The Visual Computer manuscript No.
(will be inserted by the editor)

Tobias Salzbrunn · Christoph Garth · Gerik Scheuermann · Joerg Meyer

Pathline Predicates and Unsteady Flow Structures

Abstract In most fluid dynamics applications, unsteady flow is a natural phenomenon and steady models are just simplifications of the real situation. Since computing power increases, the number and complexity of unsteady flow simulations grows, too. Besides time-dependent features, scientists and engineers are essentially looking for a description of the overall flow behavior, usually with respect to the requirements of their application domain. We call such a description a flow structure, requiring a framework of definitions for unsteady flow structure. In this paper, we present such a framework based on pathline predicates. Using the common computer science definition, a predicate is a Boolean function, and a pathline predicate is a Boolean function on pathlines that decides if a pathline has a property of interest to the user. We will show that any suitable set of pathline predicates can be interpreted as an unsteady flow structure definition. The visualization of the resulting unsteady flow structure provides a visual description of overall flow behavior with respect to the user's interest. Furthermore, this flow structure serves as a basis for pathline placements tailored to the requirements of the application.

Keywords Unsteady flow visualization · feature detection · computational fluid dynamics (CFD)

T. Salzbrunn, G.Scheuermann
Institut für Informatik
Universität Leipzig
PF 100920
D-04009 Leipzig
E-mail: salzbrunn,scheuermann@informatik.uni-leipzig.de

C. Garth
Technische Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern
E-mail: garth@informatik.uni-kl.de

J. Meyer
University of California, Irvine (UCI)
644E Engineering Tower
Irvine, CA 92697-2625
E-mail: jmeyer@uci.edu

1 Introduction

Unsteady three-dimensional flow is a better model of natural flow phenomena than steady flow in most cases. Since the computing power, especially of PC clusters, is high enough now to compute such complex simulations, unsteady flow data has become quite common and we need good post-processing tools for these datasets. Since the data is massive, interactive access is still difficult, especially for realistic unstructured grids with millions of elements. A meaningful combination of automatic analysis and visualization is needed to provide insight to engineers and scientists. Because we want to base the visualization on the analysis task, we need informations about the kind of knowledge to be extracted from the simulation data. In this paper, we assume that the user has a list of flow features of interest. Usually, this list is based on experience, theory and intention of the simulation. Therefore, a feature-based flow visualization is part of the solution to the analysis task. It is only a part, because, in our view, the user will also be interested in the overall flow in relation to the features. As an example, an engineer constructing a part of a water turbine outflow is usually interested in vortices, among other features. Therefore, a method for detecting and visualizing vortices through regions with large vortical motion is needed. But this is only a part of the story, because the engineer will ask also the question how the vortex influences the overall flow in the turbine. This article introduces the concept of unsteady flow structure to address both local and global aspects. We divide the flow in two parts: The pathlines influenced by the vortex and the remaining pathlines. Looking at the particles moving through the vortex vicinity reveals how this particles behave in comparison to their counterparts. Through this analysis we will also see some circulating behavior. Using again our framework reveals the part of flow showing this behavior and complementing our basic understanding of the flow. We call such a partitioning of the pathlines, like in the previous two examples, an unsteady flow structure.

In general, an unsteady flow structure is a partitioning of all pathlines that reflects the user's analysis criteria with respect to the particle movement. Of course, this means that totally different flow structures can be obtained for the same flow, but this is a direct consequence of varying intentions specified by different users analyzing the same simulation.

This paper defines a framework for defining and computing such flow structures. The framework is based on the notion of pathline predicates. Such a predicate assigns a Boolean value to each pathline depending on whether the respective pathline has a certain property or behavior. In this way, each predicate splits the set of all pathlines into two classes. Combining different predicates based on Boolean algebra allows a finer partitioning of the flow that goes beyond a simple partitioning into two parts. This provides the basis for definitions of a wide variety of flow structures of different complexity. But even for very complex flow structures, the user is provided with an exact meaning for each part of this flow structure, because of the exact definitions of the underlying predicates. However, despite this objectivity, it is up to the user to choose meaningful predicates in order to get a meaningful flow structure that is worth its name.

Flow structures provide a structural view of the behavior of interest. Even though this is valuable information in its own right, users still might want to track the traces of individual particles. Therefore, we will also show how to compute a sparse pathline placement that takes the underlying flow structure into account and provide the user with a dynamic view of the flow. We will see that both views inspire and complement each other.

2 Related Work

Pathlines and streaklines [14] are often used to show particle traces in time-dependent flow fields. In addition, Becker et al. [3] extend flow volumes to unsteady flows as a generalization of the concept of streaklines. Time surfaces as an extension of time lines can be handled by a level-set approach as proposed by Westermann et al. [34].

There are several dense, i.e. comprehensive, visualization techniques for unsteady flow. Forsell and Cohen [5] extend the original LIC algorithm from Cabral and Leedom [4]. Verma et al. [30] present a pseudo-LIC approach on sparsely placed pathline ribbons. A hardware-assisted texture advection technique is proposed by Jobard et al. [9]. An image-based approach (IBFV) is introduced by van Wijk [35]. Jobard et al. [10] present a Lagrangian-Eulerian advection scheme (LEA). Weiskopf et al. [33] propose a spacetime-coherent framework for texture-based visualization (UFAC). Shen and Kao [22] present a new LIC-algorithm (UFLIC), which was recently accelerated and extended to 3D by Liu and Moorhead [15]. Many of the dense visualization techniques work only for 2D unsteady flow, i.e. flow on surfaces. Even if there exists an extension for 3D unsteady flow, the usability is reduced by the occlusion problem inherent to dense visualization techniques. Park et al. [16] address this problem by using multi-dimensional transfer functions. Bauer et al. [2] use special regions of interest to selectively visualize 3D time-dependent vector fields. This paper shares the same spirit, and our approach is described in the next section.

For 2D time-dependent vector fields, there exists a number of contributions to extend the topological concepts from steady flow. Early works come from Tricoche et al. [28, 29] and more recent work from Theisel et al. [27]. Recently, Shi et al. [23] presented an information visualization approach for 3D unsteady vector fields. Nevertheless, a convincing concept of structure in unsteady flow is still missing. This paper is an attempt to fill this gap.

Another way to get useful information out of time-dependent data is feature-based visualization. One of the most challenging tasks in this context is to track the desired features over time. There are some general works on feature tracking (van Valsum et al. [31], Samataney et al. [21], Reinders et al. [19], and Silver and Wang [24]) and several works on special features and their tracking, e.g. singularities (Garth et al. [6]), closed streamlines (Wischgoll et al. [36]), vortices (Bauer and Peikert [1]) or general line type features, isosurfaces or volumes (Theisel and Seidel [26], Weigle and Banks [32], and Ji et al. [8]).

The authors have recently published a work on streamline predicates [20] that can be considered as the steady counterpart to the work presented here. In the following section, it will become clear that there are substantial differences. The main handicap to a direct application of the same concepts is the fact that unsteady flow is usually given only for a finite timespan. This prevents many pathlines from being complete in the sense of connecting an inlet to an outlet of the domain via a curved line. It is also apparent that different predicates are needed and that the visualization makes substantial use of animation in the unsteady case.

3 Unsteady Flow Features

Feature-based visualization aims at a higher level of abstraction by extracting "phenomena, structures or objects in a data set, that are of interest for a certain research or engineering problem" [18]. Typical examples in unsteady flows are shockwaves, boundary layers, recirculation zones, attachment lines, separation lines, separation bubbles, vortices, shear flow regions and vortex bursts. In each application, in each data set, and for each researcher, a different feature definition may be appropriate. Therefore, the first important task is to find an appropriate definition of the feature that permits the development of a corresponding detection algorithm. Additionally, for time-dependent features, one has to ensure a consistent tracking over time. In a last step, the extracted feature may be simplified and described quantitatively in order to be easily visualized.

