Parallel Vectors Criteria for Unsteady Flow Vortices

Category: Research

Abstract—Feature-based flow visualization is naturally dependent on feature extraction. To extract flow features, often higher-order properties of the flow data are used such as the Jacobian or curvature properties, implicitly describing the flow features in terms of their inherent flow characteristics (e.g., collinear flow and vorticity vectors). In this paper we present recent research which leads to the (not really surprising) conclusion that feature extraction algorithms need to be extended to a time-dependent analysis framework (in terms of time derivatives) when dealing with unsteady flow data. Accordingly, we present two extensions of the parallel vectors based vortex extraction criteria to the time-dependent domain and show the improvements of feature-based flow visualization in comparison to the steady versions of this extraction algorithm both in the context of a high-resolution dataset, i.e., a simulation specifically designed to evaluate our new approach, as well as for a real-world dataset from a concrete application.

Index Terms—Vortex Feature Detection, Time-Varying Data Visualization

1 Introduction

In this paper we present a solution to the challenge of *feature extraction* when dealing with *time-dependent* simulation data from computational fluid dynamics. We aim at feature-based flow visualization with focus on vortices and their central locations. In an extension of the state of the art we present two new methods for the extraction of *vortex core lines* (aka. *vortex axes*[1]) in *unsteady flow* which are truthful to the time-dependent nature of the extracted features.

A lot of work has been done in the field of feature extraction from *steady*/*time-independent* flow data, especially with focus on vortices. In the context of time-dependent flow previous work focussed on extracting features from individual time steps by interpreting the flow data as a "stack" of steady flow fields (one per time step) and by applying extraction methods for steady flow data accordingly. The time-dependent nature of these features was taken into account by connecting them afterwards over time, e.g., by tracking. In Section 2 we go into more detail with respect to related previous work.

It is favorable to inherently consider time already during feature extraction and not separately in a second step. Doing so, we find ourselves aligned with others (such as Hussain already in 1983 [10]), who demand the joint consideration of space and time when investigating features in time-dependent flow data. Accordingly, we propose to formulate the extraction criterion in a way that temporal derivatives are used for the local characterization of vortices and not only the Jacobian of the flow. This is synonymous to considering pathlines for feature extraction from unsteady flow instead of streamlines. Even though we experienced in exchange with colleagues, reviewers, and others that this extension is easily and quickly considered to be logical and straight forward, the results improve more than expected.

Very often, flow phenomena such as gas flow during combustion or air flow around a vehicle are time-dependent in their nature and steady representations are just an approximation. Datasets with time-independent flow are useful for domain experts as they provide information, about general or large-scale characteristics of the flow, at a relatively low cost in terms of dataset size, simulation time, as well as analysis time. However, we still observe a clear trend towards more unsteady flow data in scientific as well as in commercial applications mostly because of better results, especially when doing a more careful or detailed flow analysis, and also because of the availability of increased computing and storage resources.

Accordingly, we consider it important to explicitly demonstrate that feature extraction based on time, is not only logical to do, but indeed yields better results. In certain cases, we can even observe that the traditional, streamline-oriented approaches lead to displaced "features". Furthermore, we can find an improved agreement of the new approach with physical extraction schemes such as the low-pressure assumption in the midst of vortices (no need for a correction step). In Sections 3 and 5 we exemplify our point by means of selected cases both in analytic and computed form. The need for a new approach is demonstrated as well as the gain through improved results. The contributions of this paper include two mathematical examples that model real world problems. Based on the results of these examples we derive simple modifications of existing vortex core line detection algorithms to extend them to the unsteady flow domain. Real world applications where the original approaches fail are presented and it is shown that the results improve using the modified approach. Finally a numerical study evaluates the impact of time-derivative estimation on the feature extraction process. In the appendix we give details on implementation details for unstructured grid data.

[1] In other fields, e.g., in fluid mechanics, *vortex cores* are considered to be of regional type (and not of line type). In this paper we use the term *vortex core line* for line-type curve features which represent central locations in vortices.

2 Related Work

Feature-based flow visualization has been an active field of research for many years and it is beyond the scope of this paper to provide a comprehensive discussion of all of this work – we refer to Post et al. [18], who published an extended overview recently. In this section we focus on selected pieces of previous work, which are tightly related to our new approach.

The algorithms, which we take as a basis for developing our new approach, are the proven method for extracting vortex core lines from steady flow data by Sujudi and Haimes [24] as well as the related, higher-order method by Roth and Peikert [21]. Both approaches were successfully applied in many cases, especially when dealing with time-independent data. As such, we consider them as a strong starting point for approaching the case of unsteady flow data. To do so, we adopt the principle of the parallel vectors operator [16] for extracting the vortex core lines in conjunction with modified extraction criteria that are based on temporal derivatives.

Reinders et al. [19] use a graph view to show the development of flow features over time and to indicate events such as birth, death, and annihilation of features. Bauer et al. [2] discuss the tracking of vortices in scale space, which improves the consideration of important features. Garth et al. [6] show the movement of singularities relative to an axis, which is of special importance compared to the others. Theisel and Seidel [25] introduce the concept of the feature flow field and use it to improve feature tracking: the paths of the critical points are tracked as the streamlines of a new vector field, i.e., the feature flow field constructed from the original vector field.

The idea of considering pathlines when analyzing time-dependent flow data is not new as such. Theisel et al. [26] present a pathline-oriented approach to extracting the topology of 2D time-dependent vector fields – similar to a streamline-based approach, they distinguish features according to attracting, repelling, or saddle-like behavior. Haller [7] describes vortices through the stability of manifold structures which are related to fluid trajectories, i.e., pathlines, and ex-

tracts vortex regions in unsteady flow data based on this information. Sadlo and Peikert [22] extract ridges from 3D finite-time Lyapunov exponents (FTLE) for the extraction of Lagrangian coherent structures (LCS). And Garth et al. [5] present a method for the direct visualization of 2D FTLE information which results in expressive images of time-dependent flow.

