Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization (2003)
G.-P. Bonneau, S. Hahmann, C. D. Hansen (Editors)

Feature Flow Fields

H. Theisel† and H.-P. Seidel‡

Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany

Abstract

Feature tracking algorithms for instationary vector fields are usually based on a correspondence analysis of the features at different time steps. This paper introduces a method for feature tracking which is based on the integration of stream lines of a certain vector field called feature flow field. We analyze for which features the method of feature flow fields can be applied, we show how events in the flow can be detected using feature flow fields, and we show how to construct the feature flow fields for particular classes of features. Finally, we apply the technique to track critical points in a 2D instationary vector field.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Visualization, flow visualization, feature extraction and tracking

1. Introduction

Flow visualization is one of the most important subfields of scientific visualization. From its very beginning, flow visualization had to face the problem of treating large and complex data. A variety of techniques have been developed for computing expressive visual representations for 2D or 3D flow fields.

Among the flow visualization techniques, feature extraction techniques are a promising approach because of their potential capability to dramatically reduce the complexity of the flow field. An intensive research on feature flow fields has been done in the flow visualization community in recent years.

Features of flow fields represent the interesting objects or structures in a flow. For stationary flow fields, a number of techniques exist to define and extract features from the flow data [18].

To treat features of instationary (time varying) flow fields, the additional problem appears that features might change their location, their shape, and other characteristics. In addition, certain events of the features may occur. The analysis of this dynamic behavior of features is called feature tracking.

† theisel@mpi-sb.mpg.de
‡ hpseidel@mpi-sb.mpg.de

Most existing techniques for feature tracking are based on the following steps [26]:

1. Extract features of the flow at a certain number of time steps.
2. Find the corresponding features in consecutive time steps.
3. Detect events in the features.
4. Visualize the evolution of the features.

To find the corresponding features at different time steps, two general approaches exist [18]: region correspondence and attribute correspondence. For region correspondence, the correspondence of regions of interest is detected by searching minima or maxima of certain measures. These measures can be the distance of the features in two consecutive time steps, or an affine transformation matrix [12]. In addition, the overlapping of features in consecutive time steps can be used [30]. For attribute correspondence, certain attributes of the features (like position, size, volume) is observed over time. The correspondence criteria are usually described as a combination of location and size of the features [26], or it may be described by a predictor and verification approach [19, 20].

For the step of event detection, the following events are considered [26, 20]:

- continuation
- birth (creation) and death (dissipation)
- entry and exit
- split (bifurcation) and merge (amalgamation)

To detect these events, features in consecutive time steps are checked for significant changes in their characteristics (for instance a changing number of features, or a predicted negative size of the features). In these cases the events are deduced from these changes.

In recent years a number of alternative feature tracking approaches have been proposed for particular features. These methods are mainly related to an isosurface extraction in 4-dimensional space. [36] does so for features in scalar fields while [2] uses this approach for tracking vortex core lines over time (and also in scale space).

The main idea of this paper is to introduce another alternative approach to the feature tracking techniques mentioned above. This approach is based on the idea to represent the dynamic behavior of the features not as a higher-dimensional isosurface but as the stream lines (or stream surfaces or stream objects) of a higher dimensional vector field. The numerical integration of stream lines is well-established in flow visualization. Because of this, we want to make use of these techniques to track features in instationary flow fields.

The rest of the paper is organized as follows: section 2 describes the main idea of the paper in detail. Section 3 demonstrates the application for a particular feature tracking problem: the tracking of critical points in instationary vector fields. Section 4 answers the question for which features the proposed technique is applicable.

2. Description of the main idea

For the sake of simplicity, we start describing the technique for instationary vector fields in 2D. Later we show that the technique can be transformed to 3D vector fields in a straightforward way.

Given is a smooth instationary 2D vector field

$$\mathbf{v}(x,y,t) = \begin{pmatrix} u(x,y,t) \\ v(x,y,t) \end{pmatrix}. \qquad (1)$$

This means, we do not only consider $\mathbf{v}$ at discrete time steps t_i but for a continuous time domain. This can be achieved by interpolating the flow data not only in x- and y-direction but also in t-direction. Then the idea is to construct a new 3D vector field

$$\mathbf{f}(x,y,t) = \begin{pmatrix} f(x,y,t) \\ g(x,y,t) \\ h(x,y,t) \end{pmatrix}. \qquad (2)$$

in such a way that the dynamic behavior of the features of $\mathbf{v}$ is described by the stream lines of $\mathbf{f}$. In fact, tracking features of $\mathbf{v}$ over time is now carried out by tracing stream lines of $\mathbf{f}$. Since $\mathbf{f}$ describes the flow of certain features of $\mathbf{v}$ over time, we call $\mathbf{f}$ the *feature flow field* of $\mathbf{v}$. Figure 1 gives an illustration.