Features can be seen as a characterization of a set of points in the spatial domain. With respect to unsteady flow features, the temporal domain will also be taken into account. Based on information from a local neighborhood, a subregion or the whole field, each point gets an attribute assigned stating whether the specific feature exists or not at this point. Hence a feature definition can be formulated as a point predicate, i.e. a function that maps all points of the domain to a Boolean value:

$$\Pi :\ D \times I \rightarrow \{\ TRUE, FALSE\ \}$$

with the characteristic set

$$C_\Pi\ =\ \{\ (x,t) \in D \times I \mid \Pi(x,t) = TRUE\ \},$$

where $D \in \mathbf{R}^3$ denotes the spatial domain, and $I \subset \mathbf{R}$ denotes the temporal domain.

4 Pathline Predicates

We are concerned with a continuous unsteady velocity field

$$v : D\ \times I \mapsto \mathbf{R}^3$$

on a bounded domain $D \subset \mathbf{R}^3$ over a time span $I = [t_0, t_n]$ and the property that there exists a positive constant $K > 0, K \in \mathbf{R}$ such that

$$\forall x, y \in D \text{ and } \forall r, s \in I, ||v(x,r) - v(y,s)|| \leq K||(x,r) - (y,t)||$$

(Lipschitz property). We note that this property is needed for the existence and uniqueness of the following pathlines.
For any position $a \in D$ and any time $\tau \in I$, we define a pathline $p_{a,\tau}$ by

$$\begin{aligned} p_{a,\tau} :\ I_{a,\tau} &\rightarrow D \\ t &\mapsto p_{a,\tau}(t) \\ p_{a,\tau}(\tau) &= a \\ \frac{\partial p_{a,\tau}}{\partial t}(t) &= v(p_{a,\tau}(t), t), \end{aligned}$$

where v denotes the velocity field. Here $I_{a,\tau} \subset I$ is the maximal lifespan of the particle in D during I. Since we are interested in different particles, we call two pathlines $p_{a,\tau},\ p_{b,\sigma}$ equivalent if they describe the life of the same particle, i.e. we have the equivalence relation $\sim$:

$$\begin{aligned} p_{a,\tau} \sim p_{b,\sigma} \ \Leftrightarrow\ & p_{a,\tau}(\sigma) = b \\ & (\Leftrightarrow\ p_{b,\sigma}(\tau) = a\ ,\ \textit{due to uniqueness theorem}). \end{aligned}$$

Hence, we can express the set of all different particles as equivalence classes of pathlines with respect to the previous equivalence relation $\sim$ as

$$\mathscr{P}_v\ :=\ \{\ p_{a,\tau} \mid p_{a,\tau} \textit{ is pathline of } v\ \}/\sim.$$

Since equivalence classes are not intuitive, we divide the set $\mathscr{P}_v$ in four distinct classes. If we assume that no particle is created or destroyed inside D, a particle

a) can be present in the interior $\overset{\circ}{D}$ at t_0 and leave D at a time $\tau' \in (t_0, t_n]$,
b) can be present in the interior $\overset{\circ}{D}$ at t_0 and be present in the interior $\overset{\circ}{D}$ at t_n,
c) enter D at a time $\tau \in [t_0, t_n]$ and leave D at a time $\tau' \in [t_0, t_n)$ with $\tau' > \tau$, or
d) enter D at a time $\tau \in [t_0, t_n]$ and be present in the interior $\overset{\circ}{D}$ at t_n (see Figure 1).

Formally, we have

$$\mathscr{P}_v\ =\ \mathscr{P}_v{}^b\ \cup\ \mathscr{P}_v{}^p\ \cup\ \mathscr{P}_v{}^c\ \cup\ \mathscr{P}_v{}^e$$

(b = begin, p = permanent, c = complete, and e = enter) with

$$\begin{aligned} \mathscr{P}_v{}^b\ &=\ \{\ p_{a,\tau} \mid p_{a,\tau}(t_0) \in \overset{\circ}{D},\ I_{a,\tau} = [t_0, \tau'] \\ &\qquad \textit{and } p_{a,\tau}(\tau') \in \partial D))\ \} \\ \mathscr{P}_v{}^p\ &=\ \{\ p_{a,\tau} \mid p_{a,\tau}(t_0) \in \overset{\circ}{D},\ I_{a,\tau} = [t_0, t_n] \\ &\qquad \textit{and } p_{a,\tau}(t_n) \in \overset{\circ}{D}\ \} \\ \mathscr{P}_v{}^c\ &=\ \{\ p_{a,\tau} \mid a \in \partial D,\ \tau \in [t_0, t_n],\ I_{a,\tau} = [\tau, \tau'] \\ &\qquad \textit{and } (\tau' < t_n \textit{ or } (\tau' = t_n \textit{ and } p_{a,\tau}(\tau') \in \partial D))\ \} \\ \mathscr{P}_v{}^e\ &=\ \{\ p_{a,\tau} \mid a \in \partial D,\ \tau \in [t_0, t_n],\ I_{a,\tau} = [\tau, t_n] \\ &\qquad \textit{and } p_{a,\tau}(t_n) \in \overset{\circ}{D}\ \}, \end{aligned}$$

where $\overset{\circ}{D}$ denotes the interior of D and ∂D the boundary of D. We note that we know the full path through D of the particle only if it belongs to $\mathscr{P}_v{}^c$. Additionally, we note that we do not know the path of a particle outside the domain D. Hence, a reentering of the domain by a particle would be regarded as an entrance of a new particle.

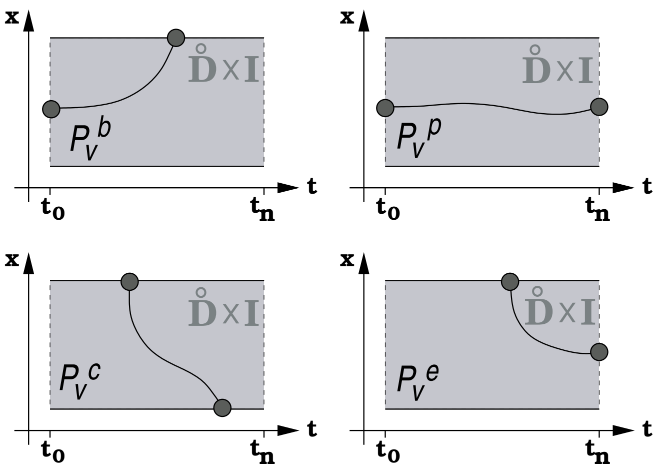
x
$\overset{\circ}{D} \times I$
$P_v{}^b$
t
t_0
t_n
x
$\overset{\circ}{D} \times I$
$P_v{}^p$
t
t_0
t_n
x
$\overset{\circ}{D} \times I$
$P_v{}^c$
t
t_0
t_n
x
$\overset{\circ}{D} \times I$
$P_v{}^e$
t
t_0
t_n

Fig. 1 Different types of particles of $\mathscr{P}_v$. The classification depends on the location of a particle in the domain D during the time interval $I = [t_0, t_n]$. (Note that for illustrative purposes the domain D is one-dimensional.)

A ***pathline predicate*** is a partial map

$$\begin{aligned} P :\ \mathscr{P}_v &\rightarrow \{\ TRUE, FALSE\ \}, \\ p &\mapsto P(p). \end{aligned}$$

We allow a partial map here, because many predicates in practice assume a full path of the particle through the domain, i.e. they can only be defined for $p \in \mathscr{P}_v{}^c$. In the special case, where there is no inflow and no outflow, $\mathscr{P}_v$ only consists of $\mathscr{P}_v{}^p$ and all pathlines are known within the time interval $[t_0, t_n]$. On this basis the particles can be compared. The corresponding characteristic set C_P is defined by

$$C_P = \{ (x,t) \in D \times I \mid P(p_{x,t}) = TRUE \}.$$

This is the set of all points in spacetime visited by particles fulfilling the predicate. For a given time t_i we define the restriction of the characteristic set C_P to t_i as

$$C_P|t_i = \{ (x,t) \in C_P \mid t = t_i \}.$$

5 Unsteady Flow Structure

We are convinced that scientists and engineers try to obtain a mental model of the overall flow behavior. A substantial part of this model is the movement of all particles which we call flow structure. Furthermore, the mental model connects this movement to detected features of the unsteady flow.

