3 shows a visualization of a 3D fluid simulation combining streamribbons, streamlines, arrows, and color coding [104]. An iconic streamtube [119] is a thick tube-shaped streamline whose radial extent shows the expansion of the flow. As a further extension of streamtubes, *dash tubes* [33] provide animated, opacity-mapped tubes. *Stream polygons* [103] trace arbitrary polygonal cross-sections along a streamline and thus are closely related to streamtubes and streamribbons. The properties of the polygons, such as the size, shape, and orientation, reflect properties of the vector field, including strain, displacement, and rotation. *Streamballs* [10] use their radii to visualize divergence and acceleration in a flow. Other geometric objects such as tetrahedra [110] may be used instead of spheres. Another extension of streamlines is provided by *stream surfaces*, which are everywhere tangent to the vector field. A stream surface can be modeled by an implicit surface [123] or approximated by explicitly connecting a set of streamlines along time lines. Stream surfaces present challenges related to occlusion, visual complexity, and interpretation, which can be addressed by choosing an appropriate placement and orientation based on *principal stream surfaces* [15] or through user interaction [47]. Ray casting can be used to render several stream surfaces at different depths [32]. *Stream arrows* [72] cut out arrow-shaped portions from a stream surface and thus provide additional information on the flow, such as flow direction and convergence or divergence. Stream surfaces can also be computed

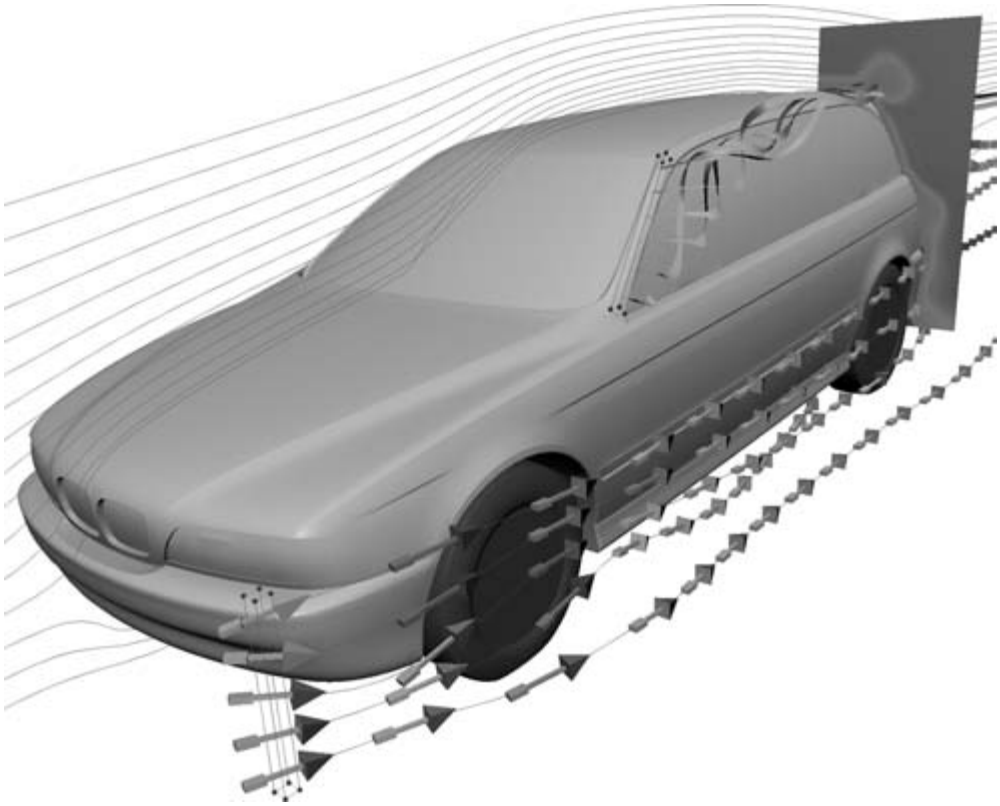

Figure 12.3 A combination of streamlines, streamribbons, arrows, and color coding for a 3D flow. Image courtesy of BMW Group and Martin Schulz. (See also color insert.)

and visualized based on surface particles [122], which are subject to less occlusion than is a full-bodied surface.

The generalization of the concept of particles released from single discrete points (for streamlines or streaklines) or from several points on a 1D line (for stream surfaces) leads to *flow volumes* [78]. A flow volume is a region of a 3D flow domain traced out by a 2D patch over time. The resulting volume can be visualized by volume rendering techniques. Since any flow volume can be (at least approximately) represented by a collection of tetrahedral cells, volume rendering techniques for unstructured grids can be applied, such as hardware-accelerated cell projection [93,107]. Flow volumes can be extended to unsteady flows [6], yielding the analog of streaklines. Finally, *time surfaces* extend time lines to surfaces that are built from particles released from a 2D patch. The evolution of time surfaces can be handled by a level-set approach [131].

A fundamental issue of all particle-based techniques is an appropriate choice of initial conditions—seed-point positioning—in order to catch all relevant features of the flow. Two main strategies can be identified: interactive or automatic placement of seed points. The interactive approach leaves the problem to the user and, in this sense, simplifies the problem from an algorithmic point of view. Nevertheless, the visualization system should be designed to

help the user identify appropriate seed points. For example, the virtual wind tunnel [11] is an early virtual-reality implementation of a flow-visualization system where particles can be interactively released by the user.