Note that the stream lines of a feature flow field do not necessarily have to go "forward" in time. Instead, both

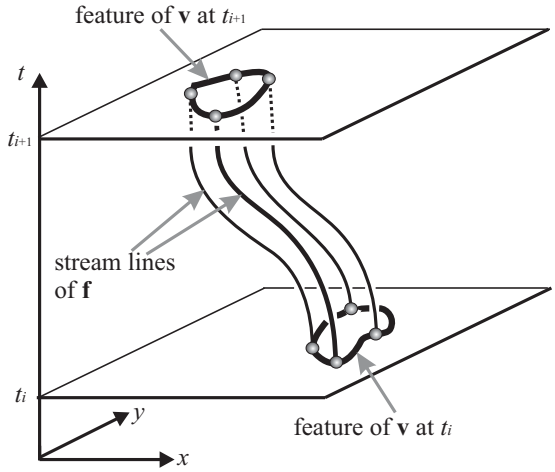

Figure 1: *Feature tracking using feature flow fields. The dynamic behavior of a feature of* $\mathbf{v}$ *at a certain time* t_i *is tracked by tracing the stream lines of* $\mathbf{f}$ *from the feature. The features at a certain time* t_{i+1} *can be observed by intersecting these stream lines with the time plane* $t = t_{i+1}$.

"backward" and "forward" flows of the features in time are possible, even for the same stream line of $\mathbf{f}$. Figure 2 illustrates this.

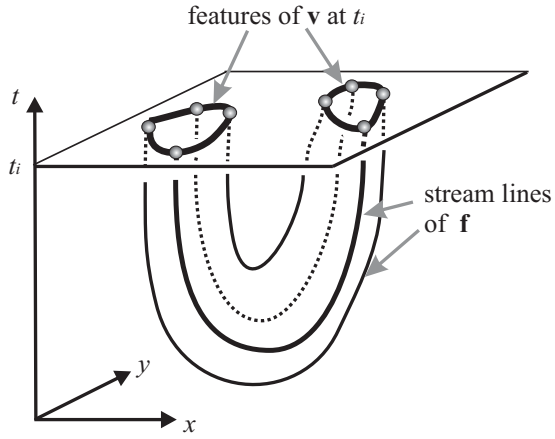

Figure 2: *The stream lines of* $\mathbf{f}$ *may go "forward" and "backward" in time.*

The examples in figures 1 and 2 show that the dynamic behavior of a feature is not necessarily described by one stream line but by a number of stream lines originating from all points which belong to a feature at a certain time. Depending on the dimensionality of the feature, the feature tracking corresponds to stream line, stream surface or stream object integration.

The stream lines of $\mathbf{f}$ can also be used to detect events

of the features. An exit event occurs if all stream lines of **f** which describe the feature leave the $x-y$-domain of the vector field. A birth event occurs at a time t_b when the feature at the time t_b is only described by one stream line of **f**, and this stream line touches the plane $t = t_b$ "from above" (i.e., the stream line in a neighborhood of the touching point is in the half-space $t \geq t_b$). Figure 3 shows an example. A split occurs at a time t_s if one of the stream lines of **f** describing the feature touches the plane $t = t_s$ "from above". Figure 3 illustrates an example. The conditions for the reverse events (entry, merge, dead) can be formulated in a similar way.

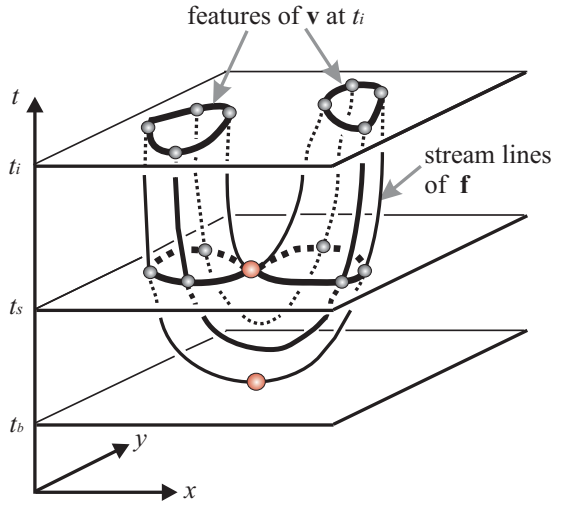

Figure 3: *Two events in a flow. Shown are two times t_b and t_s in which events take place. At the time t_b, a new feature is born, at the time t_s it splits into two features.*

Up to now we introduced the concept of feature flow fields in a rather informal way. To make it applicable, we have to treat two problems:

- For which features, feature flow fields can be used?
- How can the feature flow field **f** be obtained for these features?

Before dealing with these problems in section 4, the following section describes the usage of feature flow fields for tracking a particular feature: critical points in instationary 2D flows.

3. Tracking critical points in instationary 2D vector fields

The topology of vector fields as a visualization tool has been introduced in [7,8]. Later it was extended to to higher order topologies [27], to multilevel topologies [4], and to 3D topologies [5,6]. Topological issues play a role in simplification [3], compression [15], and construction [32] of vector fields.

The topology of a vector field essentially consists of critical points and separatrices. Critical points are points where the vector of the field vanishes; separatrices are particular stream lines separating areas of different flow behavior. In the following we restrict ourselves to first-order critical points, i.e. critical points with a non-zero Jacobian.