In general, we observe a new motivation in the field to approach even very complex cases in 3D time-dependent flow visualization. Peikert and Sadlo [17] discuss feature-based visualization for the investigation of vortex rings and vortex breakdown bubbles in recirculating flow, and Tricoche et al. [27] describe a slice-based visualization for understanding intricate flow structures where the slices are placed orthogonal to trajectories of the flow.

Another interesting class of approaches are physical criteria (instead of geometric ones) for feature extraction. Banks and Singer [1] propose a method to find vortex core lines based on a predictor/corrector method that steps through the field in the direction of the vorticity vector. At each step the normal plane is constructed and the point is reset to the nearest local pressure minimum. Jankun-Kelly et al. [11] present an improvement of this approach using a function fitting procedure to locate the extreme values, stepping along the real eigenvector of the velocity gradient. Stegmaier et al. [23] present an algorithm that combines the λ_2 method of Jeong and Hussain [12] with the predictor/corrector method of Banks and Singer. For growing the skeleton they step in the direction of the vorticity vector. In this context of physical approaches, several more methods have been presented, e.g., the Q-criterion of Hunt et al. [9], also known as the elliptic version of the Okubo–Weiss criterion by Okubo [15] and Weiss [29], or the extension of considering acceleration terms by Hua et al. [8], which includes temporal derivatives and expresses the feature extraction process from the Lagrangian perspective.

In an upcoming paper, Weinkauf et al. [28] approach the question of vortex core line extraction in a similar fashion. For finding "swirling particle cores" they analyze the real eigenvector of the velocity gradient and the acceleration vector. Even though they arrive at a similar extraction method, they reason differently (and use other, related vectors for their approach). Our solution, as presented in this paper, is based on physical principles, resulting in a corresponding modification of existing algorithms. The swirling particle cores method is based on the space-time framework and builds primarily on a geometric approach. In future work we plan to evaluate and compare the two approaches thoroughly. In this paper, we are also able to demonstrate our work in the context of an application, compare it to other, simulated quantities related to vortices, and show its good numerical behavior regarding time step width in the data set.

3 Analytic Considerations

In the following, we discuss two analytic examples which can be considered as models for related phenomena in actual flow data. This way we can concentrate on the demonstration of the need for a new approach. Looking at analytic cases we can avoid issues such as aspects related to sampling and reconstruction. This approach is analogous to the work of others who use analytic examples for motivation and for demonstration [24, 21, 7].

3.1 A Tilting Vortex

To construct our first synthetic vortex example, we aimed at an as simple as possible flow model that still can demonstrate the difference between a streamline- and a pathline-based approach. To avoid a simultaneous discussion of whether our approach is Galilean invariant we decided to go for one simple vortex which tilts over time.

Accordingly, we specify our flow model as

$$\mathbf{u}(x,y,z,t) = \begin{pmatrix} -y+tz \\ x-tz \\ z \end{pmatrix}.$$

The vortex in $\mathbf{u}$ is linearly strained in the z-direction and contains a tilt which increases over time. Considering $\mathbf{u}$ in just one time step $t = t_a$

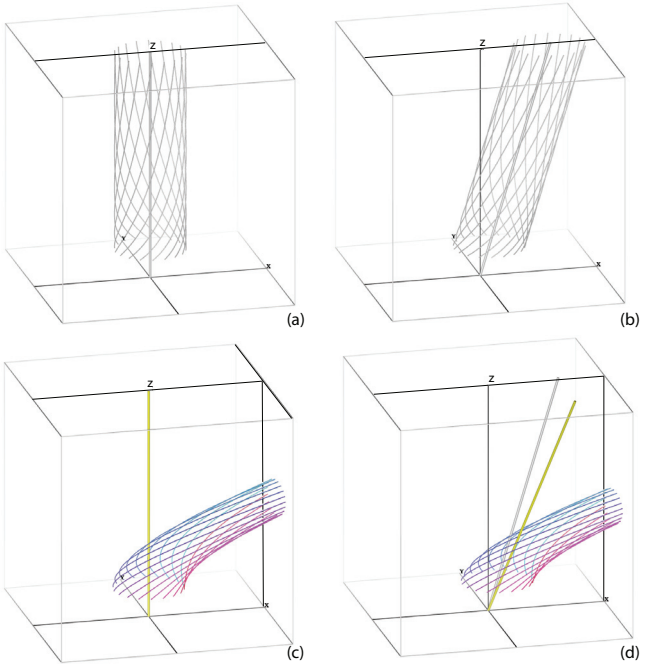

Fig. 1. A synthetic example of a tilting vortex is shown before the tilt (at the left) and a bit later (on the right). The top row shows the vortex core line (grey tube) according to Sujudi and Haimes [24] and several streamlines – the tilt into the x-direction is obvious. The bottom row shows pathlines (in color) which exhibit an additional tilt towards the viewer (yellow vortex core line).

and analyzing its – in all locations equal – Jacobian

$$J|_{t=t_a} = \begin{pmatrix} 0 & -1 & t_a \\ 1 & 0 & -t_a \\ 0 & 0 & 1 \end{pmatrix},$$

by considering the only one real eigenvector $(t_a, 0, 1)^\mathsf{T}$ of this matrix we observe a virtual[2] rotation of the instantaneous flow field around an axis which is aligned with this vector and which tilts into the positive x-direction. In the top row of Fig. 1 this situation is illustrated for two time steps $t_a = 0$ (left) and $t_a = 0.3$ (right).

We abandon the restriction to only consider the flow in just one time step and see a different picture (bottom row of Fig. 1). In addition to the above mentioned x-tilt, there is another tilt towards the viewer. The corresponding vortex core line illustrated in yellow in Fig. 1 (d) reflects this additional y-tilt.

