Time-Dependent Visualization of Lagrangian Coherent Structures by Grid Advection

Filip Sadlo[1], Alessandro Rigazzi[2], and Ronald Peikert[2]

[1] VISUS, Universität Stuttgart, Germany
sadlo@visus.uni-stuttgart.de
[2] Computer Graphics Laboratory, Computer Science Department,
ETH Zurich, Switzerland {arigazzi, peikert}@inf.ethz.ch

Abstract. Lagrangian coherent structures play an important role in the analysis of unsteady vector fields because they represent the time-dependent analog to vector field topology. Nowadays, they are often obtained as ridges in the finite-time Lyapunov exponent of the vector field. However, one drawback of this quantity is its very high computational cost because a trajectory needs to be computed for every sample in the space-time domain. A focus of this paper are Lagrangian coherent structures that are related to predefined regions such as boundaries, i.e. related to flow attachment and flow separation phenomena. It presents an efficient method for computing the finite-time Lyapunov exponent and its height ridges only in these regions, and in particular, grid advection for the efficient computation of time series of the finite-time Lyapunov exponent, exploiting temporal coherence.

1 Introduction

One of the major challenges in scientific visualization is the visualization of time-dependent velocity fields represented by hundreds of time steps, each given as a large numerical dataset. Velocity fields are among the most important results of computational fluid dynamics (CFD) simulations, and therefore visualization of such data has been extensively studied. It is quite commonly agreed that, due to the complexity of the data, a single visualization technique is in general not able to reveal all the relevant structures in the flow. Structures can not only appear at many different spatial and temporal scales, but their recognition may also depend on the correct frame of observation.

The higher resolution of today's simulation results leads to more and more intricate flow details which are to be captured by appropriate visualization techniques. Methods for such structural visualization can be divided into feature-based and topological approaches. The latter have, until recently, largely been seen as synonymous to *vector field topology* [1,8]. Because of its rigorous foundation on the theory of dynamical systems, vector field topology is very popular in the visualization community. One of its most powerful concepts is the *separatrix* which separates two regions of qualitatively different flow behavior. Vector field topology can also be applied to the wall shear stress field on no-slip boundaries where the velocity vanishes. By combining it with the topology of the velocity field in the interior, Surana et al. were able to give exact definitions of

separation and attachment surfaces, and they showed that for Navier-Stokes flows, the separation slope and angle formulas depend on on-wall quantities only [19, 20].

In a strict sense, vector field topology is only applicable to steady or instantaneous velocity fields. But the lack of alternatives, the simple concepts, and the availability in visualization systems led to the frequent use of vector field topology also for the visualization of unsteady fields. Even if most researchers were probably aware that such a visualization based only on snapshots cannot be correct, this was mostly seen as a theoretical blemish only. Shadden et al. [17] demonstrated with their simple two-dimensional "double gyre" example that the separatrix can be clearly dislocated from the actual flow separation. As a reaction, there is currently a growing interest in the search for time-dependent variants or extensions of vector field topology. Theisel et al. [21] and Shi et al. [18] presented such concepts for aperiodic and periodic velocity fields, respectively. As a more radical approach, visualization researchers started to look into the theory of *Lagrangian coherent structures* (LCS) as a replacement for vector field topology. In the original sense, an LCS was defined as a region of coherent flow behavior. In Hussain's definition [9] flow behavior is expressed by vorticity alone, while Robinson [14] defined coherent motion as "a region over which at least one fundamental flow variable exhibits significant correlation with itself or with another variable over a range of space and/or time that is significantly larger than the smallest local scales". In a more modern sense, LCS are understood as the boundaries of such regions. As Haller showed [7], they can be computed as *ridges* of the (maximal) *finite-time Lyapunov exponent* (FTLE). These ridges are lower-dimensional structures, which can be classified into attracting and repelling LCS, correspond to the unstable and stable manifolds (separatrices) in vector field topology.

Since the Lyapunov exponent is constant along a trajectory, this holds approximately for its finite-time version if the integration time is chosen to be sufficiently long. Therefore, LCS computed numerically from this quantity are close to material surfaces, i.e. they are essentially advected with the flow. Ideal LCS are material surfaces [7]. For that reason, these structures are of interest for the study of transport and mixing processes in fluid dynamics. In visualization, LCS have been used only recently. Garth et al. [5, 6] visualized the underlying FTLE (scalar) field with slicing and direct volume rendering techniques, using appropriate transfer functions to make LCS recognizable as the ridges of the field. Sadlo et al. [16] compared visualizations based on vector field topology and on LCS, and introduced visualization of the latter by explicit extraction of height ridges of the FTLE field [15]. Bürger et al. [3] computed LCS for the purpose of controlling the seeding in particle based visualizations.