For instationary vector fields, the locations of critical points over time has to be tracked since critical points might change their location. In addition, the following events may occur which change the characteristics of the critical points [34]:

- Fold bifurcation: two critical points collapse and disappear. One of them is a saddle while the other is either a source, a sink, or a center.
- Creation of two critical points (one saddle and one source/sink/center) out of an area without critical points before. This event is the reverse of a fold bifurcation.
- Hopf bifurcation: switch from source to sink or reversely via the unstable state of a center.

To track the location of critical points over time (including fold bifurcations), [33] and [34] apply a piecewise linear interpolation of the vector data. This way the faces of every cell in the (x,y,t)-domain have at most one critical point which can easily be found by solving a system of linear equations. Connecting these points on the cell faces gives the path of the critical points in the (x,y,t)-domain. Since a face of a cell can have at most one critical point, fold bifurcations (or their inverse events) can occur only at cell boundaries.

Now we want to trace critical points using the idea of feature flow fields. Let $\mathbf{v}(x,y,t)$ be an instationary 2D vector field with a critical point. We construct $\mathbf{f}(x,y,t)$ by demanding that the vectors of **f** point in the direction in which the vectors of **v** (both direction and magnitude) remain unchanged if we assume a first order approximation of **v** around the point of consideration. The direction of maximal change of the u-component of **v** is given by the gradient $\mathrm{grad}(u)$. In the plane perpendicular to $\mathrm{grad}(u)$, the u-component remains constant in a first order approximation of **v**. A similar statement holds for the v-component of **v**: it is constant in the plane perpendicular to $\mathrm{grad}(v)$. Thus, the only direction in which both u- and v-component of **v** remain constant is the intersection of the planes perpendicular to $\mathrm{grad}(u)$ and $\mathrm{grad}(v)$. This gives

$$\mathbf{f}(x,y,t) = \mathrm{grad}(u) \times \mathrm{grad}(u) = \begin{pmatrix} \det(\mathbf{v}_y, \mathbf{v}_t) \\ \det(\mathbf{v}_t, \mathbf{v}_x) \\ \det(\mathbf{v}_x, \mathbf{v}_y) \end{pmatrix}. \quad (3)$$

Figure 4 gives an illustration. This definition of the feature flow field **f** ensures that all points (x,y,t) on a stream line of **f** have the same value for $\mathbf{v}(x,y,t)$. Figure 5 illustrates this. Obviously, the path of a critical point of **v** can now be described as the stream line of **f** starting in a critical point of **v**.

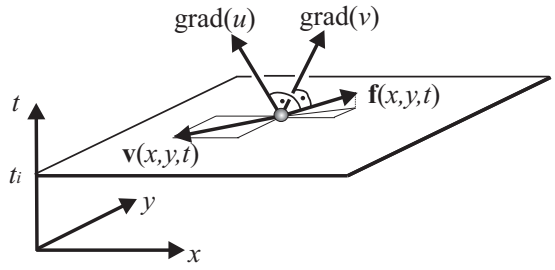

Figure 4: *Definition of the feature flow field* **f** *for critical point tracking.*

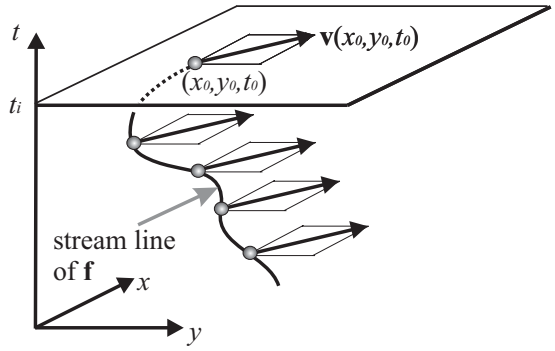

Figure 5: *All points* (x,y,t) *on a stream line of* **f** *(defined by (3)) have the same value for* $\mathbf{v}(x,y,t)$.

Now we can formulate the following algorithm to track critical points:

1. Find an appropriate number of critical points (seed points) in the (x,y,t) domain.
2. Compute the stream lines of **f** from the seed points by both backward and forward integration.
3. To obtain a critical point at a certain time t_i: intersect all stream lines of step 2 with the plane $t = t_i$.

This algorithm needs some comments:

1. The problem of finding an appropriate (i.e. minimal and complete) set of stream lines which cover all critical points is non-trivial and has similarities to the problem of finding seed points in volume data sets for isosurface extraction [29, 28, 14]. Here we used a rather straightforward approach (which cannot guarantee to find a complete set of seed points): we extracted all critical points of **v** at certain time steps $t_0, ..., t_n$. From this set of critical points we start to construct the stream lines of **f**, but before constructing a particular stream line we check if the critical point is already covered by a stream line constructed before.
2. The stream lines usually have to be integrated numerically. We used a fourth order Runge-Kutta method. If a higher precision is desired (for instance in turbulent flows), higher order numerical integration techniques can be considered.
3. Since the result of the numerical integration in step 2 is a piecewise linear approximation of the stream line (i.e. a polygon), its intersection with the plane $t = t_i$ can easily be obtained.