The design of this flow model allows to analytically find explicit solutions for stream- and pathlines. If we first consider just one time step $t = t_a$, we derive the streamline for seed location $(x_0, y_0, z_0)^\mathsf{T}$ in parameterized form as

$$\begin{aligned} x(\tau) &= (x_0 - t_a z_0)\cos(\tau) - y_0 \sin(\tau) + t_a z_0 e^\tau, \\ y(\tau) &= (x_0 - t_a z_0)\sin(\tau) + y_0 \cos(\tau), \\ z(\tau) &= z_0 e^\tau. \end{aligned}$$

The $t_a z_0 e^\tau$ term in the x-component of streamlines reflects the above discussed x-tilt. In the y-component of streamlines we do not see any corresponding tilt term.

Considering pathlines next, we derive the following solution (now

[2] We consider this rotation as "virtual" as it only exists for an infinitesimal short moment of time – the vortex axis which is detected locally in time does not yield any tightly related finite-time rotation of particles around this axis.

parameterized with time t):

$$x(t) = (x_0 + \tfrac{1}{2}z_0)\cos(t) - (y_0 + \tfrac{1}{2}z_0)\sin(t) + (t - \tfrac{1}{2})z_0 e^t,$$
$$y(t) = (x_0 + \tfrac{1}{2}z_0)\sin(t) + (y_0 + \tfrac{1}{2}z_0)\cos(t) - \tfrac{1}{2}z_0 e^t,$$
$$z(t) = z_0 e^t.$$

Now we see corresponding tilt terms in both the x- and the y-components of the pathlines and the vortex axis is found to be along the vector $(t, -t, 1)$.

3.2 A Rotating Vortex Rope

As a second example, we construct a simple synthetic model of a rotating vortex rope that has characteristics which are related to an important flow phenomenon in the draft tube of large water turbines. To start, we consider the flow field

$$\mathbf{u} = \begin{pmatrix} -(y - y_1)\cdot s \\ (x - x_1)\cdot s \\ 1 \end{pmatrix}.$$

For the degenerated case of $x_1 = y_1 = 0$, this simply is a rigid rotation about the z-axis. Assuming that the points (x_1, y_1, z) lie on a helix with radius R and pitch $\frac{2\pi}{k}$, which rotates around the z-axis with angular frequency ω and phase 0, i.e., with

$$x_1 = R\cdot\cos(kz + \omega t) \text{ and}$$
$$y_1 = R\cdot\sin(kz + \omega t),$$

we get a rotating vortex, i.e., a time-dependent flow field as desired – see Fig. 2 for selected stream- and pathlines. Note, that we assume $|k + \omega| < s$ to ensure that the structure of the helix dominates the rotation about it.

Based on this model, we can analytically derive several variants of vortex core lines (according to different extraction schemes). In all cases we obtain a helix with the same pitch, frequency, and phase, but with different radii. See table 1 for an overview of the results.

The employed methods are three state of the art approaches for steady flow data: the method proposed by Levy et al. (curl parallel to velocity [13]), the one by Sujudi and Haimes (parallel first and second derivatives of streamlines [24]), and the higher-order method by Roth and Peikert (parallel first and third derivatives of streamline [21]). We apply them to the flow data of individual time steps as discussed above.

We contrast these results with those of our new approach, i.e., the unsteady extension of Sujudi and Haimes's (as described in Section 4.1) and the unsteady version of the higher-order approach (as described in Section 4.2). We see that the traditional approaches miss the rotation of the vortex rope (missing '$+\omega$' terms in all cases), since it obviously cannot be detected from considering an individual time step only.

We also compute a correct vortex core line for this unsteady flow by using a symmetry argument. On each slice orthogonal to the z-axis ($z = z_{\text{const}}$), there is just one point $(x, y, z_{\text{const}})^{\mathsf{T}}$, with

$$x = \frac{R\cdot\cos(kz)}{1 - (k+\omega)/s} \quad \text{and } y = \frac{R\cdot\sin(kz)}{1 - (k+\omega)/s},$$

such that a particle which is released from this point at time 0 moves along a pathline of exact helical shape. Particles that are released from any other location yield pathlines of more complicated geometry (Fig. 2). In this case, we see that the material line (time line), which consists of all of these special particles, coincides with the correct vortex core line. This curve has the same radius as the helical pathlines, but exhibits a different pitch of $\frac{2\pi}{k}$ vs. $\frac{2\pi}{k+\omega}$. We note, however, that the fact that the vortex core line also is a material line is specific to this example and does not generally hold for arbitrary cases.

By comparing the different radii from table 1 with the correct solution and by considering the geometric series $(1-p)^{-1} = (1 + p + p^2 + \ldots)$, here with $p = (k+\omega)/s$, we can see a nice alignment of our new approach with the correct solution. The modified variant of the approach by Sujudi and Haimes is the first-order approximation of the correct solution and the modified variant of the higher-order approach is its second-order approximation.

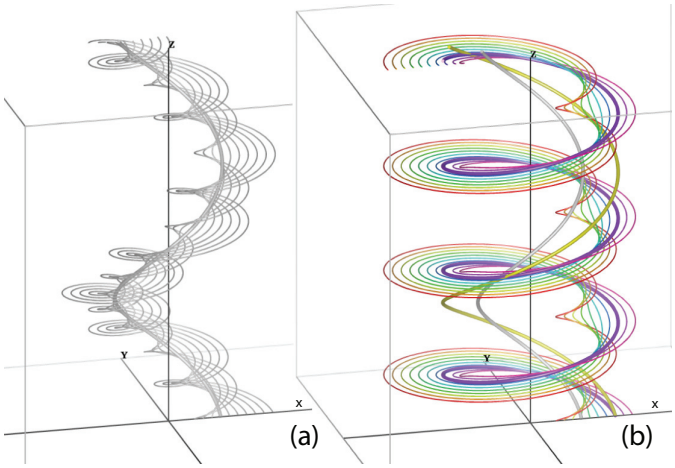

Fig. 2. Streamlines and pathlines in a model of a rotating vortex rope. (a) The vortex core line based on streamlines (according to Sujudi and Haimes [24]) is shown as a thick grey tube (it is the only grey line which also is a helix). (b) The vortex core line based on pathlines (shown in yellow on the right) has the same pitch but a larger radius (it is the only helical pathline, shown in thick magenta).