A useful approach for the automatic placement of seed points is to construct a uniform distribution of streamlines, which can be achieved for 2D vector fields [55,118] or for surfaces within curvilinear grids of a 3D flow [74]. The idea behind a uniform distribution of streamlines is that such a distribution very likely will not miss important features of the flow. Therefore, this approach can be regarded as a step towards a completely dense representation, which is discussed in the following section. Equally spaced streamlines can be extended to multiresolution hierarchies that support an interactive change of streamline density, while zooming in and out of the vector field [58]. Moreover, with this technique, the density of streamlines can be determined by properties of the flow, such as the magnitude of the velocity or the vorticity. Evenly spaced streamlines for an unsteady flow can be realized by correlating instantaneous streamline visualizations for subsequent time-steps [57]. Seeding strategies may also be based on vector-field topology; for example, flow structures in the vicinity of critical points can be visualized by appropriately setting the initial conditions for particle tracing [126].

Since all particle-tracing techniques are based on solving the differential equation for particle transport, issues of numerical accuracy and speed must be addressed. Different numerical techniques known from the literature can be applied for the initial value problem of ordinary differential equations. In many applications, explicit integration schemes such as nonadaptive or adaptive Runge-Kutta methods are used. The required accuracy for particle tracing depends on the visualization technique; for example, first-order Euler integration might be acceptable for streamlets but not for longer streamlines. A comparison of different integration schemes [111] helps to judge the tradeoff between computation time and accuracy. Besides the actual integration scheme, the grid on which the vector field is given is very important for choosing a particle-tracing technique. Point location and interpolation depend heavily on the grid and therefore affect the speed and accuracy of particle tracing. Both aspects are detailed by Nielson et al. [82], along with a comparison between C-space (computational-space) and P-space (physical-space) approaches. The numerics of particle tracing is discussed, for example, for the tetrahedral decomposition of curvilinear grids [62], especially for the decomposition of distorted cells [94], for unstructured grids [119], for analytical solutions in piecewise linearly interpolated tetrahedral grids [83], for stiff differential equations originating from shear flows [111], and for sparse grids [113].

12.7 Dense Representations for Particle-Tracing Methods

Another class of visualization approaches is based on a representation of the flow by a dense coverage through structures determined by particle tracing. Typically, dense representations are built upon texture-based techniques, which provide images of high spatial resolution. A detailed description of texture-based flow visualization and, in particular, its support by graphics hardware is discussed in Chapter 13. A summary of research in the field of dense representations can be found in the surveys [40, 99].

The distinction between dense and sparse techniques should not be taken too rigidly because both classes of techniques are closely related by the fact that they form visual structures based on particle tracing. Therefore, dense representations also lead to the same intuitive understanding of the flow. Often, a transition between both classes is possible [125]; for example, texture-based techniques with only few distinct visual elements might resemble a collection of few streamlines and, on the other

hand, evenly spaced streamline seeding can be used with a high line density.

An early texture-synthesis technique for vector-field visualization is *spot noise* [121], which produces a texture by generating a set of spots on the spatial domain. Each spot represents a particle moving over a short period of time and results in a streak in the direction of the flow at the position of the spot. Enhanced spot noise [27] improves the visualization of highly curved vector fields by adapting the shape of the spots to the local velocity field. Spot noise can also be applied on boundaries and surfaces [20,114]. A divide-and-conquer strategy makes possible an implementation of spot noise for interactive environments [19]. As an example application, spot noise was applied to the visualization of turbulent flow [21].

Line integral convolution (LIC) [14] is a widely used technique for the dense representation of streamlines in steady vector fields. An example is shown in Fig. 12.1c. LIC takes as input a vector field and a white-noise texture. The noise texture is locally smoothed along streamlines by convolution with a filter kernel. This filtering leads to a high correlation between the grey-scale values of neighboring pixels along streamlines and to little or no correlation perpendicular to streamlines. The contrast and quality of LIC images can be improved by postprocessing techniques, such as histogram equalization, high-pass filtering, or a second pass of LIC [84]. Both spot noise and LIC are based on dense texture representations and particle tracing and are, from a more abstract point of view, tightly related to each other [22]. The original LIC technique does not show the orientation and the magnitude of the velocity field, an issue that is addressed by variants of LIC. Periodic motion filters can be used to animate the flow visualization, and a kernel phase-shift can be applied to reveal the direction and magnitude of the vector field [31]. *Oriented Line Integral Convolution* (OLIC) [128] exploits the existence of distinguishable, separated blobs in a rather sparse texture and smears these blobs into the direction of the local velocity field by convolution with an asymmetric filter kernel to show the orientation of the flow. By sacrificing some accuracy, a fast version of OLIC (FROLIC) [127] is feasible. In another approach, orientation is visualized by combining animation and adding dye advection [105]. Multifrequency noise for LIC [64] visualizes the magnitude of the velocity by adapting the spatial frequency of noise.

Other visualization techniques achieve LIC-like images by applying methods not directly based on LIC. For example, fur-like textures [63] can be utilized by specifying the orientation, length, density, and color of fur filaments according to the vector field. The *integrate and draw* [90] approach deposits random grey-scale values along streamlines. *Pseudo LIC* (PLIC) [125] is a compromise between LIC and sparse particle-based representations and therefore allows for a gradual change between dense and sparse visualizations. PLIC uses LIC to generate a template texture in a preprocessing step. For the actual visualization, the template is mapped onto thin or thick streamlines, thus filling the domain with LIC-like structures. The idea of LIC textures applied to thick streamlines can be extended to an error-controlled hierarchical method for a hardware-accelerated level-of-detail approach [8].

LIC can be extended to nonuniform grids and curved surfaces, for example, to curvilinear grids [31] and to 2D unstructured or triangular meshes [4,76,112]. Multigranularity noise as the input for LIC [75] compensates for the nonisometric mapping from texture space to the cells of a curvilinear grid that differ in size. The projection of the normal component of the vector field needs to be taken into account for LIC-type visualizations on hypersurfaces [100].

Unsteady Flow LIC (UFLIC) [106] and its accelerated version [68] incorporate time into the convolution to visualize unsteady flow. The issue of temporal coherence is addressed by successively updating the convolution results over time. Forssell and Cohen [31] present a visualization of time-dependent flows on curvilinear surfaces. *Dynamic LIC* (DLIC) [109] is another extension of LIC, one that allows for

time-dependent vector fields, such as electric fields. A LIC-like image of an unsteady flow can also be generated by an adaptive visualization method using streaklines, where the seeding of streaklines is controlled by the vorticity [98].