Now we show the application of the algorithm described above to a real flow data set. Figure 6 shows the visualization of a 2D flow in a Bay area of the Baltic Sea near Greifswald, Germany (Greifswalder Bodden). This data set was obtained by a numerical simulation on a regular 115×103 grid at 25 time steps. Since the water can be considered as incompressible, the resulting instationary vector field holds $\mathrm{div}(\mathbf{v}) \equiv 0$. This gives that the only critical points we can expect are saddles and centers. The vector field was obtained by applying a trilinear interpolation of the vector values in x-, y- and t-direction. Figure 6 shows the vector field at the first and the last time step t_0 and t_{24} using the IDraw technique [21]. The figure shows that number and location of the critical points in the flow change over time.

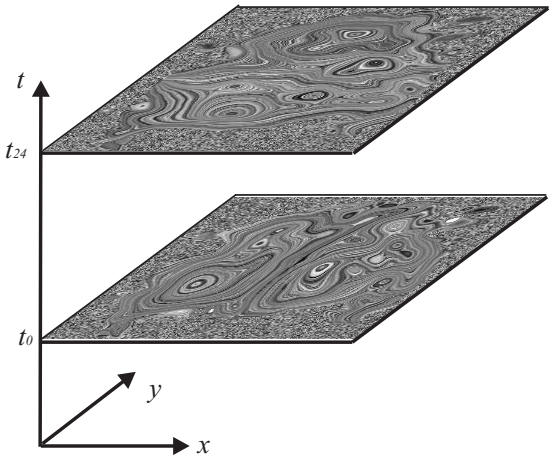

Figure 6: *Test data set at two different time steps* t_0 *and* t_{24}. *Data set provided by Department of Mathematics, University of Rostock.*

Figure 7 shows the extraction of all critical points at the time steps t_0 and t_{24} as well as a collection of stream lines of **v** at these times. Saddle points are marked with small blue spheres, at centers a red sphere was located.

Figure 8 shows again the critical points at the time steps t_0 and t_{24}, but now together with the stream lines of the feature flow field **f** (defined by (3)) which start in these critical points. To emphasize the location of the stream lines in 3D, we visualized them as cylindrical closed surfaces with a small radius around the stream lines. In addition we used an alternating color coding of the stream lines for different time intervals. For the particular data set it turned out that the critical points at the times t_0 and t_{24} are sufficient to build a seed

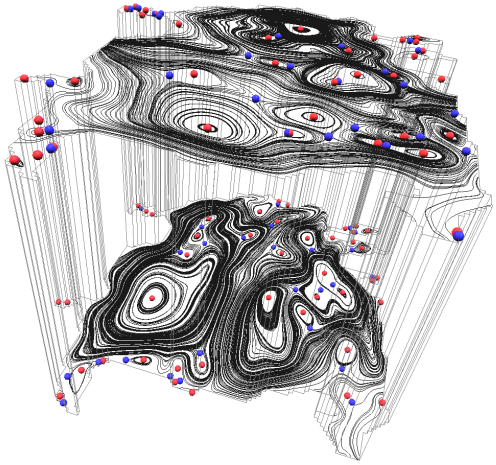

Figure 7: *Critical points and stream lines of* **v** *at the time steps* t_0 *and* t_{24}.

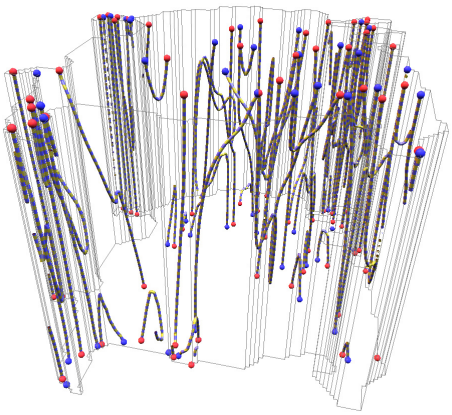

Figure 8: *Critical points and stream lines of* **f** *at the time steps* t_0 *and* t_{24}.

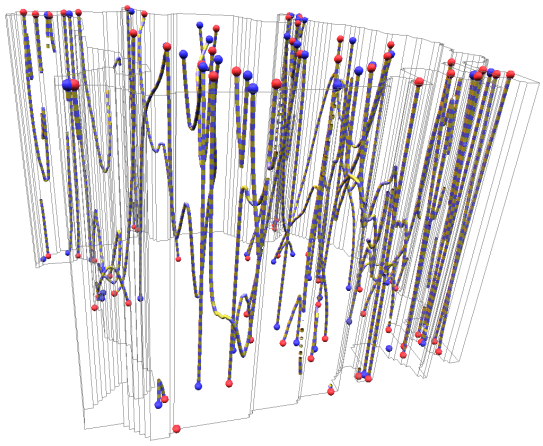

Figure 9: *Critical points and stream lines of* **f** *at the time steps* t_0 *and* t_{24}.

