6.5 VISUALIZATION OF VECTOR FIELDS

C. TEITZEL, M. HOPF

In the foregoing part of this chapter techniques for the visualization of scalar field data have been presented. These methods are not suitable, in general, for the exploration of vector field data. Hence, different concepts have been developed in order to reveal the information contained in this kind of data.

Visualizing vector field data is challenging because no existing natural representation can visually convey large amounts of three-dimensional directional information. In fluid flow experiments, external materials such as dye, hydrogen bubbles, or heat energy are injected into the flow. The advection of these external materials creates flow patterns which highlight the inherent structure of the field. Analogues to these experimental techniques have been adopted by scientific visualization researchers.

In the reminder of this chapter we will summarize some of the basic approaches for the visualization of flow fields. We will sketch ideas how to use numerical methods and three-dimensional computer graphics techniques to produce graphical icons such as arrows, motion particles, stream lines, stream ribbons, and stream tubes that act as three-dimensional depth cues. While these techniques are effective in revealing the flow field's local features, the inherent two-dimensional display of the computer screen and its limited spatial resolution restrict the number of graphical icons that can be displayed at one time. In order to overcome these limitations additional techniques for flow field visualization including global imaging techniques have been developed. These techniques can successfully illustrate the global behavior of vector fields; however, it is difficult when using such methods to control the tremendous information density in a way that effectively depicts both the direction structure of the flow and the flow magnitude.

In the following we will deal with three-dimensional flow or vector fields, which associate a vector with each point of the underlying domain. In general, the physical idea of flow can be represented by the mathematical concept of a *flux*. Detailed descriptions of this conception can be found in nearly every textbook about differential topology and in books about ordinary differential equations, e.g. [20, 175, 419, 482].

A more common point of view is to look at the flow field as a family of differentiable curves parameterized over a certain domain: Then, the curve

$$\begin{aligned} \alpha_x : \mathbb{R} &\longrightarrow M \\ t &\longmapsto \Phi_t(x) = \Phi(t, x) \end{aligned}$$

is called *integral curve* or *flow line* of x. The image $\alpha_x(\mathbb{R})$ of the integral curve is called *orbit*, *path*, or *trajectory* of x. If a flux, Φ, is given on a manifold, M, then every point of M belongs to exactly one orbit. In order to get a better geometrical understanding of a given flux or a physical flow respectively, there are two different ways to look at the flux. On the one hand, we can study how the motions Φ_t act on the manifold M. That is, the deformations of lines and volumes by the flux Φ_t are studied. On the other hand, we can look for an overview over the shape of all orbits (see Section 6.5.1). There are three kinds of orbits: An integral curve $\alpha_x : \mathbb{R} \to M$ of a given flux is either an injective immersion, a periodic immersion, i.e. α_x is immersion and there exists

$p > 0$ with $\alpha_x(t + p) = \alpha_x(t)$ for all $t \in \mathbb{R}$, or α_x is constant, $\alpha_x(t) = x$ for all t. In the latter case x is called *fixed-point* of the flux.

The actual aim in flow visualization is to analyze and to display the properties of flows. The data, which are received from measurements or numerical simulations, consist of velocity information. In the following we will outline a variety of different methods, which can be used successfully to illustrate the local or global behavior of vector fields.

6.5.1 Velocity Field and Flux Visualization

According to the last section, a flow can be analyzed by visualizing its velocity field, orbits, or motion. A number of different techniques have been evolved to make use of these three notations.

Arrow Plots. A traditional standard technique for visualizing a flow is to visualize its velocity field directly by drawing small arrows at discrete grid points. This simple arrow plot algorithm is quite fast but has some disadvantages.

If the arrow plot method is applied to curvilinear or unstructured grids by drawing the arrows at the grid nodes, more arrows are placed in regions where the cells are small than in areas where the cells are large. This variation in arrow density is unrelated to the velocity field itself. A second drawback is that regularity in the grid causes distracting patterns in the output image. Finally, the user has no control over the global arrow density.