	streamline-based	pathline-based (new)
Levy et al.	$(1+\frac{k}{2\cdot s})R$	
Sujudi & Haimes	$(1+\frac{k}{s})R$	$(1+\frac{k+\omega}{s})R$
higher-order	$(1+\frac{k}{s}+(\frac{k}{s})^2)R$	$(1+\frac{k+\omega}{s}+(\frac{k+\omega}{s})^2)R$
correct		$(1-\frac{k+\omega}{s})^{-1}R$

Table 1. Different extraction schemes all result in helical vortex core lines, but with different radii. We compare the results for the algorithms of Levy et al. [13], Sujudi and Haimes [24], the higher-order method by Roth and Peikert [21], and an analytically determined correct variant.

The deviation of the Sujudi-Haimes lines from the correct vortex core lines is the phenomenon first observed in the "bent helix" example [20], and it is due to the combination of a weakly rotating vortex and a strongly curved vortex core line. The error becomes negligible if $|k+\omega| \ll |s|$, i.e. if the sum of the spatial and the temporal frequency is much smaller than the parameter s controlling the swirl around the vortex core line. The higher-order method yields an additional term of the Taylor series in this example.

We have seen that the extension to unsteady flow for both methods results in improved results in comparison with the time frozen analysis of vortex flow features. To understand what is happening with unsteady vortices it is necessary to extend the steady versions of the vortex extraction criteria.

4 Pathline Geometry Based Feature Detectors

We can generalize existing feature extraction algorithms to unsteady flow data by replacing streamlines with pathlines in the underlying model. This way they remain unchanged for steady flows.

4.1 Sujudi-Haimes

In this section we modify the approach by Sujudi and Haimes [24] to include time derivatives.

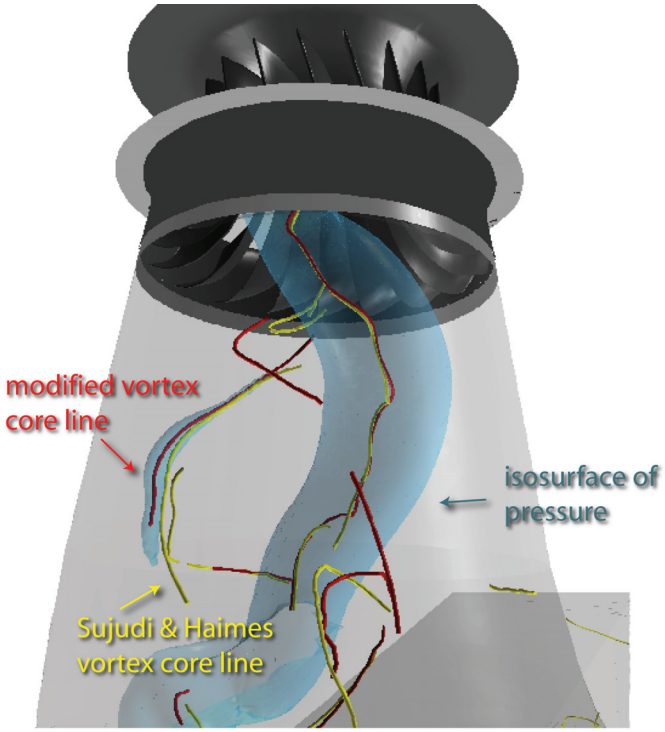

Fig. 3. For the vortex rope in the depicted dataset, iso-values of pressure give good insight on where the vortex core line is located. We can clearly see how the yellow core line (extracted using the classical approach of Sujudi and Haimes [24]) deviates from the center of pressure isosurface. The modification to time derivative aware extraction of the vortex coreline improves the results visibly.

4.1.1 Original Definition

In the original definition the first step is to compute the eigenvalues of $\nabla\mathbf{u}$ per tetrahedral cell. Only cells where a pair of complex eigenvalues exists are further processed. The existence of two complex eigenvalues is determined by the discriminant of the characteristic polynomial [4].

The next step is to compute the single real eigenvector $\varepsilon_\mathbf{r}$ for the candidate cells to extract the local direction of the vortex core line. In the final step the algorithm searches for locations where $\varepsilon_\mathbf{r}$ is parallel to $\mathbf{u}$. Linear interpolation is used between the nodes of a grid cell when searching for parallel locations. A modification in order to get connected lines instead of disjoint straight line segments is to estimate velocity gradients per node and compute parallel positions on cell faces.

4.1.2 Equivalent Definition

The eigenvector computation required by the original method is quite expensive. A more efficient method [16] is to compute the matrix-vector product $\mathbf{a_s} = (\nabla\mathbf{u})\,\mathbf{u}$ instead. Given that $\varepsilon_\mathbf{r}$ is the only real eigenvector of $\nabla\mathbf{u}$, it is parallel to $\mathbf{u}$ exactly if $\mathbf{a_s}$ is. Hence, Sujudi-Haimes vortex core lines can be equivalently defined as the locus of points where $\mathbf{u}$ and $\mathbf{a_s}$ are parallel, restricted to points where the velocity gradient has a pair of complex eigenvalues. In this context, two vectors are said to be parallel also if one or both of them are zero.

4.1.3 Modification for Unsteady Flow

The original formulation of the Sujudi-Haimes criterion is expressed in terms of the velocity field and its gradient tensor field. Using this formulation we cannot include the time derivative information since these quantities are the same for steady and unsteady flow. In contrast, the parallel vectors formulation allows for a different extension to unsteady flow. The vector $\mathbf{a_s} = (\nabla\mathbf{u})\,\mathbf{u}$ can be viewed as the steady case of the acceleration vector

$$\mathbf{a_t} = D\mathbf{u}/Dt = (\nabla\mathbf{u})\,\mathbf{u} + \partial\mathbf{u}/\partial t$$

of a particle. An obvious modification is now to use the true acceleration vector instead of the vector $\mathbf{a_s}$, i.e. to look for points where $\mathbf{a_t}$ and $\mathbf{u}$ are parallel. Besides the justification as being the natural extension to unsteady flow, this modification is also backed up by the following observation.