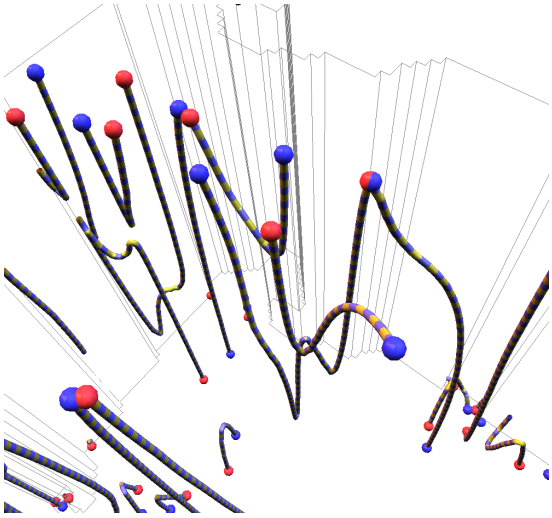

Figure 10: *Zoom into the test data set.*

set. This means, we did not find any critical point at a certain time between t_0 and t_{24} which was not covered by one of the already constructed stream lines of **f**.

Figure 9 shows the same data set as figure 8 but from another view point.

Figures 8 and 9 show that for rather complex data sets (with a higher number of critical points) the tracking of their location can lead to visual clutter. To analyze particular stream lines of **f**, we zoomed into regions of interest as shown in figures 10 and 11.

Figures 8–11 reveal a number of events in the critical points. We can observe some entry and exit events where the critical points leave the $x-y$-domain. In the figures 8–11, these events correspond to the end points of the stream lines which are not marked with a particular critical point (red or blue sphere) at the times t_0 or t_{24}. Also, we can observe a number of fold bifurcations and their reversed events. A stream line of **f** indicates a fold bifurcation a the time t_i if the stream line touches the plane $t = t_i$ "from beyond" (i.e., in a neighborhood of the touching point the stream line is completely located in the half-plane $t \leq t_i$). The event of creating a new pair of critical points at a time $t = t_i$ is indicated by a stream line of **f** which touches the plane $t = t_i$ "from above".

In figures 8–11 we also can see that two different critical points at a time level t_i can be represented by the same stream line of **f**. This means that at a certain other time (before or after), these critical points are going to collapse.

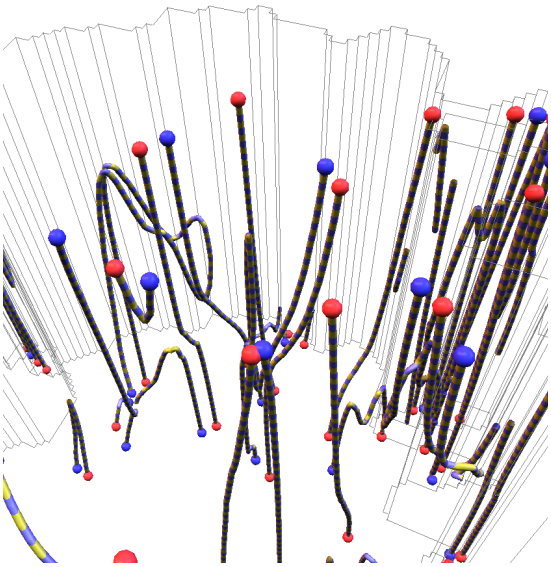

Figure 11: *Zoom into the test data set.*

As mentioned before, the vector field **v** considered in figures 6 – 11 was obtained by applying a piecewise trilinear interpolation of the vector data in x-, y- and t-direction, producing a C^0 continuous vector field. Then the feature flow field **f** defined by (3) describes a C^{-1} continuous biquadratic vector field. Thus the stream lines of **f** are G^0 continuous which explains the appearance of sharp corners in the stream lines in figures 8 – 11.

While feature flow fields are an appropriate tool for tracking the location of critical points (including the detection of fold bifurcations and its reverse), the concept cannot be used directly for tracking the classification of them (including Hopf bifurcations). However, at a point (x,y,t) on the the stream line of **f** with $h(x,y,t) = 0$, the traced critical point switches from saddle to either source, sink or center, or reversely. Since the path of the critical point is known by **f**, it is possible to trace the Jacobian of **v** along the stream line of **f** to detect for instance Hopf bifurcations.

4. Feature flow fields for general features

This section discusses for which features the concept of feature flow fields is applicable. A variety of different features has been introduced in the literature (see [18] for an overview). Generally, feature extraction of vector fields can be classified into three approaches [18]: based on image processing, on a topological analysis, and on physical characteristics. Common features for vector fields are critical points, separatrices, vortices, shock waves, and others.

To establish the relation between features and feature flow fields, we want to classify features into two groups: local and global features. Local features are obtained by locally analyzing certain points in the flow (for instance grid points). This means, local feature are characterized by the fact that for every point in the flow the decision if the point belongs to a feature can be done by locally analyzing the flow (and probably its derivatives) in this point. On the other hand, global features can only be obtained by a global analysis of the vector field.

The concept of feature flow fields is based on the fact that the direction of the movement of the feature points is computed locally as the vector of **f**. Hence, feature flow fields are only applicable for local features.

Depending on the dimensionality of the features, different characteristics in terms of **f** are possible. If the feature is a zero-dimensional point set (i.e. a single point), the feature tracking using feature flow fields turns out to be a stream line integration in **f**. For one-dimensional features (i.e. lines), the feature extraction in **f** is a stream surface integration [9]. Two-dimensional features in **v** correspond to a tracing of flow volumes [16].