Based on pathline predicates, we develop a formal definition for a flow structure of an unsteady flow. Since the flow structure depends on the user's interest in the flow behavior, different flow structures of the same flow are possible and useful. The chosen flow structure determines the outcome of the visualization and the features that are visible in the rendered flow. We start with a finite set $\mathscr{G}$ of pathline predicates with disjunct characteristic sets, i.e.

$$\mathscr{G} = \{ P_\lambda \mid \lambda \in \Gamma \},\ C_{P_\lambda} \cap C_{P_\mu} = \emptyset \quad \forall \lambda, \mu \in \Gamma,\ \lambda \neq \mu$$

for a finite set Γ. Furthermore, we demand

$$\bigcup_{\lambda \in \Gamma} C_{P_\lambda} = \mathscr{P}_v.$$

This partitioning represents the unsteady flow structure and serves as a formal definition. We note that in practice, many pathline predicates are only defined on the complete pathlines ${\mathscr{P}_v}^c$. In this case, $\mathscr{G}$ has to include the three extra predicates $P^p = (p \in {\mathscr{P}_v}^p)$ (the pathlines staying permanent in the domain), $P^e = (p \in {\mathscr{P}_v}^e)$ (the pathlines entering but not leaving the domain), $P^b = (p \in {\mathscr{P}_v}^b)$ (the pathline present in the interior at the beginning and leaving the domain).

6 Computation of Pathlines

As a preprocessing step for the construction of unsteady flow structures, we essentially transform the velocity field from an Eulerian description (given as a the resulting dataset of a simulation) to a Lagrangian description. That means, we want to analyze changes which occur as one follows a fluid particle along its trajectory, instead of examining changes as they occur at a fixed point in the velocity field. This step needs to be done only once for a given dataset. The large computational effort of this preprocessing step has to be related to the effort of the simulation itself. Even if this preprocessing step takes some hours/days on commodity hardware, simulating 3D unsteady flow takes days/weeks on supercomputers.

In theory, there exists an infinite number of pathlines. Depending on the predicate, close pathlines may have different predicate values. Obviously, we must limit the computational burden to get results in finite time. We do not know a priori which predicates will be used in general. This excludes every adaptive optimization approach based on a specific predicate. Hence, we decided to discretize the volume of particles such that two particles have a maximal distance below a value of $2\delta \in \mathbf{R}$ at each time step $t_0, \ldots, t_n$. Of course, other approaches are possible. For this purpose, we voxelize the spacial domain D of the simulation data with voxels of edge length δ. (For meaningful results, we can restrict the predicates to those that have characteristic sets with a diameter greater than δ in most parts of the volume during simulation time.) With respect to the origin of the particles, there are two possibilities. They are either present inside D at time t_0 or they enter D through an inflow part of ∂D during the time interval I.

The flow volume present at t_0 is represented by particles starting at the center of the voxels at time t_0. All the particles are propagated forward in time over the complete time interval I using the vector data of the original grid. (Because of main memory limitations, it could be in some cases faster to propagate all particles from one timestep to the next.) In this process, it becomes clear that the flow does not cover the volume in a dense enough manner, so we add particles to the centers of empty voxels at each timestep. Once we have reached timestep t_n, we have the complete pathlines of all the particles present at t_0. But we are missing the history of the particles that we introduced on the fly to get a dense covering. Therefore, we iterate backwards through the timesteps and compute the missing part of the pathlines of these introduced particles.

In the end of this process, we store each pathline $p_{a,\tau}$, its position at every timestep (i.e. its rasterization), its lifespan $I_{a,\tau}$, and its type according to the classification of $\mathscr{P}_v$ introduced in 4. The by far largest amount of storage is needed for the pathlines and depends on the resolution of their discretization as polylines. But the pathlines have not to be cached in the main memory. For the computation of the predicates, one could go through all the pathlines one by one in a linear fashion. The storage demands for our examples used in Section 10 can be found in Table 1.

7 Computation of Pathline Predicates

While the pathlines have to be computed only once per dataset, each pathline predicate needs to be evaluated on these pathlines. As the examples in the next section show, many predicates require information about the complete path of the particle through the volume. In this case, we limit the calculation to the set ${\mathscr{P}_v}^c$. Each pathline is evaluated using a function for the predicate. Sometimes it is faster to do this in parallel for all particles, especially if the original vector field is needed. If this is not the case, e.g. if the computation is based on the integration of curvature along the pathline and

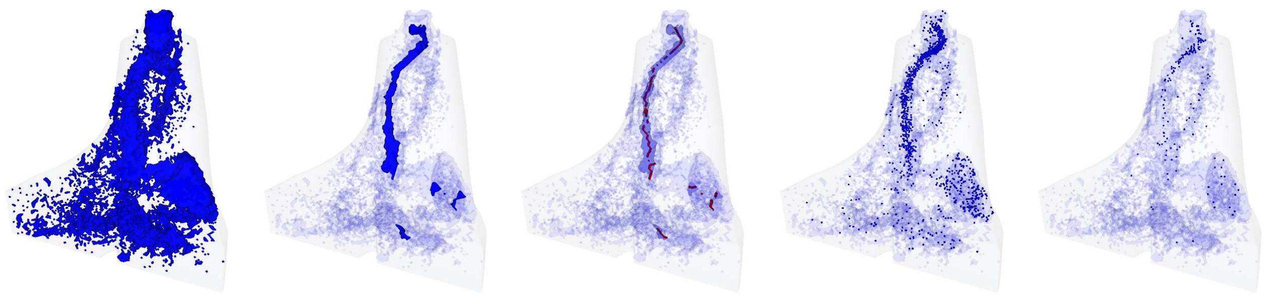

Fig. 2 Snaphot of a particle placement for a given timestep (from left to right): restricted characteristic set, double eroded restricted characteristic set, resulting skeleton, particle placement from initial resolution, and particle placement from quarter resolution

on deciding if it belongs to the top n% (n=10, for example), a sequential approach is faster. In both cases, one can use parallel machines to speed up the calculation.

8 Pathline Predicate Examples

There are two groups of pathline predicates that differ in the amount of additional information needed to evaluate the predicate. Predicates from the first group need only the information about the pathline itself. This has the advantage that we do not have to compute some derived quantities of the unsteady flow field. Examples for this group are predicates that compute geometric characteristics of the pathline like curvature or deviation from a given direction and demand a certain threshold in order to be evaluated as true. Another example uses the time information of the pathline and computes residence time in the dataset. As predicate, one can take a certain quantile of the resulting distribution of the values as a threshold.

The second group comprises predicates that rely on data derived from the unsteady flow, e.g. derived fields like vorticity, helicity, and pressure, or features of the flow like shock-wave boundaries or vortex cores. These predicates are usually computationally more expensive and need more storage, since the additional derived data has to be stored. In the following sections we give an example for each of the two groups that we use in the remainder of the paper.

8.1 Pathlines showing circulating behavior

In many engineering applications, such as fluid dynamics or circulatory systems in medicine, there is a great interest in recirculation zones. Often they lead to large stagnant flow zones that hinder the overall flow, resulting, for example, in inefficient turbines and pumps. In other applications recirculation zones are desired, e.g. to get a cleaner combustion in engines. Recirculation with respect to a recirculation plane can be computed by adding up the winding angle of the projection of the pathlines to this plane. Values above 2π indicate at least one circle. Computation load can be lowered with additional information about the extent of the recirculation area. Based on this information control regions can be defined that indicate circulating behavior if visited multiple times. The resulting pathline predicate would state whether a pathline shows recirculating behavior or not.

8.2 Pathlines Accompanying a Specific Vortex

Scientists and engineers are interested in the interplay between vortices in the flow and flow outside the vortex region. Therefore, we examine pathlines with respect to the behavior towards a vortex. We want to analyze the part of the flow that is influenced by a certain vortex. A common imagination of ”influenced” is a particle which is close enough and swirls around the vortex core. For steady flow a predicate that examines this behavior is given in [20]. Extending this approach to unsteady flow requires an exact tracking of vortex cores from one time step to the next. This is a difficult task and very time consuming. Therefore, we concentrate on the distance between a particle and a vortex. We assume that a particle that accompanies a (moving) vortex core within a small neighborhood distance for some time is influenced by this vortex. Typically, the distance lies within the vortex volume that of course changes its diameter along the vortex core and over time. To compute the vortex volume directly is a nontrivial and error prone task leading to complicated geometries. Hence, the required inside-test would be computational expensive. To reduce the computational cost, we make a simplifying assumption and use a volume with a constant diameter. Of course other loops could also be used. But the circle allows a very fast inside-test (see Section 10). For a given pathline, we track the distance to the vortex core and sum up the time the pathlines reside within the required distance to the vortex during the lifetime of the particle. Doing this for all pathlines (or a representative subset) results in a temporal distribution. Hence, we can use a certain quantile as a minimum time threshold to define the predicate. If a pathline has a longer residence time than the given threshold, the predicate is evaluated as true.