Sujudi-Haimes vortex core lines can be defined in a third equivalent way, namely as the locus of zero streamline curvature, again constrained to points where the velocity gradient has a pair of complex eigenvalues. The equivalence is shown as follows. The curvature of a curve with (time) parameter is $\kappa = \|\dot{\mathbf{x}} \times \ddot{\mathbf{x}}\|/\|\dot{\mathbf{x}}\|^3$ where the dots denote temporal derivatives. For a streamline, $\dot{\mathbf{x}} = \mathbf{u}$ and $\ddot{\mathbf{x}} = \mathbf{a_s}$, so the streamline curvature is zero exactly where the Sujudi-Haimes criterion is met. For a pathline, $\ddot{\mathbf{x}}$ is $D\mathbf{u}/Dt$ so the pathline curvature is zero exactly where the modified Sujudi-Haimes criterion is met.

In principle, the zero curvature points of streamlines or pathlines could be computed to yield vortex core lines according to the original or modified Sujudi-Haimes criterion. However, numerical integration and curvature computation are too expensive operations to make this a practical alternative to the parallel vectors method.

It was a long standing open question from our application partners why the vortex core lines resulting from the original algorithm of Sujudi and Haimes very often exhibit a small phase-shift in relation to regions of low pressure. Therefore it is a common approach to do a correction step towards pressure minima when extracting vortex core lines [1, 11]. In Figure 3 we can see that the yellow vortex core lines extracted using the eigenvector method are shifted away from the center of the pressure isosurface. Using the pathline based extraction approach we arrive at a solution located at the pressure minima without a correction step. Therefore we can assume that the deviation in the unmodified approaches results from not taking the temporal derivative into account.

4.2 Higher Order Vortex Core Lines

In this section we modify the higher order approach to work on pathlines.

4.2.1 Original Definition

Roth and Peikert [21] present an extension of the vortex extraction approach by Sujudi and Haimes to bent vortices. The eigenvector is based on a straight line model for the vortex core line. In real world data sets we can find many types of bent vortices though. Common types are hairpin, horseshoe, and ring shaped vortices. Roth and Peikert showed [20] that the eigenvector method introduces an error as soon as the vortex is bent.

To overcome these drawbacks we can weaken the conditions on a vortex core line such that we can detect bent vortices as well, but the amount of false positives will increase significantly. It is not possible to model a curved vortex based on linear fields, therefore one has to take into account higher-order derivatives when searching for vortex core lines. The second derivative following a particle in a steady velocity field is $\mathbf{b_s} = (\nabla\mathbf{a})\mathbf{u}$.

Based on the torsion of a parametric curve in $\mathbb{R}^3$ we can relax the condition on vortex core lines such that torsion is zero and that zero torsion is preserved as well as possible when following the streamline. The extraction algorithm is based on the fact that for the bent vortex model the vector $\mathbf{b_s}$ at the vortex core line is not only restricted to the $<\mathbf{u}, \mathbf{a_s}>$ plane but that the best choice is to require that $\mathbf{b_s}$ is parallel to $\mathbf{u}$. Thus, we can state the following definition for a vortex core line: the vortex core line is the location of all points where $\mathbf{b_s}$ is parallel to $\mathbf{u}$.

4.2.2 Modification for Unsteady Flow

The problems observed for curved vortices in steady flow data [21] obviously extend to curved vortices on unsteady flow data. In Section 3.2 we have seen that the modified version of the higher order model

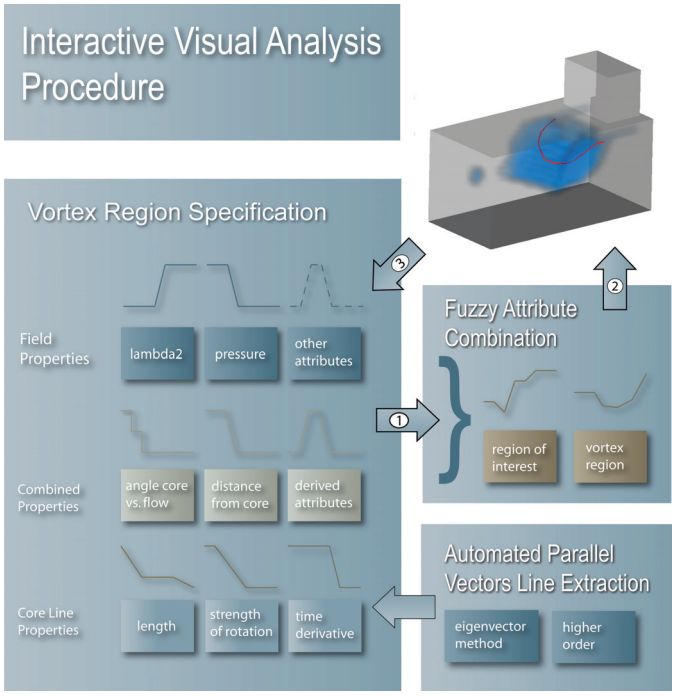

Fig. 4. Interactive visual analysis with vortex core line extraction. After the vortex core lines are computed we use interaction to remove false positives. (1) The user can interactively specify the volume of interest in attribute views to select attribute ranges of interest and another set of attribute selections that control the vortex region. (2) The selected region of interest is visualized by volume rendering (in this example the volume selection is defined by $\lambda_2 < -100$) and the vortex region controls which line segments are visible (we have selected regions that have both complex velocity gradient eigenvalues and negative λ_2 values). (3) From the vortex core lines we can derive additional attributes such as an attribute measuring the distance from the core line for further analysis.

will reproduce the correct vortex core line of the bent time dependent model if we ignore the terms of higher order in the Taylor expansion. Therefore, we can use the parallel vectors operator to apply the higher-order approach to unsteady flows.