Since LIC has to perform a convolution for each element of a high-resolution texture, computational costs are an issue. One solution to this problem utilizes the coherence along streamlines to speed up the visualization process [42,108]. Parallel implementations are another way of dealing with high computational costs [13,137]. Finally, implementations based on graphics hardware can enhance the performance of LIC [43].

From a conceptional point of view, an extension of LIC to 3D is straightforward. The convolution along streamlines is performed within a volume; the resulting grey-scale volume can be represented by volume-visualization techniques, such as texture-based volume rendering. However, computational costs are even higher than in 2D, and therefore, interactive implementations of the filtering process are hard to achieve. Even more importantly, possibly severe occlusion issues have to be considered: in a dense representation, there is a good chance that important features will get hidden behind other particle lines. A combination of interactive clipping and user intervention is one possible solution [89]. Alternatively, 3D LIC volumes can be represented by selectively emphasizing important regions of interest in the flow, enhancing depth perception and improving orientation perception [48,49,50] (Fig. 12.4).

Another related class of dense representations is based on *texture advection*. The basic idea is to represent a dense collection of particles in a texture and transport that texture according to the motion of particles [77,80]. For example, the Lagrangian coordinates for texture transport can be computed by a numerical scheme for convection equations [7]. The *motion map* [56] is an application of the texture-advection concept for animating 2D steady flows. The motion map contains a dense representation of the flow and the information required for animation. Lagrangian–Eulerian advection (LEA) [54] is a scheme for visualizing unsteady flows by integrating particle positions (i.e., the Lagrangian part) and advecting the color of the particles based on a texture representation (i.e., the Eulerian aspect). LEA can be extended to visualizing vertical motion in a 3D flow by means of time surfaces [38]. Texture advection is directly related to the texture-mapping capabilities of graphics hardware and therefore facilitates efficient implementations [53,129,130]. Another advantage of texture advection is the fact that both noise and dye advection can be handled in the same framework. Texture advection can also be applied to 3D flows [59,130].

Image-based flow visualization (IBFV) [124] is a recently developed variant of 2D texture advection. Not only is the (noise) texture transported along the flow, but a second texture is also blended into the advected texture at each time step. IBFV is a flexible tool that can imitate a wide variety of visualization styles. Another approach to the transport of a dense set of particles is based on *nonlinear diffusion* [28]. An initial noise image is smoothed along integral lines of a steady flow by diffusion, whereas the image is sharpened in the orthogonal direction. Nonlinear diffusion can be extended to the multiscale visualization of transport in time-dependent flows [12].

Finally, some 3D flow-visualization techniques adopt the idea of splatting, originally developed for volume rendering [132]. Even if some vector-splatting techniques do not rely on particle tracing, we have included them in this section because their visual appearance resembles dense curve-like structures. Anisotropic "scratches" can be modeled onto texture splats that are oriented along the flow to show the direction of the vector field [17]. *Line bundles* [79] use the splatting analogy to draw each data point with a precomputed set of rendered line segments. These semitransparent line bundles are composited together in a back-to-front order to achieve an anisotropic volume rendering result. For time-dependent flows, the animation of a large number of texture-

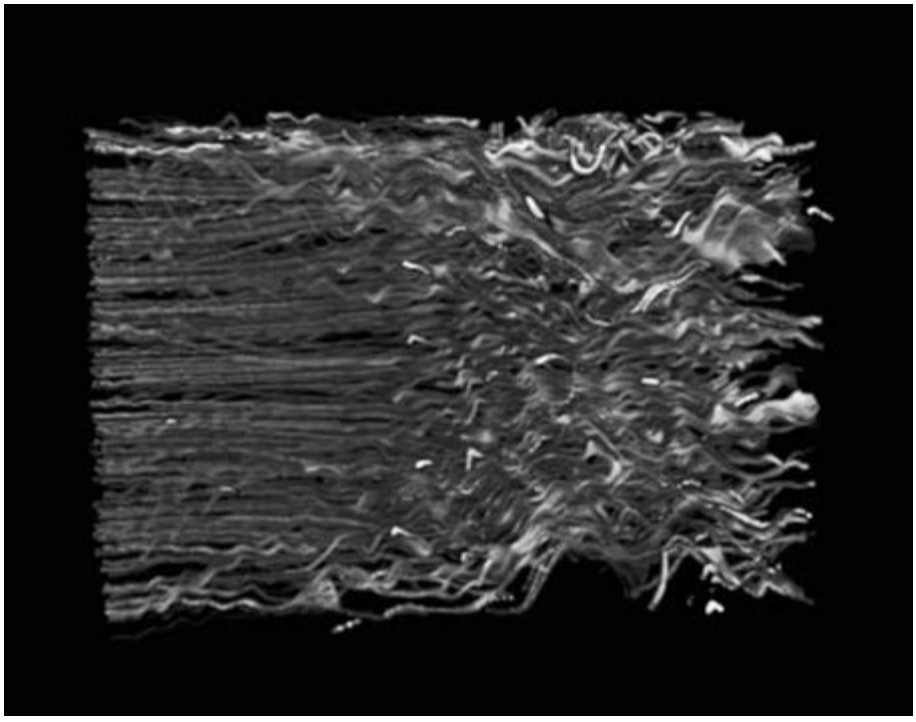

Figure 12.4 3D LIC with enhanced depth perception. Image courtesy of Victoria Interrante. (See also color insert.)

mapped particles along pathlines can be used [39]. For all splatting approaches, the density of representation depends on the number of splats.