Feature extraction techniques which are based on image processing usually map the flow information of a number of time steps into images and apply techniques from computer vision there. This class of techniques is not the target of feature flow fields. The other two classes, topological analysis and physical characteristics, are treated in the following. For physical characteristics, we especially focus on the extraction of vortices.

4.1. Topological features

The applicability of feature flow fields for tracking critical points for 2D instationary vector fields has already been shown in section 3. The concepts introduced there can be extended to 3D vector fields in the following way:

Given an instationary 3D vector field

$$\mathbf{v}(x,y,z,t) = \begin{pmatrix} u(x,y,z,t) \\ v(x,y,z,t) \\ w(x,y,z,t) \end{pmatrix}, \tag{4}$$

the feature flow field **f** is a 4-dimensional vector field which points into the direction in which the vector of **v** remains constant (assuming a local first order approximation of **v** around the considered point). Thus, **f** is determined by

$$\mathbf{f} \perp \mathrm{grad}(u) \;,\; \mathbf{f} \perp \mathrm{grad}(v) \;,\; \mathbf{f} \perp \mathrm{grad}(w) \;, \tag{5}$$

which gives

$$\mathbf{f}(x,y,z,t) = \begin{pmatrix} \det(\mathbf{v}_y, \mathbf{v}_z, \mathbf{v}_t) \\ \det(\mathbf{v}_z, \mathbf{v}_t, \mathbf{v}_x) \\ \det(\mathbf{v}_t, \mathbf{v}_x, \mathbf{v}_y) \\ \det(\mathbf{v}_x, \mathbf{v}_y, \mathbf{v}_z) \end{pmatrix}. \tag{6}$$

This way, tracking the critical points in **v** corresponds to a

4-dimensional stream line integration of **f**. To obtain the particular critical points at a certain time $t = t_i$, these stream lines have to be intersected with the hyperplane $t = t_i$. Similar to the 2D case, the computation of these intersections is straightforward since the numerical integration of stream lines yields a piecewise linear approximation.

The other aspect of the topology of vector fields, separatrices, cannot be treated by using feature flow fields, since separatrices are a global feature of the vector field.

4.2. Vortices

In recent years, an intensive research has been done in defining, extracting, and tracking vortices (or vortex core lines) of vector fields. Some of these definitions of vortices fall into the class of local features (and can therefore be tackled using feature flow fields), others are global features.

Given a 3D vector field $\mathbf{v}(x,y,z,t)$, one way of defining vortices is to consider a scalar field $s(x,y,z,t)$ and define a vortex as consisting of all points (x,y,z,t) where s is larger (or smaller) than a certain threshold. The scalar field s may be derived from **v**, or it might be given in addition to **v**. The following choices of s have been proposed:

- the magnitude of the vorticity [35]: $s = \|\mathrm{curl}(\mathbf{v})\|$,
- the projection of the vorticity onto the velocity vector [13]: $s = \mathrm{curl}(\mathbf{v}) \cdot \mathbf{v}$,
- the pressure of the flow [22],
- one of the eigenvalues of the matrix $S^2 + \Omega^2$ with $S = \frac{1}{2}(\mathbf{v} + \mathbf{v}^T)$ and $\Omega = \frac{1}{2}(\mathbf{v} - \mathbf{v}^T)$ [10].

To find the feature flow fields for **f** for one of these cases, **f** has to fulfill $\mathbf{f} \perp \mathrm{grad}(s)$. Since this underdefines **f**, we can choose three scalar fields s_1, s_2, s_3 to fully define **f** by demanding

$$\mathbf{f} \perp \mathrm{grad}(s_1) \ , \ \mathbf{f} \perp \mathrm{grad}(s_2) \ , \ \mathbf{f} \perp \mathrm{grad}(s_3). \qquad (7)$$

The additional scalar fields might be some of the abovementioned as well, or they might be elementary values like the magnitude of the flow (described by one scalar) or the flow direction (described by two scalars). Then the corresponding feature flow field can be written as

$$\mathbf{f}(x,y,z,t) = \begin{pmatrix} \det(M_1) \\ \det(M_2) \\ \det(M_3) \\ \det(M_4) \end{pmatrix}. \qquad (8)$$

with

$$M_1 = \begin{pmatrix} s_{1y} & s_{1z} & s_{1t} \\ s_{2y} & s_{2z} & s_{2t} \\ s_{3y} & s_{3z} & s_{3t} \end{pmatrix} \ , \ M_2 = \begin{pmatrix} s_{1z} & s_{1t} & s_{1x} \\ s_{2z} & s_{2t} & s_{2x} \\ s_{3z} & s_{3t} & s_{3x} \end{pmatrix}$$

$$M_3 = \begin{pmatrix} s_{1t} & s_{1x} & s_{1y} \\ s_{2t} & s_{2x} & s_{2y} \\ s_{3t} & s_{3x} & s_{3y} \end{pmatrix} \ , \ M_4 = \begin{pmatrix} s_{1x} & s_{1y} & s_{1z} \\ s_{2x} & s_{2y} & s_{2z} \\ s_{3x} & s_{3y} & s_{3z} \end{pmatrix}.$$

In this notation, s_{1x} means for example the x-partial of the scalar field s_1. Note that this feature flow field works for all three scalar fields s_1, s_2, s_3.