A solution to these problems is to re-sample the velocity field in computational space (C-space) or physical space (P-space) and to generate a random distribution of arrows. Thereby, the global arrow density can be chosen by the user. For a detailed description of this approach we refer to [187].

Glyphs. A more advanced method for visualizing the velocity field of a flow is using glyphs instead of arrows. A glyph is a local flow probe that shows some more parameters of the field. Besides the actual velocity, information about the Jacobian of the velocity is revealed. By decomposing the Jacobian into meaningful components and by their mapping using metaphors that are easy to understand, the matrix is presented in an intuitively way. The result is that the probe presents information like velocity, acceleration, curvature, shear, convergence (or divergence), and local rotation, also called torsion by some authors. This iconic visualization method was developed by de Leeuw and van Wijk [426] in 1993.

Glyphs can be used as a tool to study details of a flow field rather than to globally visualize a flow. On the one hand, it is an advantage of this technique that each probe supplies the user with a vast quantity of information. On the other hand, these icons are at first glance difficult to interpret and some users feel overwhelmed by the flood of information.

Topological Methods. Topological methods combine two of the three mentioned approaches for analyzing a given flux by visualizing properties of both the velocity field and the orbits. On the one hand, the singularities of the velocity field are considered,

on the other hand, the shape of orbits near fixed-points of the flux. In 1989 Helman and Hesselink [315] introduced techniques for visualizing the topology of vector fields, Since then this method has been of great interest in the visualization community [316]. The concept of Clifford algebra was introduced in order to detect higher order singularities [609] and to handle nonlinear vector fields [608]. Additionally, topological methods can be extended from vector fields to tensor fields [165, 421].

Particle Methods. Particle methods enjoy a good reputation in flow visualization, since they can display orbits or motion of a flux depending on the initial geometry. Both orbits and motions provide a good impression of the fluid flow. It is possible to depict details and to show global behavior as well. Given the velocity field $\boldsymbol{v}(\boldsymbol{x}, t) := \dot{\Phi}(t, \boldsymbol{x})$, the orbit, *path line*, or trajectory of $\boldsymbol{x}_0 \in M$ is given by the solution of an initial value problem for the following ordinary differential equation:

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{v}(\boldsymbol{x}(t), t), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0. \tag{6.7}$$

Physically, such a path line corresponds to a long time exposure photograph of an illuminated fluid particle. *Stream lines* correspond to the solution of the differential equation:

$$\frac{d\boldsymbol{x}(s)}{ds} = \boldsymbol{v}(\boldsymbol{x}(s), t), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0, \tag{6.8}$$

where the time t is treated as constant and s is the parameter of the resulting curve $\boldsymbol{x}(s)$. That is, we take a snapshot of the vector field $\boldsymbol{v}$ at time t. Stream lines are tangential to the instantaneous velocity at every point, except at points where $\boldsymbol{v} = 0$.

The image at time t of the *streak line* passing through the point $\boldsymbol{x}_1$ is the curve formed by all the particles which happened to pass by $\boldsymbol{x}_1$ during the time $0 < t_1 < t$. Physically, a streak line corresponds to the curve traced out by a non-diffusive tracer injected at the position $\boldsymbol{x}_1$. Note that the particles of the tracer move according to Equation 6.7. If a flow is steady, i.e. it lacks explicit time dependence, then path, stream, and streak lines coincide.

As already mentioned, the orbits of a fluid flow can be visualized applying Equation 6.7. The same equation can now be employed for displaying motions of the flow. Therefore, a set of connected particles is traced instead of a single particle. Lines, surfaces, or volumes can be considered as such sets. That is, we wonder what the evolution of the initial start configuration will be.

If all points of a given line are traced according to Equation 6.7, the result will be a stream ribbon [51, 386], a stream tube [177], or a stream surface [347]. In 1992 Hultquist improved stream surfaces by a new tiling technique and by splitting divergent ribbons [348]. In addition, Stolk and van Wijk introduced the representation of stream surfaces and tubes via surface particles [673]. Moreover, implicit stream surfaces were proposed by van Wijk [754] in order to overcome difficulties with irregular topologies of originating curves and surfaces.