9 Pathline Placement

Flow structures show a partitioning of the flow. They provide those parts of the flow exhibiting a specified behavior. In this structural view individual particles usually cannot be seen. However, tracing the behavior of individual particles within the context of a given flow structure could lead to new ideas for refining this flow structure. Hence, both the structural view and the dynamical view of individual particles cross-fertilize each other.

To obtain a view, that exposes the dynamic behavior of the flow, we need a pathline placement producing on the one hand only a small number of particles for a given time step in order to avoid clutter and on the other hand enough pathlines to be still representative of the underlying flow structure. Looking upon a respective discretized characteristic set as a 4-dimensional region, a representation that captures its essential topology can be obtained in form of a skeleton representation (medial axis). Computing this skeleton for such a hypervolume is still a very challenging task and a subject of ongoing research (e.g., see the works of Jonker [11,12]). But we can take a simpler approach that fits our needs. For every time t_i we see only the restricted characteristic set $C_P|t_i$. A representative pathline placement should assure that for this time span the respective particles should represent the structure of $C_P|t_i$. Therefore, we only compute for every time step the skeleton of the respective restricted characteristic set using a thinning approach from Kuba et. al [13].

Every voxel of the resulting skeleton represents all the voxels of the restricted discrete characteristic set that are no closer to any of the other skeleton voxels. To compute this set of voxels for every skeleton voxel (numbered consecutively through all time steps) we use a flood-fill-algorithm starting for the first run with the skeleton voxels themselves and continue by assigning the next nearest neighbors (i.e. max. 26 voxels) to the respective skeleton voxels for the next runs. In case of conflicting assignments concerning two skeleton voxels we have to calculate the actual distance and assign the voxel to the skeleton voxel with the shortest distance. If the distance is equal we take the first skeleton voxel. After several runs we get a partition of the restricted discrete characteristic set according to the assignment to a skeleton voxel.

We define that a pathline ***"visits"*** a skeleton voxel at time t_i if the respective particle is situated in the skeleton voxel itself or one of its assigned voxels at that time. A set of pathlines visiting all skeleton voxels represents the respective characteristic set. We use an heuristic approach that minimizes the set of pathlines. Starting with the first skeleton voxel we look at all visiting pathlines and choose from this set the one visiting the most unvisited skeleton voxels of later timesteps. We repeat this for the next unvisited skeleton voxel until no unvisited skeleton voxel is left. The resulting set of pathlines is used for the placement. We applied two ways to further lower the number of pathlines. First, we erode the discrete characteristic set to get the main dominating structure, using a simple morphological erosion operator. This results in fewer skeleton voxels and thus fewer pathlines. Second, we discretized the eroded voxelized characteristic set with a lower resolution yielding again fewer skeleton voxels and pathlines (see 2).

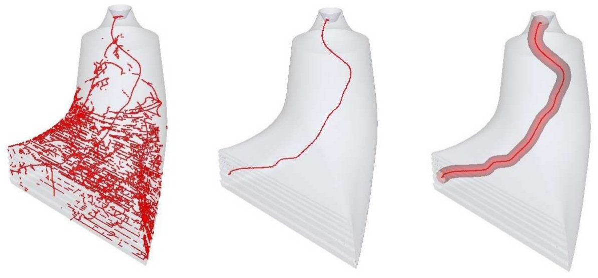

Fig. 3 Vortex core lines of Sujudi-Haimes for a single time step before (left) and after filtering (center). After the filtering one central vortex remains. The right picture shows an isosurface of the corresponding distance field (for the distance value used for pathline predicate S).

10 Results

Our first dataset results from a direct numerical simulation of fluid flow around a cuboid at a Reynolds number of $Re = 1000$. The simulation was carried out with the NaSt3DGP[1] flow solver. A version of the NaSt3DGP code as well as related information and documentation is available for download at http://wissrech.iam.uni-bonn.de/ research/ projects/ NaSt3DGP/ index.htm. The underlying regular grid contains 250,000 cells. The timespan is $I = [0.0, 100.0]$. For the pathline starting points, we voxelize the domain with a resolution of $[99] \times [99] \times [24] \times [120]$, resulting in 2,588,600 pathlines. The first example shows the importance and usefulness of discriminating different particles classes of $\mathscr{P}_v$. We analyze how long particles stay in the flow. Therefore, we compute the staying time for every pathline. From the resulting distribution of staying times we take a certain threshold t_{min} (for our example we take the 97%-quantile). We define the corresponding pathline predicate as:

$$\begin{aligned} ST &= (p \in \mathscr{P}_v{}^c) \wedge \\ &\quad (p \; stays \; more \; than \; or \; equal \\ &\quad \; to \; t_{min} \; in \; the \; domain) \\ ST' &= (p \in \mathscr{P}_v{}^c) \wedge \\ &\quad (p \; stays \; less \; than \; t_{min} \; in \; the \; domain). \end{aligned}$$

For our flow structure we use

$$\mathscr{G}_{Stay} = \{\, ST,\, ST',\, P^b,\, P^e,\, P^p\},$$

where P^b, P^e, and P^p describe the pathlines that are incomplete. Figure 4 shows snapshots of the restricted characteristic sets $C_{ST}|t_i$ (colored in blue), $C_{ST'}|t_i$ (colored in turquoise), and $C_{P^p}|t_i$ (colored in magenta) at times $t_1 = 0.0$, $t_2 = 3.3$, $t_3 = 11.6$, $t_4 = 19.7$, $t_5 = 35.8$, $t_6 = 45.8$, $t_7 = 77.5$, and $t_8 = 95.0$. $C_{P^b}|t_i$ and $C_{P^e}|t_i$ are left transparent. At time step t_0 there is no inflow. The particles from $C_{P^p}|t_0$, that will stay in the domain the entire simulation, are especially located

[1] NaSt3DGP was developed by the research group in the Division of Scientific Computing and Numerical Simulation at the University of Bonn. It is essentially based on the code described in a book by Griebel *et al.* [7].

around the cuboid. The complementary part of the flow is comprised of particles from $C_{P^b}|t_0$. The next images for time steps t_1, t_2, t_3, t_4, and t_5 show the advancing inflow. Especially the flow around the cuboid is mainly comprised of particles from C_{ST}. Flow not hindered by the cuboid moves quickly to the outflow boundaries and hence is comprised of particles from $C_{ST'}$. At time step t_6 nearly the entire domain is filled with inflow and the structure of $C_{ST}|t_6$ can be seen: a meandering pattern resulting from the vortex street behind the cuboid. In the next image the inflow of $C_{P^e}|t_7$ can be seen at time step t_7. At time step t_8 nearly all particles from C_{ST} and $C_{ST'}$ left the domain. Particles of $\mathscr{P}_p{}^c$ are still mainly centered around the cuboid indicating circular behavior of the flow around the cuboid. The series of images shows, that even with this very simple pathline predicate and with the distinction of different classes of $\mathscr{P}_v{}^c$, the user can obtain important insights into the flow. Based on this results further analysis can be started.

Next, we apply our framework to a dataset resulting from a simulation of the flow in the upper part of a draft tube of a Francis turbine. The flow enters the draft tube with a strong residual swirl. This helps to guide the flow around the 90 degree bend in the draft tube. The bounding box of the draft tube is $D = [-0.55, 0.55] \times [-0.45, 0.25] \times [-1.2, -0.07]$. The timespan is $I = [0.0, 9.6]$. The underlying unstructured grid contains 5 million tetrahedra. For the pathline starting points, we voxelize the domain with a high resolution of $[82] \times [62] \times [75] \times [1200]$, resulting in 5,838,785 pathlines.