12.8 Feature-Based Visualization Approaches

The visualization concepts discussed so far operate directly on the vector field. Therefore, it is the task of the user to identify the important features of the flow from such a visualization. Feature-based visualization approaches seek to compute a more abstract representation that already contains the important properties in a condensed form and suppresses superfluous information. In other words, an appropriate filtering process is chosen to reduce the amount of visual data presented to the user. Examples for this more abstract data are flow topology based on critical points, other flow features such as vortices and shock waves, or aggregated flow data via clustering.

After features are computed, the actual visual representation has to be considered. Different features have different attributes; to emphasize special attributes for each type of feature, suitable representations must be used. Glyphs or icons can be employed for vortices or for critical points and other topological features. Examples are ellipses or ellipsoids to encode the rotation speed and other attributes of vortices. A comprehensive presentation of feature-extraction and visualization techniques can be found in the survey [88].

Topology-based 2D vector-field visualization [44] aims to show only the essential information of the field. The qualitative structure of a vector field can be globally represented by portraying

its topology. The field's critical points and separatrices determine the nature of the flow. From a diagram of the topology, the complete flow can be inferred. Fig. 12.1d shows an example of a topology-based visualization. From a numerical point of view, the interpolation scheme is crucial for identifying critical points. Extended versions of topology-based representations make use of higher-order singularities [101] and C^1-continuous interpolation schemes [102]. Another extension is based on the detection of closed streamlines, a global property that is not detected by the aforementioned algorithms [133]. Topology-based visualization can also be extended to time-dependent vector fields by topology tracking [117]. The original topology-based techniques work well for datasets with small numbers of critical points. Turbulent flows computed on a high-resolution grid, however, may show a large number of critical points, leading to an overloaded visual representation. This issue is addressed by topology-simplification techniques [23,24,25,115,116,135], which remove some of the critical points and leave only the important features. For the visualization of 3D topology, suitable visual representations need to be used. For example, streamlines that are traced from appropriate positions close to critical points connect to other critical points or the boundary to display the topology, while glyphs can be used to visualize the various classes of critical points [37]. Topology can also serve as a means for determining the similarity between two different vector fields [3,67].

Vector-field clustering is another way to reduce the amount of visualization data. A large number of vectors of the original high-resolution field are combined into fewer vectors that approximately represent the vector field at a coarser resolution, leading to a visualization of aggregated data. An important issue is to establish appropriate error measures to control the way vectors are combined into clusters [70,114]. The extension of vector-field clustering to several levels of clustering leads to hierarchical representations [34,35,41]. In related approaches, segmentation [28], multiscale visualization [12], or topology-preserving smoothing based on level-set techniques [131] reduce the complexity of the displayed vector fields.

An important class of feature-detection algorithms is based on the identification of *vortices* and their respective *vortex cores*. Vortices are useful for identifying significant features in a fluid flow. One way of classifying vortex-detection techniques is the distinction between point-based, local techniques, which operate directly on the vector data set, and geometry-based, global techniques, which examine the properties of characteristic lines around vortices [97]. Local techniques build a measure for vortices from physical quantities of a fluid flow. An overview of these techniques [2,91] and more recent contributions [5,52,61,92] are available. A mathematical framework [87] makes it possible to unify several vortex-detection methods. Point-based methods are rather simple to compute but are more likely to miss some vortices. For example, weak vortices, which have a slow rotational component compared to the velocity of the core, are hard to detect by these techniques. Geometry-based, global methods [51,95,96] are usually associated with higher computational costs but allow a more robust detection or verification of vortices. A more detailed description of vortex-detection techniques can be found in Chapter 14.

Shock waves are another important feature of a fluid flow because they can increase drag and even cause structural failure. Shock waves are characterized by discontinuities in physical quantities, such as pressure, density, and velocity. Therefore, shock-detection algorithms are related to edge-detection methods known from image processing. A comparison of different techniques for shock extraction and visualization is available [73]. Flow *separation* and *attachment*, which occur when a flow abruptly moves away from or returns to a solid body, are other interesting features of a fluid flow. Attachment and separation lines on surfaces in a 3D flow can be automatically extracted based on a local analysis of the vector field by means of phase plane analysis [60].

Vortex cores, shock waves, and separation and attachment lines are examples of features that are tightly connected to an underlying physical model and cannot be derived from a generic vector-field description. Therefore, a profound understanding of the physical problem is necessary to develop measures for these kinds of features. Accordingly, a large body of research on feature extraction can be found in the literature on related topics of engineering and physics. Since a comprehensive collection of techniques and references on feature extraction is beyond the scope of this chapter, we refer to the survey [88] for more detailed information.

Acknowledgments

We would like to thank the following people for providing us with images: Victoria Interrante (University of Minnesota) for the image in Fig. 12.4; Gerik Scheuermann (Universität Kaiserslautern) for Fig. 12.1d; BMW Group and Martin Schulz for Fig. 12.3.

The first author thanks the Landesstiftung Baden-Württemberg for support; the second author acknowledges support from NSF under grant NSF-0083792.