With stream surfaces a good insight into the structure of the flow field can be achieved, because hidden surface elimination and shading can be used to provide cues

on depth and orientation. Furthermore, local parameters of the field or other variables can be mapped onto the surface. On the other hand, great efforts have to be undertaken in order to solve problems with convergent, divergent, or shear flows, and with flows around obstacles.

Flow volumes [475] are the volumetric equivalent of stream surfaces. That means, a polygon is given as source object and traced. The resulting flow volume is divided into a collection of tetrahedra, which are rendered by a method of Shirley and Tuchman [644].

Therefore, it is possible to render semi-transparent tetrahedra using hardware texturing and compositing in order to generate images which create the impression as if smoke were released into a gas flow.

Another approach dealing with surfaces as source objects is the method of *time surfaces* [195]. Starting with a surface at a given time, the particles of this surface are traced and further surfaces are rendered after discrete time steps. This time, the different surfaces will not be connected. Bye the way, the particles that lie on the same surface are particles of the same age.

Finally, volumes like balls or hexahedra can be used as start object. Of course, it is sufficient to trace particles of the surface of a closed volume, for instance a sphere is traced instead of a ball. The shape of the volumetric source object is distorted by the flow, which gives an impression of the stretching within the flow field. Floating volumes were presented by Duvenbeck and Schmidt [195] and Stolk and van Wijk [673].

A lot of other techniques for flow visualization based upon particle methods have been presented in the last years, e.g. stream polygons by Schroeder et al. [623] and the stream ball technique by Brill et al. [81].

Line Integral Convolution. As mentioned in the previous paragraphs, vector fields can be visualized in a number of different manners. The approaches presented so far are restricted to a rather coarse spatial resolution. In contrast to them, texture-based methods achieve a much higher resolution. Line integral convolution (LIC) is an effective and versatile technique for visualizing flow fields with small scale structures.

In an early texture-like method, introduced by van Wijk [753] in 1991, oval spots with white noise are distorted along a straight line segment oriented parallel to the local vector direction. LIC itself was introduced by Cabral and Leedom [103] in 1993. In their algorithm convolution takes place along curved stream line segments. In 1995 Stalling and Hege [662] made LIC much faster, more accurate and independent of resolution. Due to these improvements LIC turned out to be very suitable for displaying vector fields on two-dimensional surfaces and became very popular. Hence, a vast quantity of different algorithm and improvements have been developed in the last years. In 1994 Forssell [231] presented an extension that allows to map flat LIC images onto curvilinear surfaces in three dimensions. A problem of this method is the distortion of length during the mapping process. In 1997 Teitzel et al. [689] solved this problem by computing LIC images directly on triangulated surfaces in three-dimensional space without mapping. Another method for creating LIC images on surfaces in three-dimensional space was presented by Mao et al. [467] using solid texturing. Wegenkittl et al. [740] introduced oriented LIC in order to visualize the orientation of the flow and Risquet [573] presented a drastic simplification for acceler-

ating the imaging process. Many other authors have been working on enhancements by color coding or animating LIC, by accelerating the image generation, or by developing especially adapted techniques for applying LIC to unsteady flows. A comparison of LIC and recently enhanced spot noise techniques can be found in [425].

Considering volumetric data, though three-dimensional LIC volumes can be computed in the same manner as two-dimensional LIC images, volume LIC is scarcely used because of difficulties to depict inner structures of the vector field. Modern high-end graphics workstations provide a high number of tri-linear interpolation operations per second and thereby allow to perform direct volume rendering (compare Section 6.4.5) at high image quality and interactive frame rates. Although the ability to interactively manipulate the three-dimensional volume greatly improves the perception of the inner structures, the stream lines inside a three-dimensional LIC texture are too dense and intricate to be visualized as a whole. Semi-transparency and the application of sparse input textures as proposed in [353] can enhance the resulting LIC texture.