In this paper, we present an efficient method for computing the finite-time Lyapunov exponent and its height ridges as time series. The method maintains a sampling grid that grows and shrinks with the ridges that it contains and that is advected with the flow between the steps of the time series. The grid is initialized by the user in a region of interest which can be located anywhere in the domain. By initializing the grid near a solid boundary, flow separation and attachment surfaces are obtained. An advantage of this visualization method is that it does not rely mainly on the data next to the boundary, and in particular does not need the computation of derivatives in cells adjacent to the boundary.

2 Background

In this section we give a short introduction to the two concepts which are central for this paper, the finite-time Lyapunov exponent and the height ridge, and we briefly discuss practical aspects of their computation from discrete data.

2.1 Finite-Time Lyapunov Exponents

Given a time dependent velocity field $\mathbf{v}(\mathbf{x},t)$ on a domain $D \subseteq \mathbb{R}^n$, a trajectory (or path line) $\mathbf{x}(t;t_0,\mathbf{x}_0)$ starting at point $\mathbf{x}_0$ at time t_0 is a solution of the initial value problem

$$\dot{\mathbf{x}}(t;t_0,\mathbf{x}_0) = \mathbf{v}(\mathbf{x}(t;t_0,\mathbf{x}_0),t), \quad \mathbf{x}(t_0;t_0,\mathbf{x}_0) = \mathbf{x}_0. \tag{1}$$

For fixed times t_0 and t, the trajectories give rise to the *flow map*

$$\phi_{t_0}^t : D \to D, \quad \mathbf{x}_0 \mapsto \mathbf{x}(t;t_0,\mathbf{x}_0). \tag{2}$$

The gradient $\nabla\phi_{t_0}^t$ of the flow map describes the deviation of infinitesimally close trajectories started at the same time t_0, and the tensor

$$\Delta_{t_0}^t = \left(\nabla\phi_{t_0}^t(\mathbf{x}_0)\right)^\top \nabla\phi_{t_0}^t(\mathbf{x}_0) \tag{3}$$

expresses the deformation of the neighborhood of $\mathbf{x}_0$ under the flow map. This symmetric tensor has real eigenvalues λ_i based on which the i-th *Lyapunov exponent* is defined as follows:

$$\sigma_i = \lim_{T\to\infty} \frac{1}{T} \ln \sqrt{\lambda_i(\Delta_{t_0}^{t_0+T})}. \tag{4}$$

The spectrum of Lyapunov exponents is a property of an entire trajectory, i.e. it does not depend on the choice of t_0 on that trajectory. By replacing the limit with a fixed integration time T, the finite-time Lyapunov exponent (FTLE) is obtained. Usually, only the maximum FTLE is of interest, which is given by:

$$\sigma_{t_0}^{t_0+T} = \frac{1}{T} \ln \sqrt{\lambda_{\max}(\Delta_{t_0}^{t_0+T})}. \tag{5}$$

Unlike the Lyapunov exponent, the FTLE depends on both the starting time t_0 and the integration time T.

For the numerical computation of either Lyapunov exponents or FTLE, one has to estimate the flow map gradient by using trajectories started very close to the reference trajectory. However, trajectories may separate at an exponential rate from the reference trajectory. In fact, detecting this behavior is the main motivation behind these concepts. Therefore, trajectories must undergo frequent *renormalization* [2], which is equivalent to breaking up the integration in pieces and computing the flow map gradient as the product of the piece-wise obtained gradients.

The FTLE, and even more the Lyapunov exponents, can exhibit finely detailed structures with a spatial variation that can far exceed that of $\mathbf{v}(\mathbf{x},t)$. Therefore, it is often not the goal to do an accurate computation of an FTLE at a given point in the domain, but rather to compute a spatial average at a resolution defined by a discretization grid. This leads to a discrete version of the FTLE [7] where the flow map is sampled on the nodes of a grid and gradients are then estimated by finite differences (rather than using trajectories in close vicinity and applying renormalization).

2.2 Height Ridges

The notion of a local maximum of a scalar field $s : \mathbb{R}^n \to \mathbb{R}$ is unambiguously defined by a vanishing gradient and negative second derivatives in all possible directions. In contrast to this, there are several ways of relaxing this definition in order to obtain k-dimensional maxima or minima. The *height ridge* [4] is the most straightforward and the most widely used such definition. For a point on a k-dimensional ridge it requires vanishing first derivatives and negative second derivatives only in a $n-k$-dimensional subspace. Formally, if $\mathbf{H}$ denotes the Hessian of s, $\mathbf{H} = \left(\partial^2 s / \partial x_i \partial x_j\right)_{ij}$, and $y_1, \cdots y_n$ are the unit eigenvectors of $\mathbf{H}$ ordered by the associated eigenvalues $\lambda_1 \leqslant \cdots \leqslant \lambda_n$, then the two conditions for a point on a k-dimensional height ridge are $\partial s / \partial y_1 = \cdots \partial s / \partial y_{n-k} = 0$ and $\lambda_{n-k} < 0$. *Valleys* of s are obtained by applying the height ridge definition to the negative field $-s$.

While the height ridge definition is elegant, practice has shown that the generated features need to be filtered. The purpose of filtering is to remove false positives as well as "weak" features. For the case of 1-dimensional ridges, several such filters are known which can be used alone or in combination [13].