References

1. D. C. Banks. Illumination in diverse codimensions. In *Proceedings of ACM SIGGRAPH '94*, pages 327–334, 1994.
2. D. C. Banks and B. A. Singer. Vortex tubes in turbulent flows: identification, representation, reconstruction. In *IEEE Visualization '94*, pages 132–139, 1994.
3. R. Batra and L. Hesselink. Feature comparisons of 3-D vector fields using earth mover's distance. In *IEEE Visualization '99*, pages 105–114, 1999.
4. H. Battke, D. Stalling, and H.-C. Hege. Fast line integral convolution for arbitrary surfaces in 3D. In *Visualization and Mathematics* (H.-C. Hege and K. Polthier, Eds.), pages 181–195. Berlin, Springer, 1997.
5. D. Bauer and R. Peikert. Vortex tracking in scale-space. In *EG/IEEE TCVG Symposium on Visualization '02*, pages 233–240, 2002.
6. B. Becker, D. A. Lane, and N. Max. Unsteady flow volumes. In *IEEE Visualization '95*, pages 329–335, 1995.
7. J. Becker and M. Rumpf. Visualization of time-dependent velocity fields by texture transport. In *EG Workshop on Visualization in Scientific Computing*, pages 91–101, 1998.
8. U. Bordoloi and H.-W. Shen. Hardware accelerated interactive vector field visualization: a level of detail approach. In *Eurographics '02*, pages 605–614, 2002.
9. E. Boring and A. Pang. Directional flow visualization of vector fields. In *IEEE Visualization '96*, pages 389–392, 1996.
10. M. Brill, H. Hagen, H.-C. Rodrian, W. Djatschin, and S. V. Klimenko. Streamball techniques for flow visualization. In *IEEE Visualization '94*, pages 225–231, 1994.
11. S. Bryson and C. Levit. The virtual wind tunnel. *IEEE Computer Graphics and Applications*, 12(4):25–34, 1992.
12. D. Bürkle, T. Preußer, and M. Rumpf. Transport and anisotropic diffusion in time-dependent flow visualization. In *IEEE Visualization '01*, pages 61–67, 2001.
13. B. Cabral and C. Leedom. Highly parallel vector visualization using line integral convolution. In *SIAM Conference on Parallel Processing for Scientific Computing*, pages 802–807, 1995.
14. B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. In *Proceedings of ACM SIGGRAPH '93*, pages 263–270, 1993.
15. W. Cai and P.-A. Heng. Principal stream surfaces. In *IEEE Visualization '97*, pages 75–80, 1997.
16. J. Clyne and J. M. Dennis. Interactive direct volume rendering of time-varying data. In *EG/IEEE TCVG Symposium on Visualization '99*, pages 109–120, 1999.
17. R. Crawfis and N. Max. Texture splats for 3D scalar and vector field visualization. In *IEEE Visualization '93*, pages 261–266, 1993.
18. R. Crawfis, H.-W. Shen, and N. Max. Flow visualization techniques for CFD using volume rendering. In *9th International Symposium on Flow Visualization*, pages 64/1–64/10, 2000.
19. W. C. de Leeuw. Divide and conquer spot noise. In *Supercomputing '97 Conference*, pages 12–24, 1997.
20. W. C. de Leeuw, H.-G. Pagendarm, F. H. Post, and B. Walter. Visual simulation of experimental oil-flow visualization by spot noise images from numerical flow simulation. In *EG Workshop on Visualization in Scientific Computing*, pages 135–148, 1995.

21. W. C. de Leeuw, F. H. Post, and R. W. Vaatstra. Visualization of turbulent flow by spot noise. In *EG Workshop on Virtual Environments and Scientific Visualization '96*, pages 286–295, 1996.
22. W. C. de Leeuw and R. van Liere. Comparing LIC and spot noise. In *IEEE Visualization '98*, pages 359–365, 1998.
23. W. C. de Leeuw and R. van Liere. Collapsing flow topology using area metrics. In *IEEE Visualization '99*, pages 349–354, 1999.
24. W. C. de Leeuw and R. van Liere. Visualization of global flow structures using multiple levels of topology. In *EG/IEEE TCVG Symposium on Visualization '99*, pages 45–52, 1999.
25. W. C. de Leeuw and R. van Liere. Multi-level topology for flow visualization. *Computers and Graphics*, 24(3):325–331, 2000.
26. W. C. de Leeuw and J. J. van Wijk. A probe for local flow field visualization. In *IEEE Visualization '93*, pages 39–45, 1993.
27. W. C. de Leeuw and J. J. van Wijk. Enhanced spot noise for vector field visualization. In *IEEE Visualization '95*, pages 233–239, 1995.
28. U. Diewald, T. Preußer, and M. Rumpf. Anisotropic diffusion in vector field visualization on Euclidean domains and surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):139–149, 2000.
29. D. Dovey. Vector plots for irregular grids. In *IEEE Visualization '95*, pages 248–253, 1995.
30. D. S. Ebert, R. Yagel, J. Scott, and Y. Kurzion. Volume rendering methods for computational fluid dynamics visualization. In *IEEE Visualization '94*, pages 232–239, 1994.
31. L. K. Forssell and S. D. Cohen. Using line integral convolution for flow visualization: Curvilinear grids, variable-speed animation, and unsteady flows. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):133–141, 1995.
32. T. Frühauf. Raycasting vector fields. In *IEEE Visualization '96*, pages 115–120, 1996.
33. A. Fuhrmann and E. Gröller. Real-time techniques for 3D flow visualization. In *IEEE Visualization '98*, pages 305–312, 1998.
34. H. Garcke, T. Preußer, M. Rumpf, A. Telea, U. Weikard, and J. J. van Wijk. A phase field model for continuous clustering on vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):230–241, 2001.
35. H. Garcke, T. Preußer, M. Rumpf, A. Telea, U. Weikard, and J. van Wijk. A continuous clustering method for vector fields. In *IEEE Visualization '00*, pages 351–358, 2000.
36. T. Glau. Exploring instationary fluid flows by interactive volume movies. In *EG/IEEE TCVG Symposium on Visualization '99*, pages 277–283, 1999.
37. A. Globus, C. Levit, and T. Lasinski. A tool for visualizing the topology of 3D vector fields. In *IEEE Visualization '91*, pages 33–40, 1991.
38. J. Grant, G. Erlebacher, and J. O'Brien. Case study: visualizing ocean flow vertical motions using Lagrangian–Eulerian time surfaces. In *IEEE Visualization '02*, pages 529–532, 2002.
39. S. Guthe, S. Gumhold, and W. Straßer. Interactive visualization of volumetric vector fields using texture based particles. In *WSCG 2002 Conference Proceedings*, pages 33–41, 2002.
40. R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, and D. Weiskopf. The state of the art in flow visualization: dense and texture-based techniques. *Computer Graphics Forum*, 23(2):143–161, 2004.
41. B. Heckel, G. Weber, B. Hamann, and K. I. Joy. Construction of vector field hierarchies. In *IEEE Visualization '99*, pages 19–25, 1999.
42. H.-C. Hege and D. Stalling. Fast LIC with piecewise polynomial filter kernels. *Mathematical Visualization* (H.-C. Hege and K. Polthier, Eds.), pages 295–314. Berlin, Springer, 1998.
43. W. Heidrich, R. Westermann, H.-P. Seidel, and T. Ertl. Applications of pixel textures in visualization and realistic image synthesis. In *ACM Symposium on Interactive 3D Graphics*, pages 127–134, 1999.
44. J. Helman and L. Hesselink. Representation and display of vector field topology in fluid flow data sets. *Computer*, 22(8):27–36, 1989.
45. M. W. Hirsch. *Differential Topology*, 6th Ed. Berlin, Springer, 1997.
46. M. W. Hirsch and S. Smale. *Differential Equations, Dynamical Systems, and Linear Algebra*. New York, Academic Press, 1974.
47. J. P. M. Hultquist. Interactive numerical flow visualization using stream surfaces. *Computing Systems in Engineering*, 1(2–4):349–353, 1990.
48. V. Interrante. Illustrating surface shape in volume data via principal direction-driven 3D line integral convolution. In *Proceedings of ACM SIGGRAPH 97*, pages 109–116, 1997.
49. V. Interrante and C. Grosch. Strategies for effectively visualizing 3D flow with volume LIC. In *IEEE Visualization '97*, pages 421–424, 1997.
50. V. Interrante and C. Grosch. Visualizing 3D flow. *IEEE Computer Graphics and Applications*, 18(4):49–53, 1998.
51. M. Jiang, R. Machiraju, and D. Thompson. Geometric verification of swirling features in