Animation is an intuitive way to add information about the absolute value of velocity to static LIC images. Techniques used to animate two-dimensional LIC, which compute LIC textures for each time step, are less applicable for three-dimensional LIC, because of the great computational expense and the immense amount of data. To avoid the performance penalty and the high memory requirements that come with loading and storing large pre-computed three-dimensional textures, the idea was not to animate the three-dimensional LIC itself, but to use an animated clipping object instead. Rezk-Salama et al. provided two different approaches [568], which both use a single three-dimensional LIC texture and a set of clipping objects. To display animated three-dimensional flow at interactive frame rates texture-based volume rendering is performed.

Basics of Line Integral Convolution. The LIC algorithm filters an input volume along path or stream lines of a given vector field and generates a three-dimensional texture as output. In most cases in scientific visualization a texture with white noise is used as input. The Intensity I of an output texture voxel located at $\boldsymbol{x}_0 = \boldsymbol{\sigma}(s_0)$ is

$$I(\boldsymbol{x}_0) = \int_{s_0-L}^{s_0+L} k(s - s_0) T(\boldsymbol{\sigma}(s)) ds, \tag{6.9}$$

where $\boldsymbol{\sigma}(s)$ denotes a stream line of the vector field parameterized by arc length, T the intensity of the input texture and k a filter kernel. If we choose a constant filter kernel k and consider that T is constant at each voxel, the convolution integral can be computed by sampling the input texture T at locations $\boldsymbol{x}_i$ along the stream line $\boldsymbol{\sigma}(s)$:

$$I(\boldsymbol{x}_0) = k \sum_{i=-n}^{n} T(\boldsymbol{x}_i), \tag{6.10}$$

where we choose $k = 1/(2n + 1)$ to normalize the intensity. The convolution causes voxel intensities to be highly correlated along individual stream lines but independent in directions perpendicular to them. In the resulting images the stream lines are clearly visible.

6.5.2 Vortex Visualization

Up to now, we have analyzed the flow and its velocity field. What can be done next is to investigate the derivative of the velocity field, the Jacobian field of the velocity. In order to do this, we firstly decompose the Jacobian into a symmetric matrix $\mathbf{\Lambda}$ and a skew-symmetric one $\mathbf{\Omega}$:

$$\begin{aligned} D_{\boldsymbol{x}}\boldsymbol{v} &= \underbrace{\frac{1}{2}\left(D_{\boldsymbol{x}}\boldsymbol{v} + (D_{\boldsymbol{x}}\boldsymbol{v})^T\right)}_{\mathbf{\Lambda}} + \underbrace{\frac{1}{2}\left(D_{\boldsymbol{x}}\boldsymbol{v} - (D_{\boldsymbol{x}}\boldsymbol{v})^T\right)}_{\mathbf{\Omega}}\,. \end{aligned} \tag{6.11}$$

$\mathbf{\Lambda}$ is called deformation or stretching tensor and its proper directions are the maximum directions of stretching.$\mathbf{\Omega}$ represents the local rotation, which is a skew-symmetric matrix, called vorticity or spin matrix. The rotation matrix is of the form

$$\mathbf{\Omega} = \frac{1}{2}\begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \tag{6.12}$$

with the vector field $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)^T$, which is the curl or rotation of the velocity, i.e. $\boldsymbol{\omega} = \nabla \times \boldsymbol{v}$. The vector field $\boldsymbol{\omega}$ is called *vorticity* and is readily interpreted as twice the local angular velocity in the fluid. Note also that $\mathrm{div}(\boldsymbol{\omega}) = \mathrm{div}(\mathrm{rot}(\boldsymbol{v})) = 0$.

Since vorticity is a vector field, all the methods presented in the previous Subsections can be employed for visualizing the vorticity. Prominent examples are vortex lines [488] and vortex tubes [39], which are the stream lines and tubes of the vorticity. However, the vorticity is only one component of the derivative of the velocity, which is itself the derivative of the flux. That is, it seems rather difficult to gather information about the flow from vortex lines or tubes and even experienced researchers can be surprisingly misled by vortex lines.