One natural criterion for the filtering of raw ridge features is to prescribe a minimum height of the ridge:

$$s \geq s_{min}. \tag{6}$$

In the case of FTLE ridges, the effect of this filter is to suppress ridges with low separation property. The reader is referred to [16] for further details on the influence of this filtering criterion.

A related filtering criterion would be to prescribe a maximum for the second derivative λ_n across the ridge, which results in suppressing regions with too "flat" ridge property:

$$\lambda_n \leq \lambda_{max}. \tag{7}$$

In the case of FTLE ridges this filter is relevant, since the "sharpness" of an FTLE ridge was shown [17] to be a measure for the flux across an LCS, i.e. the quality of an LCS as a flow barrier.

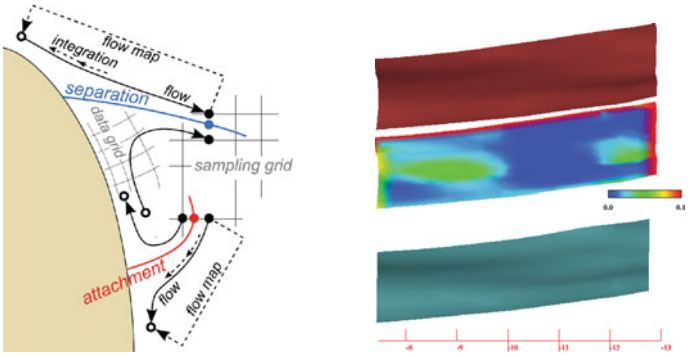
For a reliable and temporally coherent visualization, it is important that criteria such as (6) or (7) are not implemented as binary "pass/fail" filters. By allowing for a certain ratio of exceptions per neighborhood, holes and popping artifacts can be avoided to a large extent. But there is also the problem of noisy ridge extraction results containing many small ridges. These are not necessarily removed by (6) or (7) because the ridge might be sharp and at a high field value. Furthermore, the application of filters can generate additional small ridges. Therefore it is important to filter the ridges also by their size, which requires a connected component labeling of the set of ridges.

3 Motivation

This paper, i.e. the adaptation of the uniform sampling grid to the regions containing ridges, is motivated by the work by Sadlo et al. [15]. The goal is to optimize the computation of time series of FTLE ridges to make the method more applicable in every-day

applications in research and engineering. The increase in efficiency is achieved by restricting the computation to regions that contain the LCS of interest, and, in particular, by exploiting the temporal coherence of unsteady vector fields for the computation of time series of FTLE by advection of the sampling grid.

One of the application goals of this paper is to offer a method for the analysis of unsteady flow separation and attachment. Separation phenomena are the cause of many undesired effects in engineering, such as development of recirculation zones, reduced throughput, reduced lift, vortex generation, lowered mixing, and reduced flow control in general. Flow separation exhibits diverging trajectories in backward time and flow attachment exhibits diverging trajectories in forward time (see Figure 1a). This fact is the reason why separation and attachment lines (or points) are usually accompanied by corresponding LCS and why these processes are amenable to an analysis by FTLE ridges. Shadden et al. [17] have already demonstrated the utility of those ridges for the analysis of unsteady flow separation in their example of flow separation over an airfoil. We also believe that an analysis based on LCS provides a deeper and more precise insight into these unsteady phenomena compared to standard techniques such as stream surface integration or particle tracing.



(a) (b)

Fig. 1. (a) Flow separation and flow attachment. The unstable manifold (blue) attracts the fluid along the boundary and guides it into the interior of the domain whereas the stable manifold (red) guides the fluid in opposite direction. (b) Intake dataset. Comparison of ridge from advected grid (red) and uniform grid (blue) together with ridge from advected grid color-coded by distance to ridge from uniform grid (colored).

4 Method

The proposed method can be subdivided into two parts: one that constrains the sampling grid to filtered ridges of interest at a given time step (or the only time step in case of steady vector fields), described in Sections 4.1 and 4.2, and one that exploits temporal

coherence to speed up the computation of time series of quantities computed from trajectories, such as the FTLE, in the case of unsteady vector fields (Section 4.3). The time series are obtained by variation of the t_0 parameter of the FTLE (compare Section 2.1).

Algorithm 2 describes the methods presented in Sections 4.1 to 4.3 for the case of FTLE ridge extraction. However, it also holds for ridges of other quantities based on trajectories and could be easily modified for the scalar quantity itself instead of its ridges. If the quantity is not computed using local operators such as gradients, larger distortions may be acceptable and hence longer advection times could be used, leading to a further increase in speedup.