Another way of describing vortex core lines is the usage of the concept of parallel vectors [17]. Here, two vector fields $\mathbf{v}_1$ and $\mathbf{v}_2$ are derived from **v**, and a vortex core is assumed at locations where $\mathbf{v}_1$ and $\mathbf{v}_2$ are parallel. Examples for the choice of $\mathbf{v}_1$ and $\mathbf{v}_2$ are

- **v** and $\mathrm{curl}(\mathbf{v})$ [23],
- **v** and certain eigenvectors of the Jacobian [23, 31],
- **v** and the acceleration $\mathbf{a} = \frac{D\mathbf{v}}{dt}$ [24],
- **v** and $\mathbf{b} = \frac{D^2\mathbf{v}}{dt^2}$ [24].

For such a definition of vortices, we define a new vector field **w** as $\mathbf{w} = \mathbf{v}_1 \times \mathbf{v}_2$ and search for **f** in such a way that it points to the direction of constant **w** (assuming a local linear approximation around the considered point). This way we obtain

$$\mathbf{f}(x,y,z,t) = \begin{pmatrix} \det(\mathbf{w}_y, \mathbf{w}_z, \mathbf{w}_t) \\ \det(\mathbf{w}_z, \mathbf{w}_t, \mathbf{w}_x) \\ \det(\mathbf{w}_t, \mathbf{w}_x, \mathbf{w}_y) \\ \det(\mathbf{w}_x, \mathbf{w}_y, \mathbf{w}_z) \end{pmatrix}. \qquad (9)$$

Another way of tracking features which are defined by the parallel vector operator is introduced in [2]. This approach is based on a 4-dimensional isosurface extraction.

Other techniques of constructing vortex core lines are based on the integration of certain stream lines of **v** [5] or of $\mathrm{curl}(\mathbf{v})$ [1]. Since these definitions of vortex core lines have a global character, they cannot be treated using feature flow fields. The same holds for some geometric methods [25] as well as for the method introduced in [11].

5. Conclusion

We have introduced a method for tracking features in instationary vector fields which is based on the analysis of the stream lines of a certain vector field called feature flow field. This method has the following characteristics:

- The method applies well-established techniques of numerical stream line integration instead of the common way of feature tracking.
- The method works independently of the underlying grid for any smooth vector field. In particular, no assumptions about the chosen interpolation of the flow data are necessary.
- The method does not require any correspondence analysis of features in consecutive time steps.
- The method of feature low fields can be used for different features from different applications areas.

However, there remain a large number of open questions for future research.

- Considering feature flow fields for features based on scalar fields (as shown in (8)), it has to be explored which scalar fields s_1, s_2, s_3 can be combined best to build **f**.

- Further relations of **f** and **v** have to be explored. In particular, the role of possible critical points in **f** (where stream lines of **f** may start or end) for the original vector field **v** is unknown.
- Since the technique is proposed for a variety of different features, a particular implementation was only done for critical point tracking. Other local features have to be tested using feature flow fields. In particular, for tracking one- and two-dimensional features by using feature flow fields, a number of problems appear which are not treated yet. These problems are related with the choice of an appropriate number of stream lines to build stream surfaces or stream objects. Here adaptive techniques can be applied which adjust the density of the stream lines. In particular, in areas of a small last component of **f** (i.e., in areas where events can expected), the density of stream lines has to be increased.