flow fields. In *IEEE Visualization '02*, pages 307–314, 2002.
52. M. Jiang, R. Machiraju, and D. Thompson. A novel approach to vortex core region detection. In *EG/IEEE TCVG Symposium on Visualization '02*, pages 217–225, 2002.
53. B. Jobard, G. Erlebacher, and M. Y. Hussaini. Hardware-accelerated texture advection for unsteady flow visualization. In *IEEE Visualization '00*, pages 155–162, 2000.
54. B. Jobard, G. Erlebacher, and M. Y. Hussaini. Lagrangian–Eulerian advection for unsteady flow visualization. In *IEEE Visualization '01*, pages 53–60, 2001.
55. B. Jobard and W. Lefer. Creating evenly spaced streamlines of arbitrary density. In *EG Workshop on Visualization in Scientific Computing*, pages 43–55, 1997.
56. B. Jobard and W. Lefer. The motion map: efficient computation of steady flow animations. In *IEEE Visualization '97*, pages 323–328, 1997.
57. B. Jobard and W. Lefer. Unsteady flow visualization by animating evenly spaced streamlines. In *Eurographics '00*, pages 31–39, 2000.
58. B. Jobard and W. Lefer. Multiresolution flow visualization. In *WSCG 2001 Conference Proceedings*, pages P34–P37, 2001.
59. D. Kao, B. Zhang, K. Kim, and A. Pang. 3D flow visualization using texture advection. In *IASTED Conference on Computer Graphics and Imaging 01 (CGIM)*, pages 252–257, 2001.
60. D. N. Kenwright. Automatic detection of open and closed separation and attachment lines. In *IEEE Visualization '98*, pages 151–158, 1998.
61. D. N. Kenwright and R. Haimes. Automatic vortex core detection. *IEEE Computer Graphics and Applications*, 18(4):70–74, 1998.
62. D. N. Kenwright and D. A. Lane. Interactive time-dependent particle tracing using tetrahedral decomposition. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):120–129, 1996.
63. L. Khouas, C. Odet, and D. Friboulet. 2D vector field visualization using furlike texture. In *EG/IEEE TCVG Symposium on Visualization '99*, pages 35–44, 1999.
64. M.-H. Kiu and D. C. Banks. Multi-frequency noise for LIC. In *IEEE Visualization '96*, pages 121–126, 1996.
65. R. V. Klassen and S. J. Harrington. Shadowed hedgehogs: A technique for visualizing 2D slices of 3D vector fields. In *IEEE Visualization '91*, pages 148–153, 1991.
66. S. Lang. *Differential and Riemannian Manifolds*, 3rd Ed. New York, Springer, 1995.
67. Y. Lavin, R. Batra, and L. Hesselink. Feature comparisons of vector fields using Earth mover's distance. In *IEEE Visualization '98*, pages 103–109, 1998.
68. Z. P. Liu and R. J. Moorhead. AUFLIC: An accelerated algorithm for unsteady flow line integral convolution. In *EG/IEEE TCVG Symposium on Visualization '02*, pages 43–52, 2002.
69. S. K. Lodha, A. Pang, R. E. Sheehan, and C. M. Wittenbrink. UFLOW: visualizing uncertainty in fluid flow. In *IEEE Visualization '96*, pages 249–254, 1996.
70. S. K. Lodha, J. C. Renteria, and K. M. Roskin. Topology preserving compression of 2D vector fields. In *IEEE Visualization '00*, pages 343–350, 2000.
71. H. Löffelmann and E. Gröller. Enhancing the visualization of characteristic structures in dynamical systems. In *EG Workshop on Visualization in Scientific Computing*, pages 59–68, 1998.
72. H. Löffelmann, L. Mroz, E. Gröller, and W. Purgathofer. Stream arrows: enhancing the use of stream surfaces for the visualization of dynamical systems. *The Visual Computer*, 13(8):359–369, 1997.
73. K.-L. Ma, J. V. Rosendale, and W. Vermeer. 3D shock wave visualization on unstructured grids. In *1996 Volume Visualization Symposium*, pages 87–94, 1996.
74. X. Mao, Y. Hatanaka, H. Higashida, and A. Imamiya. Image-guided streamline placement on curvilinear grid surfaces. In *IEEE Visualization '98*, pages 135–142, 1998.
75. X. Mao, L. Hong, A. Kaufman, N. Fujita, and M. Kikukawa. Multi-granularity noise for curvilinear grid LIC. In *Graphics Interface*, pages 193–200, 1998.
76. X. Mao, M. Kikukawa, N. Fujita, and A. Imamiya. Line integral convolution for 3D surfaces. In *EG Workshop on Visualization in Scientific Computing*, pages 57–69, 1997.
77. N. Max and B. Becker. Flow visualization using moving textures. In *Proceedings of the ICASE/LaRC Symposium on Visualizing Time-Varying Data*, pages 77–87, 1995.
78. N. Max, B. Becker, and R. Crawfis. Flow volumes for interactive vector field visualization. In *IEEE Visualization '93*, pages 19–24, 1993.
79. N. Max, R. Crawfis, and C. Grant. Visualizing 3D velocity fields near contour surfaces. In *IEEE Visualization '94*, pages 248–255, 1994.
80. N. Max, R. Crawfis, and D. Williams. Visualizing wind velocities by advecting cloud textures. In *IEEE Visualization '92*, pages 179–184, 1992.