4.1 Grid Initialization

In the filtered AMR ridge extraction method [15] the complete domain (or region of interest) is sampled at low resolution and the sampling is adaptively refined in regions containing filtered ridges. Although this results in a speedup compared to a uniform sampling at the finest subdivision level, the method suffers from several drawbacks when applied to quantities that can not be evaluated in a point-wise manner but that are computed using local operators, such as gradients in the case of FTLE ridges. The main problem here is that the value is inherently sampling dependent because the gradient can be underestimated if the sampling is too coarse. Together with a restrictive threshold for the ridge filtering this sometimes results in missed ridges, because they are not detected in the coarse sampling and hence the corresponding regions are not refined which would capture the ridges later on. The remedy is either to use a finer initial sampling, a lower threshold for filtering, or to increase the look-ahead count (Section 3.1 of [15]), all leading to an increased number of samples and hence lowered speedup. See also Section 3.2 of [15] for further information on the implications for quantities based on local operators.

In the present approach one typically avoids sampling the whole domain (or region of interest). Instead, we require initial sampling regions that already capture part of the ridges (cf. Figures 4a and 5a) and adapt the sampling regions to the present ridges (Figures 4b and 5b). This allows to use initial samplings of sufficient resolution and avoids the need for lowered filtering thresholds. In the case of FTLE analysis of flow separation and attachment, possible ways for choosing the initial regions include:

- Definition from special regions of the simulation mesh, e.g. the complete boundary of the domain, or a subset thereof such as the blades of a turbine. These regions are often explicitly available from the simulation file formats.
- Automatic definition by quantities such as "surface divergence" or its local maxima as presented by Tricoche et al. [22].
- Automatic definition by features. A possibility is to extract separation and attachment lines according to Kenwright [11] or Tricoche et al. [22] and to place initial sampling regions around (part of) those.
- Manual identification and definition by preceding interactive exploration using standard techniques such as path line integration or (AMR) ridge extraction [15] of the FTLE in regions of interest. It might seem cumbersome to extract FTLE ridges in a first step with a standard technique, but this can be afforded if the goal is to compute time series of FTLE (Section 4.3).

Algorithm 2 Grid Advection for FTLE Ridges

```
 1: steps: number of steps for the FTLE ridge time series
 2: range: topological neighborhood range around ridges
 3: tolerance: tolerance for RMS of FTLE
 4:
 5: place initial sampling grid
 6: compute all trajectories and compute FTLE
 7: ℛ ← detect ridge cells from FTLE
 8: 𝒩 ← cells in range around ℛ // 𝒩 may contain existing cells and cells to add
 9:
10: // compute frames of FTLE time series
11: r ← 2 // number of advection steps
12: lastResampleStep ← 0
13: for step=1 to steps do
14:     // grid growing
15:     while first iteration at step or grid changed do
16:         // add cells in neighborhood range around ℛ
17:         for all cells c ∈ 𝒩 and not yet in sampling grid do
18:             add c directly if grid regular, or by advection or meshing
19:         end for
20:         compute (or reuse) trajectories and compute FTLE
21:         ℛ ← detect ridge cells from FTLE
22:         𝒩 ← cells in range around ℛ
23:     end while
24:
25:     // grid shrinking
26:     for all cells c of sampling grid do
27:         if c outside domain or c ∉ (ℛ ∪ 𝒩) then
28:             remove c
29:         end if
30:     end for
31:
32:     // grid advection
33:     if step < steps then
34:         advect grid nodes to next time step
35:         compute (or reuse) trajectories and compute FTLE
36:         // resampling
37:         if step − lastResampleStep > r then
38:             resample uniformly, recompute all trajectories and compute new FTLE
39:             RMS ← measure RMS between old FTLE and FTLE on resampled grid
40:             r ← max(1, ⌊r∗ tolerance / RMS⌋)
41:             lastResampleStep ← step
42:         end if
43:         ℛ ← detect ridge cells from FTLE
44:         𝒩 ← cells in range around ℛ
45:     end if
46: end for
```

We require the initial sampling regions and resampled regions (Section 4.3) to be parts of a virtual uniform grid that covers the complete domain. This makes sure that separated grids are consistently sampled and hence can merge (even after advection) when cells are added by the procedure described in Section 4.2.

4.2 Grid Adaptation

This section describes how the initial sampling grid from Section 4.1 is adapted to the filtered ridges (cf. Figures 4b, 4d, 5b, and 5d). To prevent long extraction times in cases where the ridges extend into regions that are of no interest to the user, a region of interest can be defined which restricts the adaptation, possibly leading to truncated ridges.

Grid Growing The first adaptation step is to add new cells to the boundary of the sampling grid where necessary. We define a *ridge cell* to be a cell that has an edge intersected by a filtered ridge according to (6) or (7). Because we aim at results that are identical to those from a uniform sampling, the support range of the Hessian, which is needed for the height ridge extraction, has to be taken into account. If the underlying scalar quantity is computed using a local operator (as in the case of FTLE), its support radius has to be added to that of the Hessian as well. Having the total support range, one needs to add all cells to the grid that are within that topological neighborhood of any ridge cell.