A criterion based on zero curvature in principle searches for straight vortex core lines. The line that is classified as the vortex core line by the parallel vectors approach of the previous section can deviate to some extend from this restriction. But for strongly bent vortices the result will show the same inconsistencies as observed for streamline based geometries. For the higher order vortex core line detection algorithm the required modification is therefore to replace the vector $\mathbf{b_s}$ by the actual jerk vector (rate of change of acceleration) $\mathbf{b_t} = \mathbf{D^2u}/\mathbf{Dt^2}$. See Figure 7 for an example.

4.3 Interactive Vortex Core Line Extraction and Filtering

Both the eigenvector method and the higher order method produce many line segments that cannot be considered as vortex core lines. For this reason we use the interactive visual analysis features of the SimVis framework to extract the meaningful vortex core lines. This way we get confidence in the extracted vortex core lines and can improve their quality. Here we rely on smooth expressions of vortex detectors to select the vortex core lines of interest [3]. The other way round we use the extracted core lines to derive other attributes in the data. Figure 4 illustrates this approach.

To our knowledge there is no fully satisfying approach to extract only the relevant vortex core lines automatically from the data. The interactive multi-field approach of SimVis handles this problem using visual analysis. To be able to do this we modify the parallel vectors algorithm slightly:

1. Generate additional field $\mathbf{a_t}$ or $\mathbf{b_t}$ (see Section 4.1.3, Section 4.2.2 and Appendix A).
2. Compute closed parallel vectors lines without additional criteria (see Section A.3).
3. Use interactive region of interest specification to extract correct subsections of the lines (see Figure 4).

The delta discriminant used as an additional criterion both by the method of Sujudi and Haimes and the higher order method was introduced by Chong et al. [4]. This physics-based criterion does not take into account the time-dependent components of the flow. Nevertheless physics-based criteria such as delta, Q, and λ_2 are often directly applicable to unsteady flow, when it is possible to derive them from instantaneous properties of the flow. The delta criterion is prone to finding false positives in large regions of the flow (e.g. in the turbine dataset it is true almost everywhere). In our experience it has shown to reduce the number of spurious solutions to use additional vortex core region detectors in combination with the delta criterion. Another type of additional criteria includes information derived from the vortex core line [16]. Examples are the angle between flow and vortex core line, number of core line segments or vortex strength. These are difficult to tune optimally. By combining multiple vortex region criteria as suggested in [3] we can avoid criteria involving the extracted vortex core line.

Building on the information we get from the extracted vortex core lines, we get access to a whole new type of information that we can use in further analysis steps. To include information on the vortex core line we derive for each cell an attribute that measures the distance from the final vortex core line in a simple breadth-first traversal starting with cells that contain a vortex core line segment.

5 Application Study - Engine Datasets

We have implemented the presented vortex core line detection algorithms in the SimVis framework [31] and applied it to two engine datasets to verify the approach on real world data. For these datasets we have found that using the Green-Gauss approach for computing gradients gives better results than a least-squares approach (see Appendix A).

The first dataset results from a simulation of the compression and combustion phase in the combustion chamber of a standard engine model. In Figure 6 we can see the vortex core lines based on the original and the modified versions of the parallel vectors criteria. Obviously the results differ significantly and one of the vortex core lines is not extracted at all using the original algorithm.

The second dataset is a high-performance two-stroke engine dataset, which contains the complete simulation results from the injection and the combustion of fuel during one crank revolution. The engine geometry is shown in Figure 8. Table 2 shows a comparison of the datasets discussed in the following sections.

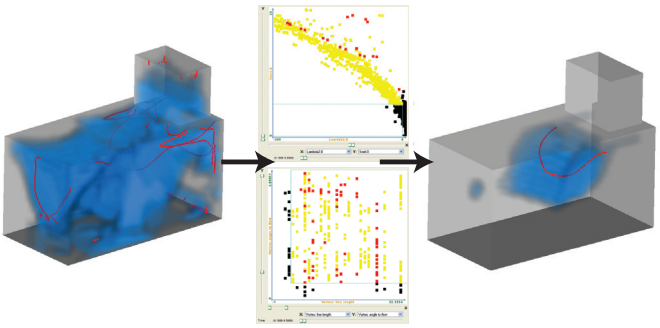

Fig. 5. Multiple views and brushing allow the user to apply vortex core line filter rules interactively. Starting with a large number of spurious solutions we can select the main part of the largest vortex core line by applying two brushes in two scatterplots of additional flow attributes.

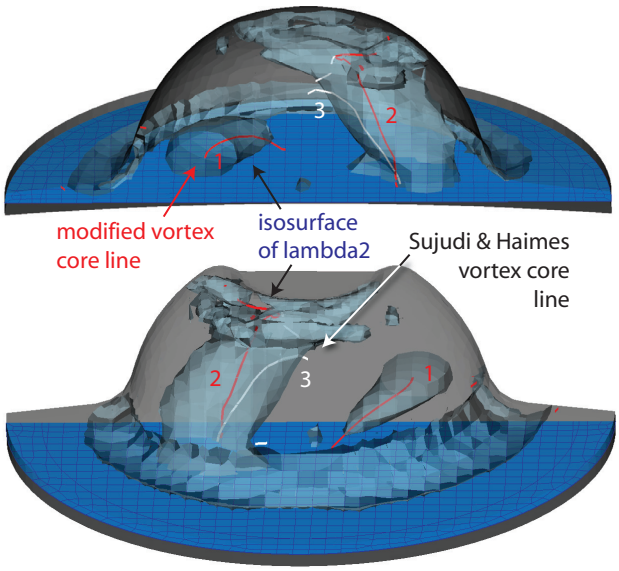

Fig. 6. We compare the vortex core lines found by the original method of Sujudi and Haimes [24] and the modified version. Two views of the same timestep show the benefits of the modification. Both views show the same vortex core line and isosurface. (1) In one case the original method does not detect one vortex core at all. (2) The time-aware modification traverses the full length of the vortex core and continues into the region of strong turbulence at the top of the cylinder. (3) The original vortex core line leaves the core region of the vortex and vanishes in a substantial portion of the vortex region.