81. J. W. Milnor. *Topology from the Differentiable Viewpoint*. Charlottesville, VA, University Press of Virginia, 1965.
82. G. M. Nielson, H. Hagen, and H. Müller. *Scientific Visualization: Overviews, Methodologies, and Techniques*. Los Alamitos, CA, IEEE Computer Society Press, 1997.
83. G. M. Nielson and I.-H. Jung. Tools for computing tangent curves for linearly varying vector fields over tetrahedral domains. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):360–372, 1999.
84. A. Okada and D. Kao. Enhanced line integral convolution with flow feature detection. In *Proceedings of IS&T/SPIE Electronic Imaging '97*, pages 206–217, 1997.
85. K. Ono, H. Matsumoto, and R. Himeno. Visualization of thermal flows in an automotive cabin with volume rendering method. In *EG/IEEE TCVG Symposium on Visualization '01*, pages 301–308, 2001.
86. A. T. Pang, C. M. Wittenbrink, and S. K. Lodha. Approaches to uncertainty visualization. *The Visual Computer*, 13(8):370–390, 1997.
87. R. Peikert and M. Roth. The "parallel vectors" operator—a vector field visualization primitive. In *IEEE Visualization '99*, pages 263–270, 1999.
88. F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. The state of the art in flow visualization: feature extraction and tracking. *Computer Graphics Forum*, 22(4): 775–792, 2003.
89. C. Rezk-Salama, P. Hastreiter, C. Teitzel, and T. Ertl. Interactive exploration of volume line integral convolution based on 3D-texture mapping. In *IEEE Visualization '99*, pages 233–240, 1999.
90. C. P. Risquet. Visualizing 2D flows: integrate and draw. In *EG Workshop on Visualization in Scientific Computing (Participant Edition)*, pages 57–67, 1998.
91. M. Roth and R. Peikert. Flow visualization for turbomachinery design. In *IEEE Visualization '96*, pages 381–384, 1996.
92. M. Roth and R. Peikert. A higher-order method for finding vortex core lines. In *IEEE Visualization '98*, pages 143–150, 1998.
93. S. Röttger, M. Kraus, and T. Ertl. Hardware-accelerated volume and isosurface rendering based on cell projection. In *IEEE Visualization '00*, pages 109–116, 2000.
94. I. A. Sadarjoen, A. J. de Boer, F. H. Post, and A. E. Mynett. Patricle tracing in σ-transformed grids using tetrahedral 6-decomposition. In *EG Workshop on Visualization in Scientific Computing*, pages 71–80, 1998.
95. I. A. Sadarjoen and F. H. Post. Geometric methods for vortex extraction. In *EG/IEEE TCVG Symposium on Visualization '99*, pages 53–62, 1999.
96. I. A. Sadarjoen and F. H. Post. Detection, quantification, and tracking of vortices using streamline geometry. *Computers and Graphics*, 24(3):333–341, 2000.
97. I. A. Sadarjoen, F. H. Post, B. Ma, D. C. Banks, and H.-G. Pagendarm. Selective visualization of vortices in hydrodynamic flows. In *IEEE Visualization '98*, pages 419–422, 1998.
98. A. Sanna, B. Montrucchio, and R. Arina. Visualizing unsteady flows by adaptive streaklines. In *WSCG 2000 Conference Proceedings*, pages 84–91, 2000.
99. A. Sanna, B. Montrucchio, and P. Montuschi. A survey on visualization of vector fields by texture-based methods. *Recent Res. Devel. Pattern Rec.*, 1:13–27, 2000.
100. G. Scheuermann, H. Burbach, and H. Hagen. Visualizing planar vector fields with normal components using line integral convolution. In *IEEE Visualization '99*, pages 255–261, 1999.
101. G. Scheuermann, H. Hagen, H. Krüger, M. Menzel, and A. Rockwood. Visualization of higher order singularities in vector fields. In *IEEE Visualization '97*, pages 67–74, 1997.
102. G. Scheuermann, X. Tricoche, and H. Hagen. C^1-interpolation for vector field topology visualization. In *IEEE Visualization '99*, pages 271–278, 1999.
103. W. J. Schroeder, C. R. Volpe, and W. E. Lorensen. The stream polygon: A technique for 3D vector field visualization. In *IEEE Visualization '91*, pages 126–132, 1991.
104. M. Schulz, F. Reck, W. Bartelheimer, and T. Ertl. Interactive visualization of fluid dynamics simulations in locally refined Cartesian grids. In *IEEE Visualization '99*, pages 413–416, 1999.
105. H.-W. Shen, C. R. Johnson, and K.-L. Ma. Visualizing vector fields using line integral convolution and dye advection. In *1996 Volume Visualization Symposium*, pages 63–70, 1996.
106. H.-W. Shen and D. L. Kao. A new line integral convolution algorithm for visualizing time-varying flow fields. *IEEE Transactions on Visualization and Computer Graphics*, 4(2):98–108, 1998.
107. P. Shirley and A. Tuchman. A polygonal approximation to direct scalar volume rendering. In *Workshop on Volume Visualization '90*, pages 63–70, 1990.
108. D. Stalling and H.-C. Hege. Fast and resolution independent line integral convolution.