In cases of steady vector fields, where the grid advection from Section 4.3 does not apply, the sampling grid is uniform and adding cells is a trivial procedure. However, if the grid is advected, adding cells is a challenging problem due to the distortion of the grid. Nevertheless, the initial grid is uniform and the grid gets uniformly resampled from time to time. If we need to add a cell to the distorted grid, we simply go back to the last time step where the grid was uniform, add the nodes of the corresponding cell there and advect the added nodes to the actual time step. This makes the cell fit to the desired position. Additionally, the computed trajectories for the advection of the nodes can be reused for computing the quantity (FTLE), resulting in little overhead.

However, if a node of the cell in the uniform grid is located outside of the domain, there is no vector field that could be used to advect it to the desired timestep and position. In this case the cell can be constructed by extrapolation of the grid or any standard meshing technique. The grid growing procedure is iterated until convergence, meaning that each added cell and its neighbors are tested for being a ridge cell and if this is the case, it is attempted to add the cells inside the neighborhood range. This way, the sampling grid grows to the necessary extent.

Grid Shrinking The next step is to remove unnecessary cells from the grid. These are cells that are neither ridge cells nor in the relevant neighborhood of any ridge cell, or cells where one or more nodes are outside of the domain.

4.3 Grid Advection

Lagrangian coherent structures are material lines or material surfaces [7], in other words, they get advected with the flow, such as streak lines (surfaces) and time lines

(surfaces). This would allow to exploit temporal coherence for the generation of time series of FTLE ridges by advection of the extracted ridges. One could compute the ridges only after every n time steps and advect them with the flow in between. However, this would not account for changes of the FTLE during advection. New ridges can originate and existing ones can grow, shrink, or disappear, especially if the ridges are filtered by the FTLE value as in our case. Therefore we propose a different approach: the advection of the sampling grid itself during the advection intervals. This results in a generic method for quantities based on trajectories, not only FTLE.

During advection, a short trajectory has to be computed for each node of the sampling grid to advance it to the next position. The striking advantage is that these short trajectories can be appended to the existing trajectories needed for the computation of the FTLE, making it possible to reuse large parts of the trajectories and hence improving efficiency, see Figures 4c and 5c.

As already mentioned, advection of the sampling grid tends to distort its cells and this in turn tends to affect the computation of derivatives, which are needed for FTLE computation and ridge extraction. Additionally, the FTLE tends to be sampling dependent. All in all this generally leads to artifacts in the extracted FTLE ridges such as deformation, false negatives, and even false positives.

To restrict the artifacts to an acceptable level, the FTLE is periodically resampled on a subset of the virtual uniform grid spanning the whole domain: only those cells of the grid are generated (and the corresponding trajectories are computed) which overlap with the advected grid or which are contained in the region of the initial sampling. An additional strategy is to place the sampling grid outside regions producing high distortion such as wakes and vortices. Although this looks like a compromise, it is often a natural choice to analyze LCS away from disturbing phenomena since they would also distort them, even when uniformly sampled, and hence complicate their interpretation.

Because the flow map is computed by integrating trajectories in numerical vector fields and because of the intricacy of gradient computation on unstructured grids, aside from the difficulty to provide an error measure between the ridges extracted from the distorted grid and those extracted from a corresponding uniform grid, it is generally not possible to provide error bounds regarding the distortion of the grid. Garth et al. [5] measure the error for their subdivision scheme in the flow map. Similarly, we propose to measure the error based on the FTLE, not its ridges, and to use it for triggering the resampling procedure.

The grid is uniformly resampled (recomputing the trajectories and the FTLE) after every r advection steps with an initial value of $r = 2$. After the FTLE has been computed on the resampled grid, the FTLE of the advected grid is interpolated at the node positions of the new grid and the root mean square (RMS) of the difference over all nodes is computed. The RMS is then compared to a user-defined tolerance and a new r is estimated from the RMS and from the tolerance by linearization of the RMS over the advection steps (line 40 of the algorithm). The algorithm then proceeds to the next advection phase. However, the linearization of the error can fail in the sense that r is chosen too large such that after the next r advection steps the RMS exceeds the prescribed tolerance. One solution to this problem is to enforce the tolerance by reducing r (and hence taking back advection steps) until the RMS tolerance is fulfilled. However,

experiments have shown that it is usually sufficient not to enforce the tolerance and instead to prescribe a reduced tolerance, e.g. by 15 percent, to satisfy the intended precision.

To support the user in an appropriate choice of the RMS tolerance and the sampling region, we allow visualization of both sets of ridges, those before and after resampling, or color-coding the former ones by their distance to the latter ones as in Figure 1b, serving as uncertainty information. Another approach is to judge the popping artifacts visually when moving from a time step based on an advected grid to a subsequent time step where the grid was uniformly resampled.

Note that for the analysis of separation, time series of FTLE ridges have to get generated by advecting the grid in positive time direction (Figure 4), whereas for attachment the grid has to get advected in negative-time direction (Figure 5). This is necessary since the ridges are captured at the regions of interest (at the boundaries) and FTLE ridges for attachment approach the boundary in positive time. Hence it is necessary to start with the last time step and to compute them in negative time direction in order to capture all of them at the boundary, even those that separate from the boundary. So finally, all ridges (LCS) that come in contact with the boundary (or region of interest) at any time will get captured.