6.5.3 Particle Tracing

In order to compute a particle path, Equation 6.7 used to be solved on uniform meshes. In 1989 Buning [99] presented the stencil walk algorithm, which makes it possible to perform particle tracing on curvilinear grids. Sadarjoen et al. [590] compared physical space (P-space) and computational space (C-space) methods for curvilinear grids. The errors introduced in the C-space method by transforming the velocity by means of possibly inaccurate Jacobians favorite the P-space approach. Here however, point location usually computed by the stencil walk algorithm, which also needs the Jacobians, is a slight drawback. Kenwright and Lane [385] suggested a tetrahedral decomposition of the hexahedral cells for speeding up the point location but introducing an additional cell search on the tetrahedral cells. Frühauf [234] and Ueng et al. [718] investigated fast algorithms for computing particle traces in steady flows on unstructured grids.

In the following sections we are going to describe how different properties of a flow field can be visualized by appropriate particle tracing techniques. The visualized flow characteristics are *orbits*, *speed*, and *local rotation*. In addition, extra scalar values like temperature, density, or pressure of the fluid flow can be visualized, for instance

by color coding. The geometric primitives used for the particle visualization are *lines*, *tubes*, *balls*, *ribbons*, and *tetrahedra*.

Path. *Lines* are the native approach for visualizing path, stream, or streak lines. The path is numerically computed and then a broken line is drawn to depict the result. Furthermore, a scalar variable, e.g. temperature, density, pressure, or the absolute value of the velocity can by displayed by color coding.

If the velocity field $\boldsymbol{v}(\boldsymbol{x}, t)$ of the flow is given in an analytical form, integration algorithms of high order are preferable like extrapolation methods or high order Runge-Kutta schemes. However, in real applications vector fields arise that are defined on discrete grids, since these velocity fields are given by numerical simulation or by measurement. For such rough vector fields higher order algorithms are useless. As a result of careful analysis of numerical efficiency and accuracy of different integration methods on discrete data, it can be shown that an adaptive RK3(2) scheme is accurate enough in relation to the interpolation error and significantly more efficient than higher order integration algorithms [688]. Furthermore, so-called implicit integration methods can be applied to handle stiff systems of ordinary differential equations. These algorithms are slower than explicit methods but in stiff data sets, where explicit methods fail, they can create proper trajectories.

Tubes are lines with spatial extent. In principle, a line is computed and each point of the line is replaced by a circle lying in the plane perpendicular to the current velocity direction. A benefit of tubes is that they are polygonal objects and therefore support the spatial perception by both hidden surface removal and shading. Moreover, tubes have the advantage that the radius of the tube can be varied in order to visualize an second scalar variable.

Another possibility to create tube-like stream objects is to choose a circle as source element and to trace all its points. Hereby, the convergence and divergence in the flow can be easily recognized. On the other hand, it often happens that the tube is getting very thin or fat, which means that the tube nearly either disappears or occludes all the scene.

Speed. If *balls* are used, the speed of the flow can be perceived in a natural way from the distance between successive balls. Since size and color of balls are utilized as before, balls can visualize an additional scalar value compared to tubes. However, in case of adaptive integration methods, the distance between successive particles does not depict the absolute value of the velocity but the size of the current integration step.

In this case, we can just visualize the same properties of the flow applying either tubes or balls. Nevertheless, balls have the advantage that they give valuable insight into the adaptive integration process.

Local Rotation. In addition to the visualization of the orbits, speed, and some extra scalar variables, we now concentrate on the visualization of the local rotation in the fluid flow (compare Section 6.5.2).