In *Proceedings of ACM SIGGRAPH '95*, pages 249–256, 1995.
109. A. Sundquist. Dynamic line integral convolution for visualizing streamline evolution. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):273–282, 2003.
110. C. Teitzel and T. Ertl. New approaches for particle tracing on sparse grids. In *EG/IEEE TCVG Symposium on Visualization '99*, pages 73–84, 1999.
111. C. Teitzel, R. Grosso, and T. Ertl. Efficient and reliable integration methods for particle tracing in unsteady flows on discrete meshes. In *EG Workshop on Visualization in Scientific Computing*, pages 31–41, 1997.
112. C. Teitzel, R. Grosso, and T. Ertl. Line integral convolution on triangulated surfaces. In *WSCG 1997 Conference Proceedings*, pages 572–581, 1997.
113. C. Teitzel, R. Grosso, and T. Ertl. Particle tracing on sparse grids. In *EG Workshop on Visualization in Scientific Computing*, pages 81–90, 1998.
114. A. Telea and J. J. van Wijk. Simplified representation of vector fields. In *IEEE Visualization '99*, pages 35–42, 1999.
115. X. Tricoche, G. Scheuermann, and H. Hagen. Continuous topology simplification of planar vector fields. In *IEEE Visualization '01*, pages 159–166, 2001.
116. X. Tricoche, G. Scheuermann, and H. Hagen. A topology simplification method for 2D vector fields. In *IEEE Visualization '00*, pages 359–366, 2000.
117. X. Tricoche, T. Wischgoll, G. Scheuermann, and H. Hagen. Topology tracking for the visualization of time-dependent 2D flows. *Computers and Graphics*, 26(2):249–257, 2002.
118. G. Turk and D. Banks. Image-guided streamline placement. In *Proceedings of ACM SIGGRAPH '96*, pages 453–460, 1996.
119. S.-K. Ueng, C. Sikorski, and K.-L. Ma. Efficient streamline, streamribbon, and streamtube constructions on unstructured grids. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):100–110, 1996.
120. S. P. Uselton. Volume rendering for computational fluid dynamics: initial results. Technical Report RNR-91-026, NASA Ames Research Center, 1991.
121. J. J. van Wijk. Spot noise—texture synthesis for data visualization. *Computer Graphics (Proceedings of AGM SIGGRAPH '91)*, 25:309–318, 1991.
122. J. J. van Wijk. Flow visualization with surface particles. *IEEE Computer Graphics and Applications*, 13(4):18–24, 1993.
123. J. J. van Wijk. Implicit stream surfaces. In *IEEE Visualization '93*, pages 245–252, 1993.
124. J. J. van Wijk. Image based flow visualization. *ACM Transactions on Graphics*, 21(3):745–754, 2002.
125. V. Verma, D. Kao, and A. Pang. PLIC: bridging the gap between streamlines and LIC. In *IEEE Visualization '99*, pages 341–348, 1999.
126. V. Verma, D. Kao, and A. Pang. A flow-guided streamline seeding strategy. In *IEEE Visualization '00*, pages 163–170, 2000.
127. R. Wegenkittl and E. Gröller. Fast oriented line integral convolution for vector field visualization via the Internet. In *IEEE Visualization '97*, pages 309–316, 1997.
128. R. Wegenkittl, E. Gröller, and W. Purgathofer. Animating flow fields: Rendering of oriented line integral convolution. In *Computer Animation '97*, pages 15–21, 1997.
129. D. Weiskopf, G. Erlebacher, M. Hopf, and T. Ertl. Hardware-accelerated Lagrangian–Eulerian texture advection for 2D flow visualization. In *Vision, Modeling, and Visualization VMV '02 Conference*, pages 77–84, 2002.
130. D. Weiskopf, M. Hopf, and T. Ertl. Hardware-accelerated visualization of time-varying 2D and 3D vector fields by texture advection via programmable per-pixel operations. In *Vision, Modeling, and Visualization VMV '01 Conference*, pages 439–446, 2001.
131. R. Westermann, C. Johnson, and T. Ertl. Topology-preserving smoothing of vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):222–229, 2001.
132. L. Westover. Footprint evaluation for volume rendering. *Computer Graphics (Proceedings of ACM SIGGRAPH '90)*, 24:367–376, 1990.
133. T. Wischgoll and G. Scheuermann. Detection and visualization of closed streamlines in planar flows. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):165–172, 2001.
134. C. M. Wittenbrink, A. T. Pang, and S. K. Lodha. Glyphs for visualizing uncertainty in vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 2(3):266–279, 1996.
135. P. C. Wong, H. Foote, R. Leung, E. Jurrus, D. Adams, and J. Thomas. Vector field simplification—a case study of visualizing climate modeling and simulation data sets. In *IEEE Visualization '00*, pages 485–488, 2000.

136. M. Zöckler, D. Stalling, and H.-C. Hege. Interactive visualization of 3D-vector fields using illuminated stream lines. In *IEEE Visualization '96*, pages 107–113, 1996.

137. M. Zöckler, D. Stalling, and H.-C. Hege. Parallel line integral convolution. In *EG Workshop on Parallel Graphics and Visualisation*, pages 111–127, 1996.