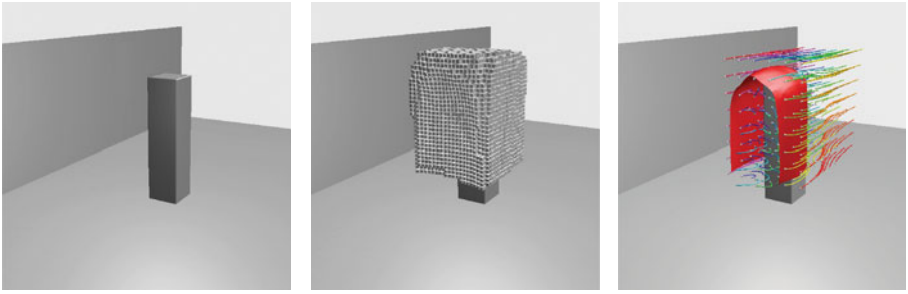
5 Results

In this section the presented method is applied to two unsteady CFD simulations. The first one examines the flow around a cuboid, exhibiting flow similar to a von Kármán vortex street, but in this case the vortices become tilted soon after they detach from the cube. This leads to flow separation behavior that differs from the standard von Kármán vortex street. Secondly, the method is applied to a simulation of an intake of a river power plant. The scope there is a construct that prevents salmon from getting into the runner of the turbine.

5.1 Flow around a Cuboid

This example produces a kind of a von Kármán vortex street. The unsteady flow comes from the right back and follows to the left front (Figure 2a). The main difference to a common von Kármán vortex street is that there is also flow over the "top" face of the cuboid. The flow separation at the cuboid is the subject of analysis in this case. The resulting FTLE ridge (Figure 2c) shows that flow separation is in progress on both sides and on the top of the cuboid. It can be seen that the FTLE ridge separates the vortex street region from the outer flow. However, further downstream the FTLE ridge does not exhibit this property anymore: it crosses the vortices. Time series of FTLE ridges reveal that the separation zones are oscillating consistently with the von Kármán vortex street.

Table 1 shows some performance details for this example. The achieved speedup in this case is only about 2.3. This is due to the relatively short trajectories. The prescribed RMS tolerance was 15.0 and at step 33 this was exceeded by 0.88 percent. There have been 12 advection phases, each performed 5 advection steps in average. Because of the



(a) (b) (c)

Fig. 2. Flow around a cuboid. (a) Geometry. (b) Sampling grid adapted to ridge region and advected. (c) Resulting FTLE ridge with some upstream trajectories (colored) from uniform grid, and their seeds (white spheres).

	uniform	grid advection
grid [nodes]	16399	14220 (step 33)
flow map [s]	13704.87	2944.88
total [s]	13707.92	5800.31
speedup	1	2.36
Figure		2b

Table 1. Performance analysis for the cuboid dataset. 60 steps of grid advection compared to 61 direct evaluations on uniform grid. See also Figure 2b.

	uniform	direct on adapted grid	grid advection
grid [nodes]	8800	5007	3913 (step 39)
flow map [s]	15369.17	9062.73	355.62
total [s]	15374.22	9489.96	1026.81
speedup	1	1.62	14.97
Figure	3a		3b

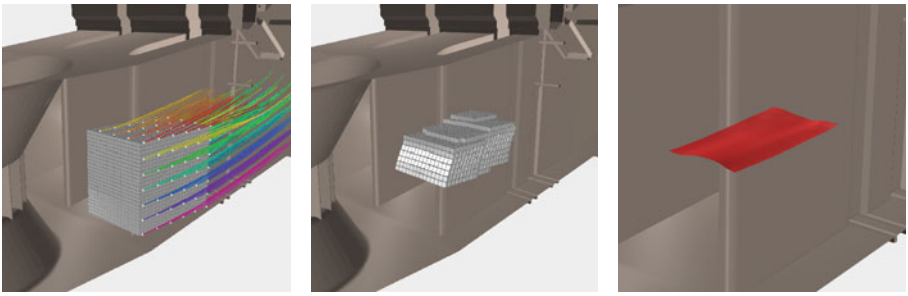
Table 2. Performance analysis for the turbine intake dataset. 100 steps of grid advection compared to 101 direct evaluations (on uniform grid and adapted grid). See also Figures 3a and 3b.

shape of the FTLE ridge and because the initial sampling grid is already well adapted to the FTLE ridge, the expected speedup from the grid adaptation is small and was therefore not measured.

5.2 Intake of a Power Plant

The underlying data of this section is an existing run-of-river plant in the US. All devices shown are installed in the intake of the plant in order to protect juvenile salmon from passing through the runner. The water flow of the unsteady CFD simulation comes from the right back and follows to the left front where it enters the turbine (Figure 3).