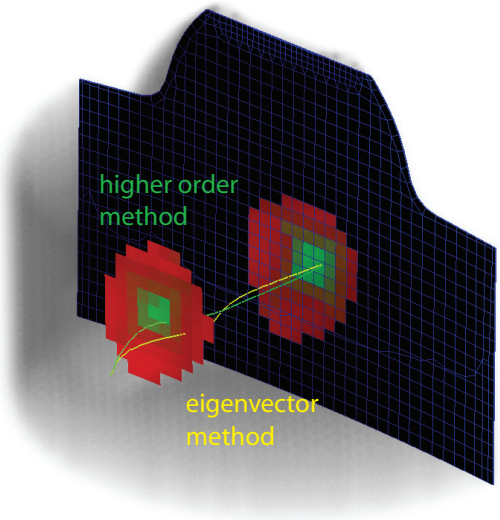

Fig. 7. In this early timestep of the combustion chamber dataset we can see that the extracted vortex core lines for the modified version of the eigenvector method and the modified higher order method differ at the weakest part of the vortex. The cutting plane with color mapped to pressure shows that the modified method of Sujudi and Haimes fails to detect the exact core line of the vortex in this case.

5.1 Impact of Time-derivatives

The question remains whether and where the time derivative information has significant impact on the vortex core line extraction results. In the engine datasets we have found the vortex core lines extracted by the modified and the unmodified methods to be similar but shifted for most timesteps. But in Figure 6 we can observe that in a timestep shortly after ignition the vortex core line based on a_s and the vortex core line

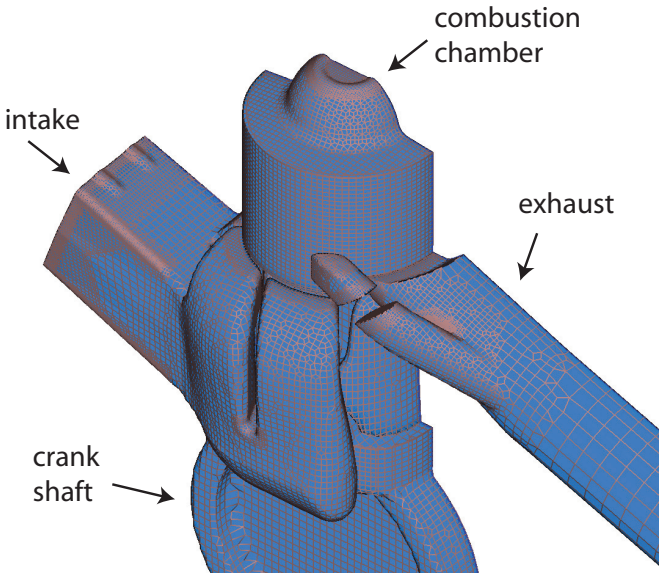

Fig. 8. Overview of the geometry of the two-stroke engine dataset.

cell type	comb. chamber	two-stroke eng.	T-junction
tetrahedra	40	1156	46792
hexahedra	8493 – 23877	129658 – 148247	0
prisms	483 – 2188	11849 – 13241	125960
pyramids	214 – 428	6505	0
Timesteps	48	91	1570

Table 2. Comparison of the datasets evaluated in the application study and for the numerical evaluation. Since the grids of the engine datasets vary over time the number of cells changes accordingly.

based on a_t can differ significantly. This is due to the strong impact of the time derivative in these time steps. To illustrate the close correlation between these two vectors in early timesteps and the large impact of the time derivative after ignition we show the magnitudes of the vectors normalized with mean and standard deviation in scatterplots (see Figure 9). Very often the timesteps that include large changes over time are critical for the application. They have vital impact on mixing, material wear and engine performance and therefore the analysis benefits from improving vortex core line extraction in these time steps.

5.2 Equivalence Ratio

One key attribute that is related both to emission and engine performance is equivalence ratio (ER), which is the relation between fuel and air within a volume cell. It is crucial that ER lies in the optimal interval between 0.7 and 1.4 for most fluid cells at the moment of ignition. The mixing process happens at earlier time steps during compression when the influence of the time derivative is less than after ignition. Even though the difference between the core lines generated by the modified version is smaller it is still not negligible. In Figure 10 (a) we show iso values of the λ_2 vortex detector and concentrate on the vortex core lines detected for this vortex. In the center of the combustion chamber of the two-stroke engine we can see the large vortex region that plays a central role in the mixing process. The question in this example is, why the vortex core region is not of tubular shape. The second vortex core line (3) is not detected by the original