For our first analysis of this dataset, we examine the flow with respect to its vortices. Therefore, we extract the dominating vortices and build up a flow structure based on the pathline predicates from section 8.

First we compute the vortex core lines for each time step. The left image in figure 3 shows the resulting segments for one time step after applying the method of Sujudi-Haimes [25] formulated with the parallel vector operator of Peikert and Roth [17]. One dominating vortex is clearly visible but also much "noise" (false positives or small vortices) and artefacts at the boundaries. Hence, we filter out all lines under a certain length (arc length = 0.7). The result of the filtering is also shown in figure 3 (center).

To compute the pathline predicate from 8.2, we need an effective method for the necessary distance calculations. For each time step, we compute a distance field for the dominant vortex on the positions of the dataset grid. This reduces minimum distance calculations to a simple interpolation in the distance field at an interrogated position in space and time. Figure 3 (right) shows the isosurface of the distance field concerning the main vortex for an isovalue of 0.04 (which we use as maximal neighborhood distance for our computations) in a total range from 0.0 to 1.0 For each pathline, we calculate the residence time within the neighboring distance to the vortex. From the resulting distribution of residence times, we take the value of the 50%-quantile as minimum residence time $t_{min} = 0.056$ and define the following pathline predicate S and its opposite S' as

$$\begin{aligned} S &= (p \in \mathscr{P}_v{}^c) \wedge \\ &\quad (p\ stays\ more\ than\ or\ equal \\ &\quad\ to\ t_{min}\ in\ the\ vortex\ neighborhood) \\ S' &= (p \in \mathscr{P}_v{}^c) \wedge \\ &\quad (p\ stays\ less\ than\ t_{min}\ in\ the\ vortex\ neighborhood). \end{aligned}$$

The predicates P^b, P^e, and P^p describe the pathlines that are incomplete. We use the finite set of pathline predicates

$$\mathscr{G}_{Vortex} = \{\, S,\, S',\, P^b,\, P^e,\, P^p \}$$

to build up the flow structure. Figure 5 shows snapshots of the restricted characteristic sets $C_S|t_i$ (colored in blue) and $C'_S|t_i$ (colored in turquoise) at times $t_1 = 0.024,\ t_2 = 0.104,\ t_3 = 0.312,\ t_4 = 0.456,\ t_5 = 0.488,\ t_6 = 0.608,\ t_7 = 6.92$, and $t_8 = 9.512$. $C_{P^e}|t_i$ and $C_{P^b}|t_i$ are left transparent. At timestep t_1 and t_2, an early stage of the simulation can be seen. The main inflow is at the top of the draft tube and a minor inflow at the bottom. The flow close to the walls streams much faster than the remaining flow. The basic periodic movement of the vortex can be seen at t_3, t_4, and t_5. The vortex alternates from the left to the right side of the draft tube. The vortex looses particles and strength especially when hitting the right side. The particles outside the vortex travel towards the outlet. At t_6 and t_7, one can see a small opposite vortex fed with particles from the main vortex. The vortex moves particles upwards contrary to the main vortex. Once the particles hit the main vortex again they change their direction and flow again downward. This indicates the existence of a circulation flow embedded in the overall flow. Finally, at t_8, we see a larger set $C_{P^e}|t_8$ that will stay in the domain till the end. Additional to the remaining downward flow of $C_S|t_8$ and $C'_S|t_8$ there is a small part of $C_S|t_8$ residing at the top of the draft tube indicating a circle in the flow. Figure 6 shows individual particles resulting from our pathline placement method. A subset of this particles is shown together with faded traces of the last three timesteps. The traces' length gives a hint of the particle velocity. The particles of the subset result from a pathline placement based on a coarser voxelization of the characteristic set (see section 9). The series of particle snapshots shows the vortex hitting the right side of the draft tube and at the same time losing particles. To see the whole motion of particles and the animation of the structural view we refer the reader to the videos accompanying this paper.

Next, we investigate the part of the flow showing the circulating behavior indicated earlier. We compute the circulating behavior as explained in section 8 using an xy-plane at the center of the draft tube as a control plane. We define the following predicate C_1 and and its opposite C'_1:

$$\begin{aligned} R_1 &= (p \in \mathscr{P}_v{}^c) \wedge \\ &\quad (p\ makes\ at\ least\ one\ circle\ around\ the\ center) \\ R'_1 &= (p \in \mathscr{P}_v{}^c) \wedge \\ &\quad (p\ makes\ no\ circle\ around\ the\ center) \end{aligned}$$

and the predicates P^b, P^e, and P^p again describe the pathlines that are incomplete. For our flow structure we use

$$\mathscr{G}_{Recirc} = \{\, R_1,\, R'_1,\, P^b,\, P^e,\, P^p \}.$$

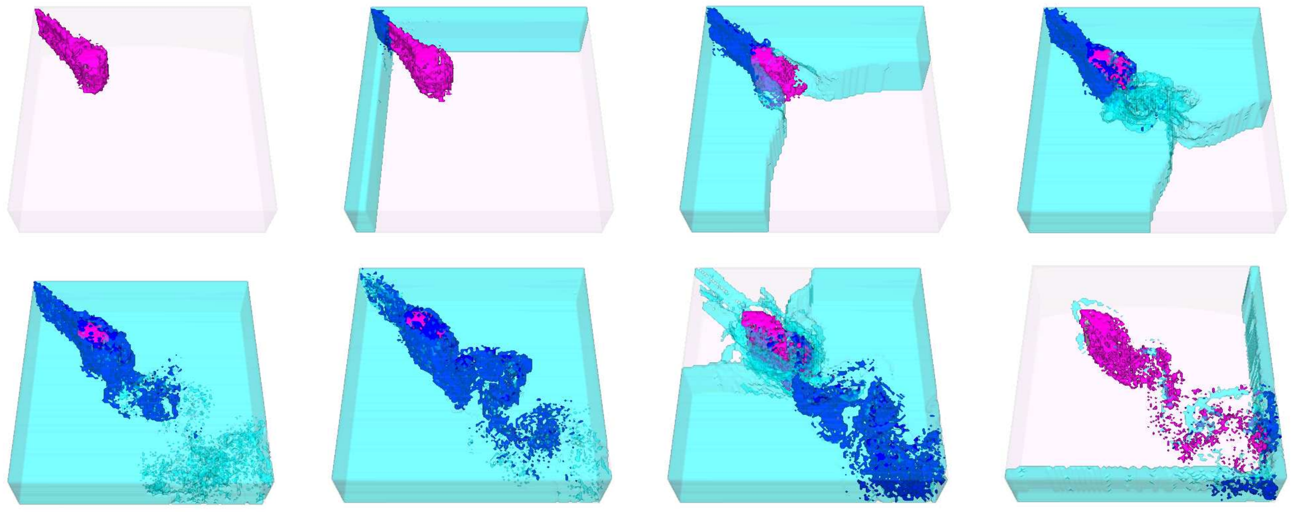

Fig. 4 Structural view of $\mathcal{G}_{Stay}$: boundaries of restricted characteristic sets $C_S|t_i$ (colored in blue), $C_{ST}|t_i$ (colored in turquoise), and $C_{P^p}|t_i$ (colored in magenta) at times $t_1 = 0.0,\ t_2 = 3.3,\ t_3 = 11.6,\ t_4 = 19.7,\ t_5 = 35.8,\ t_6 = 45.8,\ t_7 = 77.5,$ and $t_8 = 95.0$. $C_{P^e}|t_i$ and $C_{P^b}|t_i$ are left transparent.

The upper series of pictures in figure 7 shows the restricted characteristic sets $C_{R_1}|t_i$ (colored in dark green) and $C_{R'_1}|t_i$ (colored in light green) at time steps $t_1 = 0.08,\ t_2 = 0.28,\ t_3 = 0.592,\ t_4 = 1.192$. $C_{P^e}|t_i$ and $C_{P^b}|t_i$ are left transparent. During this time period the recirculation area is built up on the left side of the draft tube. The lower series of pictures illustrates the flow at time $t_5 = 6.04$ using the same color scheme. It seems that the recirculation area now covers nearly the entire draft tube. But $C_{R_1}|t_5$ contains all particles leaving this area downstream or getting into this area from above besides the recirculation area. To get the recirculation area, we therefore apply an erosion operation on $C_{R_1}|t_5$ as shown in the second and third picture of this series. The fourth picture shows in lightgray $C_S|6.04$ together with the recirculation area, illustrating that this area is generated by the main vortex. Figure 8 shows a series of snapshots of individual particles showing this recirculatory behavior.