The horizontal rods at the right hand side of the image lead the salmon into the vertical channels at the top in the installation. However, these rods produce a noticeable wake in the upper part of the main channel (see path lines in Figure 3a). Additionally, the backflow from the salmon channel (the opening at the top downstream from the rods) also is involved in a recirculation zone at the top wall, located above the sampling grid of Figure 3a. On the one hand, a FTLE ridge was extracted using a regular grid at the confluence of the three main channels (Figure 3a), on the other it was extracted using the presented grid advection method (Figures 3b and 3c). The obtained FTLE



(a) (b) (c)

Fig. 3. Intake of a water turbine. (a) Uniform grid with some of the upstream trajectories (colored) used for FTLE computation, and their seeds (white spheres). (b) Sampling grid adapted to ridge region and advected. (c) Resulting FTLE ridge.

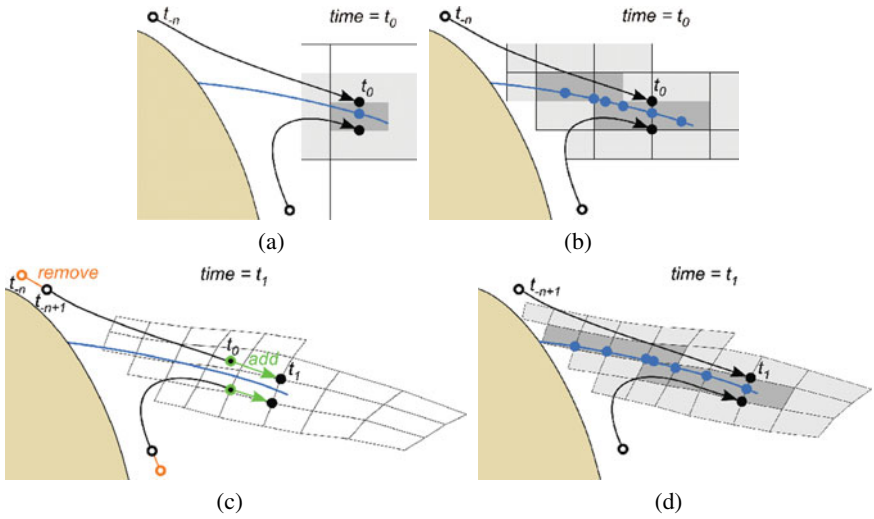


Fig. 4. Grid advection for flow separation. (a) Initial sampling grid. Ridge cells (dark gray) and their neighboring cells (light gray). Cell edge intersected by negative-time FTLE ridge (blue point). Neighborhood range is 1 for illustration purposes. (b) After grid adaptation. (c) After one step of grid advection. (d) After grid adaptation of advected grid.

ridge separates well the fast flow at the bottom of the channel from the slower flow in the upper half of the channel.

Table 2 shows some performance measurements of the presented case. The speedup from the grid adaptation is quite low (1.62) because of the relatively low resolution of the sampling grid and because the sampling region was already quite well adapted to the ridge. The speedup from including grid advection is significantly higher (about

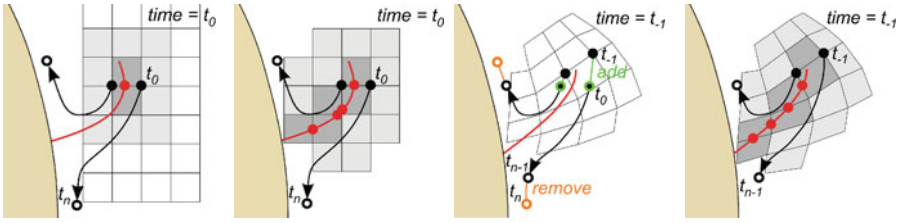


Fig. 5. Grid advection for flow attachment. (a) Initial sampling grid. Ridge cells (dark gray) and their neighboring cells (light gray). Cell edge intersected by positive-time FTLE ridge (red point). Neighborhood range is 1 for illustration purposes. (b) After grid adaptation. (c) After one step of grid advection. (d) After grid adaptation of advected grid.

15) and would further increase with increasing the integration time for the trajectories. The RMS tolerance was set to 0.012 and at step 39 this was exceeded by 14.2 percent, which was the maximum during the 13 advection phases. Figure 1b shows the corresponding distance error of the ridge. On average, 7.7 advection steps were performed per advection phase.

6 Conclusion

We presented a generic method for accelerating the computation (of time series) of quantities based on trajectories, such as FTLE. On the one hand the efficiency is improved by restricting the sampling grid to the phenomena of interest, on the other hand and more important, the computation is accelerated by reusing part of the trajectories, which is made possible by advection of the sampling grid. In the case of gradient-based visualizations, such as FTLE ridges, the quality tends to suffer if the distortion caused by the advection of the grid is high. Therefore, the obtained quality is inferior to evaluations on regular grids or that by Sadlo et al. in terms of quality, but superior in terms of speed if time series of FTLE ridges are computed. A comparison to the approximative method by Garth et al. deduces from a comparison of that method to FTLE samplings on a regular grid. All in all we propopse to use the method at least as a fast preview technique and to use low RMS error thresholds (leading to low acceleration) or even exact methods, such as direct computation on uniform grids or that by Sadlo et al. [15], if exact time series are required. Future work could include local strategies for reducing the distortion of the grid and thus lowering the frequency at which resampling is needed. We would like to thank Sulzer Innotec for the cuboid dataset and VATECH Hydro for the intake dataset. This work was funded by Swiss Commission for Technology and Innovation grant 7338.2 ESPP-ES.