In fact, *bands* or *ribbons* reveal the local rotation in an intuitive way. There are two different ways to generate a band that visualizes the local rotation. The first method

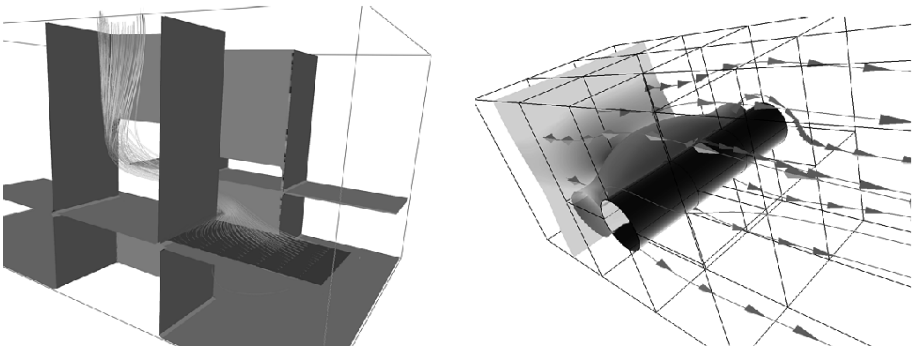

Figure 6.23. On the left hand side, the air flow in a clean air laboratory is investigated. The ventilation is improved step by step by displacing the workbenches, hoods, and gasper fans, starting a new numerical simulation, and visualizing the air flow again. On the right, the time-dependent flow around a cylinder is visualized by means of streak tetrahedra. The curvilinear multi-block data sets consists of 24 blocks. The pressure is visualized by the color of the tetrahedra. In addition, an iso-surface of the pressure has been computed in the right image. This time, the difference between successive tetrahedra does not visualize the speed of the fluid, since an adaptive Runge-Kutta method (RK3(2)) has been used for calculating the streak lines. Hence, the tetrahedra near the cylinder reveal the small integration step sizes that have been used in this region.

is to compute two particle traces that are close together and to fill the space between the lines with a shaded polygon [51]. In order to obtain a ribbon of constant width, the distance between the particle lines has to be normalized after each integration step. Moreover, since this technique is just an approximation to the local rotation, the bands have to start close enough to achieve proper results. Another drawback of this method is the fact that two lines have to be computed. Hence, it is a good idea to calculate the local rotation directly. One can use the vorticity matrix $\boldsymbol{\Omega}$ (see Section 6.5.2) to generate the band. Assume, just for simplicity, that we were using Euler's method for integrating the orbit:

$$\boldsymbol{x}_n = \boldsymbol{x}_{n-1} + \tau \cdot \boldsymbol{v}(\boldsymbol{x}_{n-1}) \quad . \tag{6.13}$$

Furthermore, let the band vector $\boldsymbol{b}$, which determines the direction of the ribbon, be a small vector perpendicular to the tangent vector of the orbit. Then, $\boldsymbol{b}$ is transformed according to the expression

$$\boldsymbol{b}_n = \boldsymbol{b}_{n-1} + \tau \cdot D_{\boldsymbol{x}}\boldsymbol{v}(\boldsymbol{x}_{n-1}) \cdot \boldsymbol{b}_{n-1} + O(\|\boldsymbol{b}_{n-1}\|^2) \tag{6.14}$$

if Euler's integration scheme is used. Neglecting higher order terms, expansion, and shear, we obtain the following equation:

$$\boldsymbol{b}_n = \boldsymbol{b}_{n-1} + \tau \cdot \boldsymbol{\Omega}(\boldsymbol{x}_{n-1}) \cdot \boldsymbol{b}_{n-1} \quad . \tag{6.15}$$

Figure 6.24. The turbulent jet stream in the Pegase data set is visualized by means of streak balls (left) and streak bands (right). The absolute value of the velocity is revealed by color coding and in the left image by the size of balls as well.

Because of approximation and rounding errors due to using finite differences for calculating the Jacobian $D_x v$ and due to neglecting terms of higher order, it is safer to project the new band vector b after each integration step onto the plane perpendicular to the tangent vector of the orbit. This results in ribbons that spin longitudinally in a swirling flow.

Another variable that can be visualized by bands is the torsion of the current path, stream, or streak line [386]. However, notice that the local rotation of the flow is in general different from the torsion of the stream line through the current position.

Further geometric primitives for visualization purposes are tetrahedra as shown in Figure 6.23. They have the advantages from both balls and bands (see Figure 6.24).