The discussed examples proved pathline predicates and unsteady flow structures to be a useful tool for the analysis of 3D unsteady flow. Working with large 3D unsteady vector fields by means of commodity PC hardware is still one of the most challenging tasks because of storage demands and computational costs. Tables 2 and 3 show computation times on a PC (AMD Opteron 224, 8GB RAM, single core) for the different tasks leading to a flow structure. The by far most time consuming task is the initial preprocessing step needed once for each dataset. Additionally, the computation of the predicates and the final animations of the structures take some time. Therefore, future work will exploit acceleration potentials especially by means of parallelization. Emerging multi-core CPUs, the use of GPUs for integration purposes, and powerful PC clusters should dramatically cut down computation times.

Dataset	Pathline Data	Rasterization	Attributes
Cuboid	18 GB	1.2 GB	5.5 MB
Turbine	42 GB	27 GB	45 MB

Table 1 Memory needed to store pathline, raster and attribute data for different datasets.

Task	Computation Time
integration + rasterization + storage	89h
predicate vortex	2.5h
predicate circulation	1h
particle placement + animation	1.2h

Table 2 Computation times for the turbine dataset

Task	Computation Time
integration + rasterization + storage	35h
predicate staying timex	0.3h

Table 3 Computation times for the cuboid dataset

11 Conclusion

This article has introduced the notion of pathline predicates and unsteady flow structure. These concepts allow to formalize and compute a partitioning of the dynamics, i.e. the selection of particles based on the interests of a user. Different structures can be defined on the same dataset reflecting varying user interests. Based on the flow structure, we proposed a pathline placement allowing the user to track individual particles showing the behavior specified in the pathline predicates. Combining the structural view of a flow structure with the dynamic view of tracking individual particles yields to a proper illustration of the flow the user is interest in. For our future work, we intend to study more datasets and other predicates allowing different flow structures. Additionally,

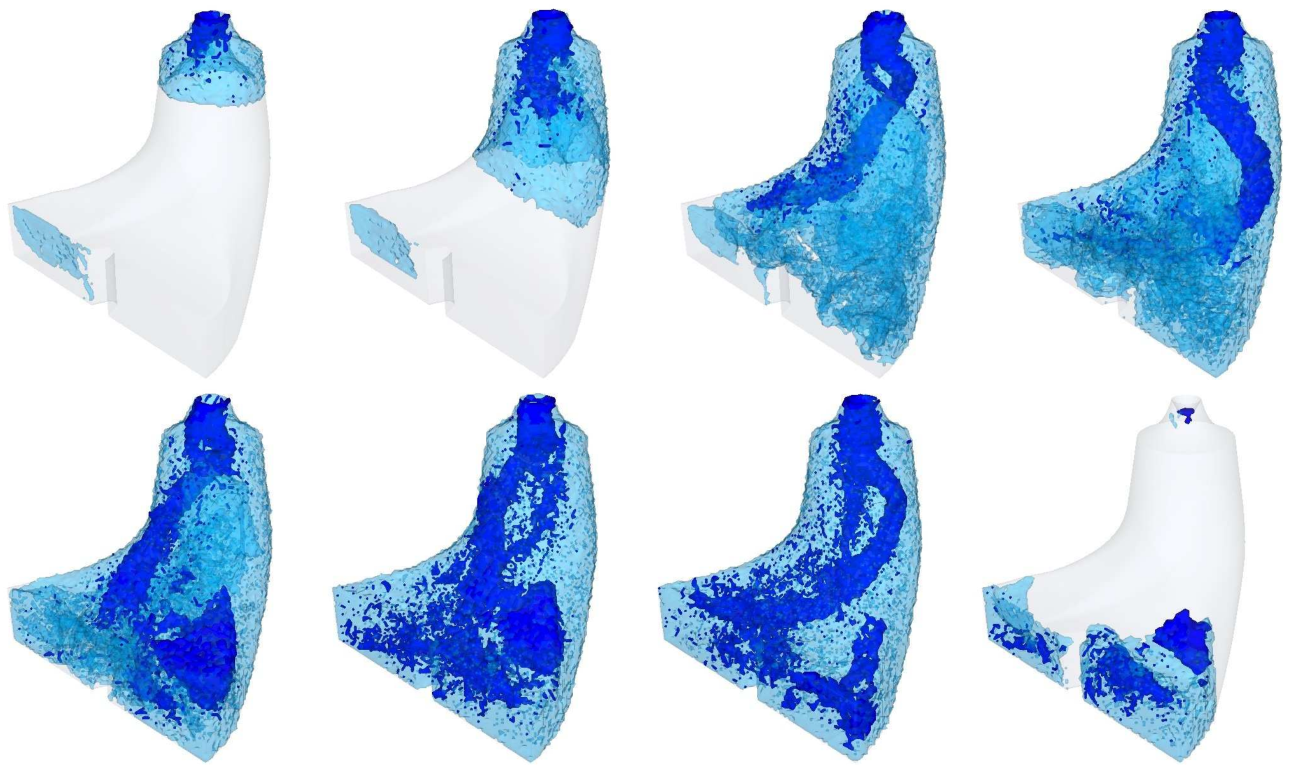

Fig. 5 Structural view of $\mathscr{G}_{Vortex}$: boundaries of restricted characteristic sets $C_S|t_i$ (colored in blue) and $C'_S|t_i$ (colored in turquoise) at times $t_1 = 0.024,\ t_2 = 0.104,\ t_3 = 0.312,\ t_4 = 0.456,\ t_5 = 0.488,\ t_6 = 0.608,\ t_7 = 6.92$, and $t_8 = 9.512$. $C_{P^e}|t_i$ and $C_{P^b}|t_i$ are left transparent.

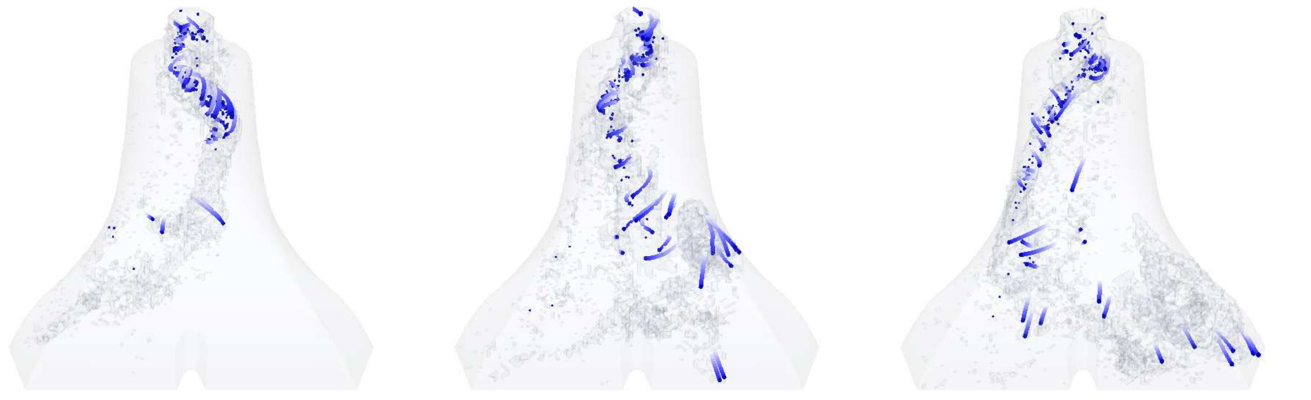

Fig. 6 Dynamic view of $\mathscr{G}_{Vortex}$: Individual particles at three time steps $t_1 = 0.416,\ t_2 = 0.544$, and $t_3 = 0.624$ are depicted in blue. Corresponding restricted characteristic sets $C_S|t_1,\ C_S|t_2$, and $C_S|t_3$ are colored in light gray. A subset of particles have an additional faded trace of the last three time steps. This series of snapshots shows the loss of particles of the vortex colliding with the right side of the draft tube.

we want to improve computation times by exploiting all parallelization potentials.