References

1. D. Asimov. Notes on the Topology of Vector Fields and Flows. Technical Report RNR-93-003, NASA Ames Research Center, 1993.
2. G. Bennetin, L. Galgani, A. Giorgilli, and J. Strelcyn. All Lyapunov exponents are effectively computable. *Physical Review A*, 14:2238, 1976.
3. K. Bürger, P. Kondratieva, J. Krüger, and R. Westermann. Importance-Driven Particle Techniques for Flow Visualization. In *Proceedings of IEEE VGTC Pacific Visualization Symposium 2008*, pages 71–78, March 2008.
4. D. Eberly. *Ridges in Image and Data Analysis. Computational Imaging and Vision*. Kluwer Academic Publishers, 1996.
5. C. Garth, F. Gerhardt, X. Tricoche, and H. Hagen. Efficient Computation and Visualization of Coherent Structures in Fluid Flow Applications. *IEEE Trans. Vis. Comput. Graph.*, 13(6):1464–1471, 2007.
6. C. Garth, G.-S. Li, X. Tricoche, C. Hansen, and H. Hagen. Visualization of Coherent Structures in Transient 2D Flows. In *Topology-based Methods in Visualization II*. Springer, pages 1–13, 2008.
7. G. Haller. Distinguished material surfaces and coherent structures in three-dimensional fluid flows. *Physica D*, 149:248–277, 2001.
8. J. Helman and L. Hesselink. Representation and Display of Vector Field Topology in Fluid Flow Data Sets. *IEEE Computer*, 22(8):27–36, 1989.
9. F. Hussain. Coherent structures and turbulence. *Journal of Fluid Mechanics*, 173:303–356, 1986.
10. J. Jeong and F. Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285(69):69–94, 1995.
11. D. N. Kenwright. Automatic detection of open and closed separation and attachment lines. In *VIS '98: Proceedings of the conference on Visualization '98*, pages 151–158, Los Alamitos, CA, USA, 1998.
12. K. J. Lockey, M. Keller, M. Sick, M. H. Staehle, and A. Gehrer. Flow induced vibrations at stay vanes: Experience at site and CFD simulation of von Kármán vortex shedding. In *Proceedings of Hydro2006*, pages 25-28, 2006.
13. R. Peikert and F. Sadlo. Height Ridge Computation and Filtering for Visualization. In I. Fujishiro, H. Li, and K.-L. Ma, editors, *Proceedings of IEEE VGTC Pacific Visualization Symposium 2008*, pages 119–126, March 2008.
14. S. K. Robinson. Coherent motions in the turbulent boundary layer. *Annu. Rev. Fluid Mech.*, 23:601–639, 1991.
15. F. Sadlo and R. Peikert. Efficient Visualization of Lagrangian Coherent Structures by Filtered AMR Ridge Extraction. *IEEE Transactions on Visualization and Computer Graphics*, 13(5):1456–1463, 2007.
16. F. Sadlo and R. Peikert. Visualizing Lagrangian Coherent Structures and Comparison to Vector Field Topology. In *Topology-based Methods in Visualization II*. Springer, pages 15–30, 2008.
17. S. C. Shadden, F. Lekien, and J. E. Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D Nonlinear Phenomena*, 212:271–304, Dec. 2005.
18. K. Shi, H. Theisel, T. Weinkauf, H. Hauser, H.-C. Hege, and H.-P. Seidel. Path Line Oriented Topology for Periodic 2D Time-Dependent Vector Fields. In *Proc. Symposium on Visualization (EuroVis '06)*, pages 139–146, 2006.
19. A. Surana, O. Grunberg, and G. Haller. Exact theory of three-dimensional flow separation. Part I: Steady separation. *J. Fluid Mech.*, 564:57-103, 2006.

20. A. Surana, G. Jacobs, and G. Haller. Extraction of Separation and Reattachment Surfaces from 3D Steady Shear Flows. *AIAA Journal*, 45(6):1290–1302, 2007.
21. H. Theisel, T. Weinkauf, H.-C. Hege, and H.-P. Seidel. Stream Line and Path Line Oriented Topology for 2D Time-Dependent Vector Fields. In *IEEE Visualization*, pages 321–328, 2004.
22. X. Tricoche, C. Garth, and G. Scheuermann. Fast and Robust Extraction of Separation Line Features. In *Scientific Visualization: The Visual Extraction of Knowledge from Data*, Berlin, 2005. Springer.