References

1. D.C. Banks and B.A. Singer. A predictor-corrector technique for visualizing unsteady flow. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):151–163, 1995.
2. D. Bauer and R. Peikert. Vortex tracking in scale space. In *Data Visualization 2002. Proc. VisSym 02*, pages 233–240, 2002.
3. W. de Leeuw and R. van Liere. Collapsing flow topology using area metrics. In D. Ebert, M. Gross, and B. Hamann, editors, *Proc. IEEE Visualization '99*, pages 149–354, Los Alamitos, 1999.
4. W. de Leeuw and R. van Liere. Visualization of global flow structures using multiple levels of topology. In *Data Visualization 1999. Proc. VisSym 99*, pages 45–52, 1999.
5. A. Globus and C. Levit. A tool for visualizing of three-dimensional vector fields. In *Proc. IEEE Visualization '91*, pages 33–40, Los Alamitos, 1991. IEEE Computer Society Press.
6. H. Hauser and E. Gröller. Thorough insights by enhanced visualization of flow topology. In *9th international symposium on flow visualization*, 2000.
7. J. Helman and L. Hesselink. Representation and display of vector field topology in fluid flow data sets. *IEEE Computer*, 22(8):27–36, August 1989.
8. J. Helman and L. Hesselink. Visualizing vector field topology in fluid flows. *IEEE Computer Graphics and Applications*, 11:36–46, May 1991.
9. J. Hultquist. Constructing stream surfaces in steady 3D vector fields. In *Proc. IEEE Visualization '92*, pages 171–177, Los Alamitos, 1992. IEEE Computer Society Press.
10. J. Jeong and F. Hussain. On the identification of a vortex. *J. Fluid Mechanics*, 285:69–94, 1995.
11. M. Jiang, R. Machiraju, and D. Thompson. A novel approach to vortex core region detection. In *Data Visualization 2002. Proc. VisSym 02*, pages 217–225, 2002.
12. D.S. Kalivas and A.A. Sawchuk. A region matching motion estimation algorithm. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 54(2):275–288, Sept. 1991.
13. Y. Levy, D. Degani, and A. Seginer. Graphical visualization of vortical flows by means of helicity. *AIAA Journal*, 28(8):1347–1352, 1990.
14. Y. Livnat, H.W. Shen, and C. R. Jonson. A Near Optimal Isosurface Extraction Algorithm Using the Span Space. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):73–84, 1996.
15. S.K. Lodha, J.C. Renteria, and K.M. Roskin. Topology preserving compression of 2d vector fields. In T. Ertl, B. Hamann, and A. Varshney, editors, *Proc. IEEE Visualization 2000*, pages 343–350, 2000.
16. N. Max, B. Becker, and R. Crawfis. Flow volumes for interactive vector field visualization. In *Proc. Visualization 93*, pages 19–24, 1993.
17. R. Peikert and M. Roth. The parallel vectors operator - a vector field visualization primitive. In *Proc. Visualization 99*, pages 263–270, 1999.
18. F.H. Post, B. Vrolijk, H. Hauser, R.S. Laramee, and H. Doleisch. Feature extraction and visualisation of flow fields. In *Proc. Eurographics 2002, State of the Art Reports*, pages 69–100, 2002.
19. F. Reinders, F.H. Post, and H.J.W. Spoelder. Attribute-based feature tracking. In *Data Visualization 1999. Proc. VisSym 99*, pages 63–72, 1999.
20. F. Reinders, F.H. Post, and H.J.W. Spoelder. Visualization of time-dependent data using feature tracking and event detection. *The Visual Computer*, 17(1):55–71, 2001.
21. C. Perez Risquet. Visualizing 2D flows: Integrate and Draw. In *Proceedings 9th Eurographics Workshop on Visualization in Scientific Computing*, pages 57–67, 1998.
22. S.K. Robinson. Coherent motions in the turbulent boundary layer. *Ann. Rev. Fluid Mech.*, 23:601–639, 1991.
23. M. Roth and R. Peikert. Flow visualization for turbomachinery design. In *Proc. Visualization 96*, pages 381–384, 1996.
24. M. Roth and R. Peikert. A higher-order method for finding vortex core lines. In D. Ebert, H. Hagen, and H. Rushmeier, editors, *Proc. IEEE Visualization '98*, pages 143–150, Los Alamitos, 1998. IEEE Computer Society Press.
25. I.A. Sadarjoen and F.H. Post. Geometric methods for vortex detection. In *Data Visualization 1999. Proc. VisSym 99*, pages 53–62, 1999.
26. R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *IEEE Computer*, 27(7):20–27, 1994.
27. G. Scheuermann, H. Krüger, M. Menzel, and A. Rockwood. Visualizing non-linear vector field topology. *IEEE Transactions on Visualization and Computer Grapics*, 4(2):109–116, 1998.
28. H.W. Shen. Isosurface extraction in time-varying fields using a temporal hierarchical index tree. In D. Ebert, H. Hagen, and H. Rushmeier, editors, *Proc. IEEE Visualization '98*, pages 159–164, Los Alamitos, 1998.
29. H.W. Shen, C.D. Hansen, Y. Livnat, and C.R. Jonson. Isosurfacing in Span Space with Utmost Efficiency (ISSUE). In R. Yagel and G. M. Nielson, editors, *Proc. IEEE Visualization '96*, pages 287–294, Los Alamitos, 1996.
30. D. Silver and X. Wang. Tracking and visualizing turbulent 3D features. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):129–141, 1997.
31. D. Sujudi and R. Haimes. Identification of swirling flow in 3d vector fields. Technical report, Department of Aeronautics and Astronautics, MIT, 1995. AIAA Paper 95-1715.
32. H. Theisel. Designing 2D vector fields of arbitrary topology. *Computer Graphics Forum (Eurographics 2002)*, 21(3):595–604, 2002.
33. X. Tricoche, G. Scheuermann, and H. Hagen. Topology-based visualization of time-dependent 2D vector fields. In *Data Visualization 2001. Proc. VisSym 01*, pages 117–126, 2001.
34. X. Tricoche, T. Wischgoll, G. Scheuermann, and H. Hagen. Topology tracking for the visualization of time-dependent two-dimensional flows. *Computers & Graphics*, 26:249–257, 2002.
35. J. Villasenor and A. Vincent. An algorithm for space recognition and time tracking of vorticity tubes in turbulence. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 55(1):27–35, Jan. 1992.
36. C. Weigle and D. C. Banks. Extracting iso-valued features in 4-dimensional scalar fields. In *Proc. Symposium on Volume Visualization*, pages 103–110, 1998.