Acknowledgements The authors wish to thank Ronald Peikert from ETH Zürich for providing the datasets. Furthermore, thanks go to all members of the FAnToM development team for their programming efforts. This work was partly supported by DFG grant SCHE 663/3-7.

References

1. Bauer, D., Peikert, R.: Vortex tracking in scale space. In: Data Visualization 2002. Proc. VisSym 2002, pp. 233–240 (2002)
2. Bauer, D., Peikert, R., Sato, M., Sick, M.: A case study in selective visualization of unsteady 3d flow. In: IEEE Visualization 2002, pp. 525–528 (2002)
3. Becker, B., Lane, D., Max, N.: Unsteady Flow Volumes. In: IEEE Visualization 1995, pp. 329–335 (1995)

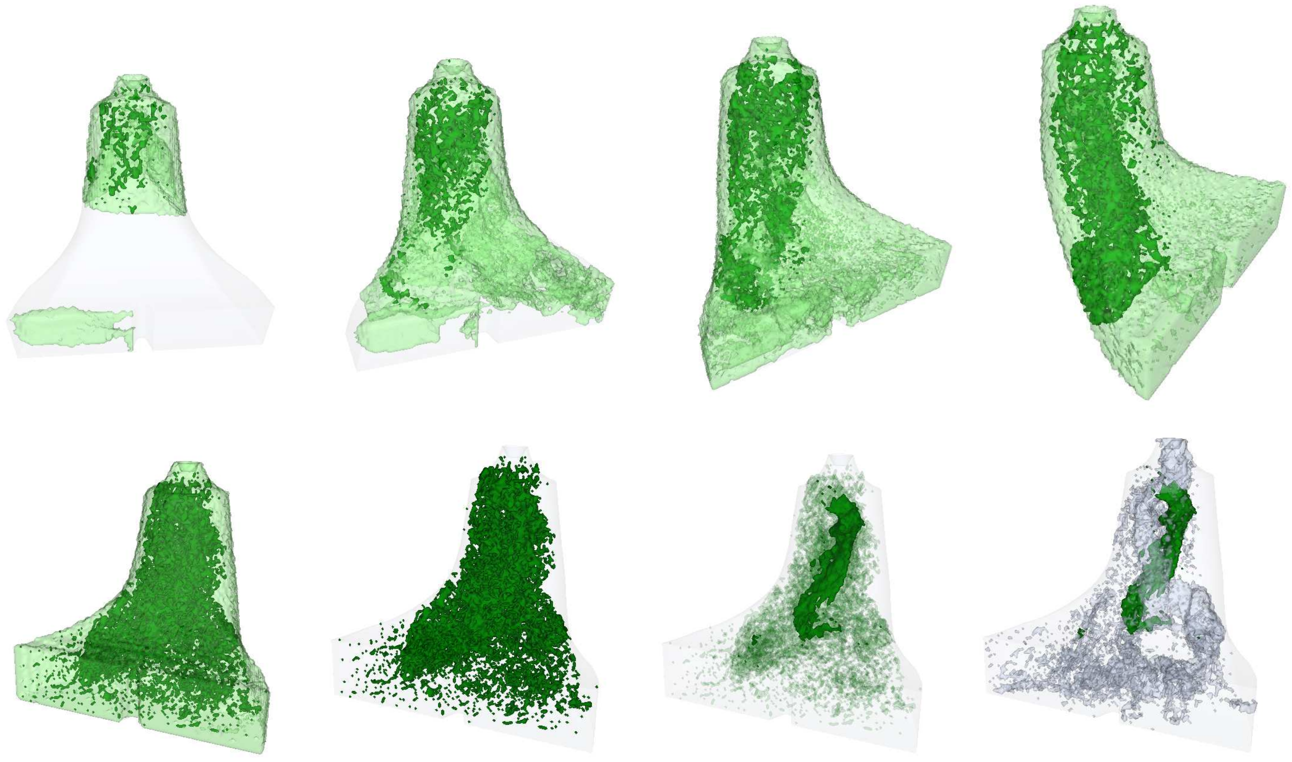

Fig. 7 Structural view of $\mathscr{G}_{Recirc}$: The upper series of pictures shows the restricted characteristic sets $C_{R_1}|t_i$ (colored in dark green) and $C_{R'_1}|t_i$ (colored in light green) at time steps $t_1 = 0.08,\ t_2 = 0.28,\ t_3 = 0.592$, and $t_4 = 1.192$. $C_{P^e}|t_i$ and $C_{P^b}|t_i$ are left transparent. The lower series of pictures illustrate the flow at time step $t_5 = 6.04$ using the same color scheme. To get the main region of $C_{R_1}|t_5$, an erosion operator is applied to $C_{R_1}|t_5$ two times (last but one picture). The last picture shows the eroded region of $C_{R_1}|t_5$ together with $C_S|t_5$ (colored in light gray).

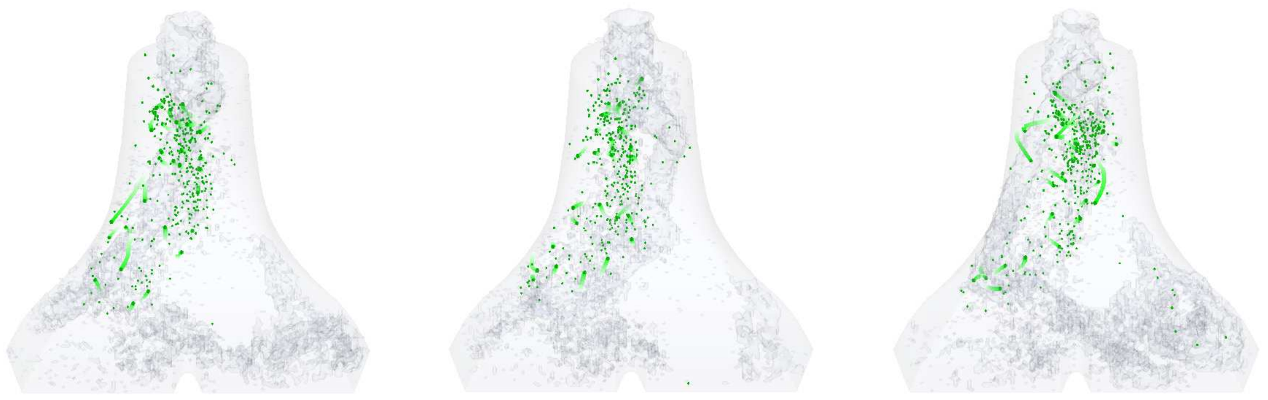

Fig. 8 Dynamic view view of $\mathscr{G}_{Recirc}$: Individual particles at three time steps $t_1 = 1.736,\ t_2 = 1.808$, and $t_3 = 2.016$ are depicted in green. For some particles an additional faded trace of the last three timesteps is shown. The corresponding restricted characteristic sets $C_S|t_1,\ C_S|t_2$, and $C_S|t_3$ of the vortex pathline predicate (shown in light gray) illustrate the connection between vortex movement and recirculation of the flow.

4. Cabral, B., Leedom, L.C.: Imaging Vector Fields Using Line Integral Convolution. Computer Graphics **27**(4), 263 – 270 (1993). SIGGRAPH '93
5. Forsell, L., Cohen, S.: 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)
6. Garth, C., Tricoche, X., Scheuermann, G.: Tracking of Vector Field Singularities in Unstructured 3d Time-Dependent Datasets. In: IEEE Visualization 2004, pp. 329–336 (2004)
7. Griebel, M., Dornseifer, T., Neunhoeffer, T.: Numerical Simulation in Fluid Dynamics, a Practical Introduction. SIAM, Philadelphia (1998)
8. Ji, G., Shen, H., Wenger, R.: Volume Tracking Using Higher Dimensional Isosurfacing. In: IEEE Visualization 2003, pp. 209–216 (2003)
9. Jobard, B., Erlebacher, G., Hussaini, M.: Hardware-Assisted Texture Advection for Unsteady Flow Visualization. In: IEEE Visualization 2000, pp. 155–162 (2000)
10. Jobard, B., Erlebacher, G., Hussaini, M.: Lagrangian-Eulerian Advection of Noise and Dye Textures for Unsteady Flow Visualization. IEEE Transactions on Visualization and Computer Graphics **8**(3), 211 – 222 (2002)
11. Jonker, P.: Morphological Operations on 3d and 4d Images: From Shape Primitive Detection to Skeletonization. In: DGCI, pp. 371–391 (2000)
12. Jonker, P.: Skeletons in n Dimensions Using Shape Primitives. Pattern recognition letters **23**(6), 677–686 (2002)
13. Kuba, A., Palagyi, K.: A 3D 6-Subiteration Thinning Algorithm for Extracting Medial Lines. Pattern Recognition Letters **19**(7), 613 – 627 (1998)
14. Lane, D.: Scientific Visualization of Large-Scale Unsteady Fluid Flows. In: Scientific Visualization, pp. 125 – 145. IEEE Computer Society, Los Alamitos, CA (1997)
15. Liu, Z., Moorhead II, R.: Accelerated Unsteady Flow Line Integral Convolution. IEEE Transactions on Visualization and Computer Graphics **11**(2), 113 – 125 (2005)
16. Park, S., Budge, B., Linsen, L., Hamann, B., Joy, K.: Dense Geometric Flow Visualization. In: K. Brodlie, D. Duke, K. Joy (eds.) Proceedings of Eurographics/IEEE-VGTC Symposium on Visualization 2005 (EuroVis 2005) (2005)
17. Peikert, R., Roth, M.: The Parallel Vectors Operator - a Vector Field Visualization Primitive. In: IEEE Visualization 1999, pp. 263 – 270. San Francisco, CA (1999)
18. Post, F., Vrolijk, B., Hauser, H., Laramee, R., Doleisch, H.: The State of the Art in Flow Visualization: Feature Extraction and Tracking. In: Computer Graphics Forum 22, vol. 4, pp. 775–792 (2003)
19. Reinder, F., Post, F.H., Spoelder, H.: Attribute-Based Feature Tracking. In: Data Visualization 1999. Proc. VisSym 1999, pp. 63–72 (1999)
20. Salzbrunn, T., Scheuermann, G.: Streamline Predicates. IEEE Transactions on Visualization and Computer Graphics **12**(6), 1601 – 1612 (2006)
21. Samtaney, R., Silver D. Zabusky, N., Cao, J.: Visualizing Features and Tracking Their Evolution. IEEE Computer **27**(7), 20 – 27 (1994)
22. Shen, H., Kao, D.: A new line integral concolution algorithm for visualizing time-varying flow fields. IEEE Transactions on Visualization and Computer Graphics **4**(2), 98 – 108 (1998)
23. Shi, K., Theisel, H., Hauser, H., Weinkauf, T., Matkovic, K., Hege, H., Seidel, H.: Path Line Attributes - an Information Visualization Approach to Analyzing the Dynamic Behavior of 3d Time-Dependent Flow Fields. In: submitted to TopoInVis 2007
24. Silver, D., Wang, X.: Tracking and visualizing turbulent 3d features. IEEE Transactions on Visualization and Computer Graphics **3**(2), 129 – 141 (1997)
25. Sujudi, D., Haimes, R.: Identification of Swirling Flow in 3D Vector Fields. Tech. Rep. AIAA Paper 95–1715, American Institute of Aeronautics and Astronautics (1995)
26. Theisel, H., Seider, H.: Feature Flow Fields. In: Data Visualization 2003. Proc. VisSym 2003, pp. 141–148 (2003)
27. Theisel, H., Weinkauf, T., Hege, H., Seidel, H.: Topological Methods for 2d Time-Dependent Vector Fields Based on Streamlines and Pathlines. IEEE Transactions on Visualization and Computer Graphics **11**(4), 383 – 394 (2005)
28. Tricoche, X., Scheuermann, G., Hagen, H.: Topology-Based Visualization of Time-Dependent 2d Vector Fields. In: Data Visualization 2001. Proc. VisSym 2001, pp. 117–126 (2001)
29. Tricoche, X., Wischgoll, T., Scheuermann, G., Hagen, H.: Topological Tracking for the Visualization of Timedependent Two-Dimensional Flows. Computers & Graphics **26**(2), 249–257 (2002)
30. Verma, V., Kao, D., Pang, A.: Plic: Bridging the Gap Between Streamlines and Lic. In: IEEE Visualization 1999, pp. 341–348 (1999)
31. van Walsum, T., Post, F.H., Silver, D., Post, F.J.: Feature Extraction and Iconic Visualization. IEEE Transactions on Visualization and Computer Graphics **2**(2), 111 – 119 (1996)
32. Weigle, C., Banks, D.: Extracting Iso-Valued Features in 4-Dimensional Scalar Fields. In: Proc. Symposium on Volume Visualization, pp. 103–110 (1998)
33. Weiskopf, D., Erlebacher, G., Ertl, T.: A Texture-Based Framework for Spacetime-Coherent Visualization of Time-Dependent Vector Fields. In: IEEE Visualization 2003, pp. 107–114 (2003)
34. Westermann, R., Johnson, C., Ertl, T.: Topology-Preserving Smoothing of Vector Fields. IEEE Transactions on Visualization and Computer Graphics **7**(3), 222 – 229 (2001)
35. van Wijk, J.: Image Based Flow Visualization. In: ACM SIGGRAPH 2002, pp. 745–754 (2002)
36. Wischgoll, T., Scheuermann, G., Hagen, H.: Tracking Closed Streamlines in Time-Dependent Planar Flows. In: T. Ertl, B. Girod, H. Niemann, H.P. Seidel (eds.) Vision, Modeling, and Visualization 2001, pp. 447 – 454. Aka, Berlin (2001)

Tobias Salzbrunn received an M.S. degree in computer science in 2003 from the University of Kaiserslautern, Germany. Afterwards he worked as a research assistant at the same university. Currently, he is working as a research assistant at the Department of Computer Science at the University of Leipzig, Germany. His research interests include dynamical systems and scientific visualization, especially flow visualization.

Christoph Garth received his Diplom (M.S.) degree in mathematics and computer science in 2003 from the University of Kaiserslautern, Germany. Currently, he is working as a Ph.D. student in the Computer Graphics & Visualization Group at the University of Kaiserslautern. His research interests include computer graphics, visualization systems and scientific visualization of vector fields and simulation data.

Gerik Scheuermann received his B.S. and M.S. degrees in mathematics in 1995 from the University of Kaiserslautern, Germany. In 1999, he received a Ph.D. degree in computer science, also from the University of Kaiserslautern. From 1995-1997, he conducted research at Arizona State University for about one year. He worked as a postdoctoral researcher at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis, in 1999 and 2000. Between 2001 and 2004, he was an assistant professor of computer science at the University of Kaiserslautern. Currently, he is a full professor in the Computer Science Department of the University of Leipzig, Germany. His research interests include algebraic geometry, topology, Clifford algebra, image processing, graphics, and scientific visualization. He is a member of the ACM, IEEE Computer Society, and GI.

Joerg Meyer is an Assistant Professor with a shared appointment in the Department of Electrical Engineering & Computer Science and the Department of Biomedical Engineering in the Henry Samueli School of Engineering at the University of California, Irvine. He joined UC Irvine in 2002. Dr. Meyer is also affiliated with the Center of GRAVITY (Graphics, Visualization and Imaging Technology) in the California Institute for Telecommunications and Information Technology (Calit2). He received his Ph.D. from the University of Kaiserslautern, Germany, in 1999. He held an appointment as a post-doctoral researcher and lecturer in the Computer Science Department at the University of California, Davis, from 1999 to 2000, and maintains an Adjunct Assistant Professorship at the Computer Science and Engineering Department at Mississippi State University, where he was also affiliated with an NSF-sponsored Engineering Research Center (2000-2002). His research interests include scientific visualization, computer graphics, biomedical imaging, and virtual reality.