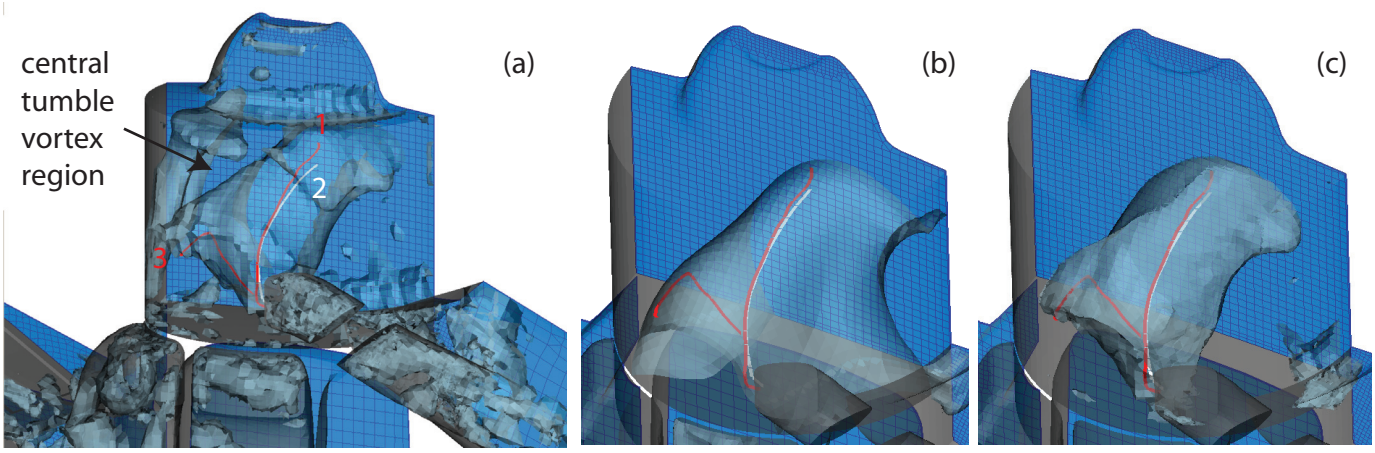

Fig. 10. We compare the computed corelines with respect to λ_2 and equivalence ratio. (a) The modified algorithm detects two vortex core lines (red) whereas the original version only detects the main vortex core line (white). (b) An isosurface of equivalence ratio at 0.7 containing the region of optimal mixing. (c) The surface containing the region of equivalence ratio of 0.5 and $\lambda_2 \leq -1000$.

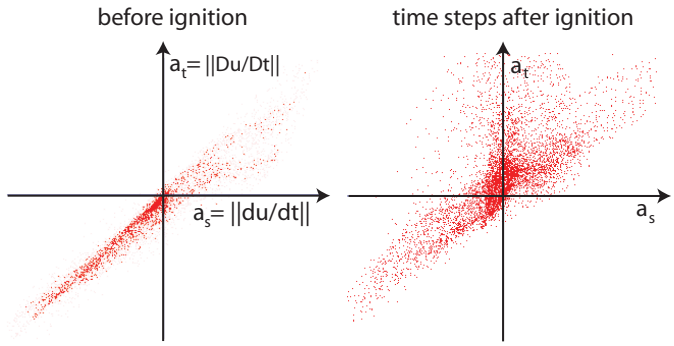

Fig. 9. Comparison of acceleration vector magnitudes: The scatterplots show that the magnitudes of the two variants of the acceleration vectors can differ significantly in the crucial timesteps, i.e., timesteps of major changes over time, after ignition (we have normalized the magnitudes such that the center of gravity corresponds to the origin).

approach. Combining (1) and (3) we can gain insight into the controlling skeleton of the main tumble vortex. In Figure 10 (b) we can distinguish the regions of sub-optimal and optimal to very high concentrations of fuel at iso value surface of 0.7. The bend part of vortex core line (3) closely follows the boundary of this region. In Figure 10 (c) the surface describes the boundary of the region defined by slightly sub-optimal to high mixing and high λ_2 values. The core line generated for this vortex with the original vortex core extraction method (1) and the modified approach (2) are similar and both traverse the full region detected by the λ_2 vortex region detector. Another core line is not detected though. Obviously we miss an important aspect without the second vortex core line since we can see in Figure 10 (c) that it influences the region of the vortex where non optimal mixing occurs.

6 Assessment of Numerical Behavior

In engineering applications it is not common to store all the information computed in the course of the CFD simulation permanently. Especially time derivative information is not generally stored in the data. Furthermore, the solver does not include all the timesteps computed in the solution file. In general we can expect the simulation design regarding cell types and cell sizes to be adequately chosen by the simulation designer. The simulation designer considers the necessary resolution for postprocessing such that reliable streamlines and pathlines can be constructed. From experience we know these settings to work well for computing vortex core lines in the steady case. Since time-derivative information is not stored and not all time steps are written out into the final dataset we need to evaluate the impact of larger step widths on the feature extraction process. Our application partners from Arsenal Research [30] have computed an unsteady solution to a pulsating flow in a tube t-junction (see Figure 11). Time dependent boundary conditions are used to produce flow separation inside the tube. The total mesh size is about 170000 cells.

During simulation 1570 timesteps have been generated resulting in 26 GB of compressed information. This is 10 times the temporal resolution our application partners would have stored usually for this simulation setting. To exclude possible interference from numerical problems introduced by the plane fitting technique we use to estimate the material derivatives also the Jacobian computed during the simulation have been included in the dataset. This way we can analyze how strong the impact of larger timesteps is when computing vortex core lines. We can use the time derivative computed for step width 1 as reference for the other step widths and measure the influence of larger step widths by computing the difference between the reference derivative and the respective derivative for the given step width. In Figure 11 the magnitude of this difference is mapped to color. To analyze the impact on vortex extraction, we focus on a horseshoe vortex directly behind the top inlet. We see the difference between the acceleration vector from step width of 1 and step widths 10 and 20. The vortex core lines resulting from smaller step widths than 10 do not differ significantly from each other. This is exactly the default step width resulting from the standard simulation procedure. For larger step widths the resulting vortex core line begins to deteriorate due to the noise introduced by the time derivative component of the acceleration vector. At step width of 20 we still get a similar but jagged result. At larger step widths the extracted line no longer resembles the horseshoe vortex in the data set. At step width 100 the line breaks into 3 unconnected components that follow the vortex core line for some length and then trail off in random directions.

We conclude that for standard step widths in well prepared simulations the time-aware vortex core line extraction method produces reliable results. Both for the especially designed dataset and the real world examples (where the Jacobian had to be estimated) we did not find the estimation of the time derivatives to introduce significant additional noise.

7 Conclusion

This paper proposes a new method to find vortex core lines in unsteady flows. Localization of vortices has been shown to be dependent on the temporal developments of the flow. We have given examples where vortex core extraction on time-frozen fields fails and have shown how to solve this problem. This result is not only relevant to vortex core extraction algorithms but to unsteady flow feature extraction methods

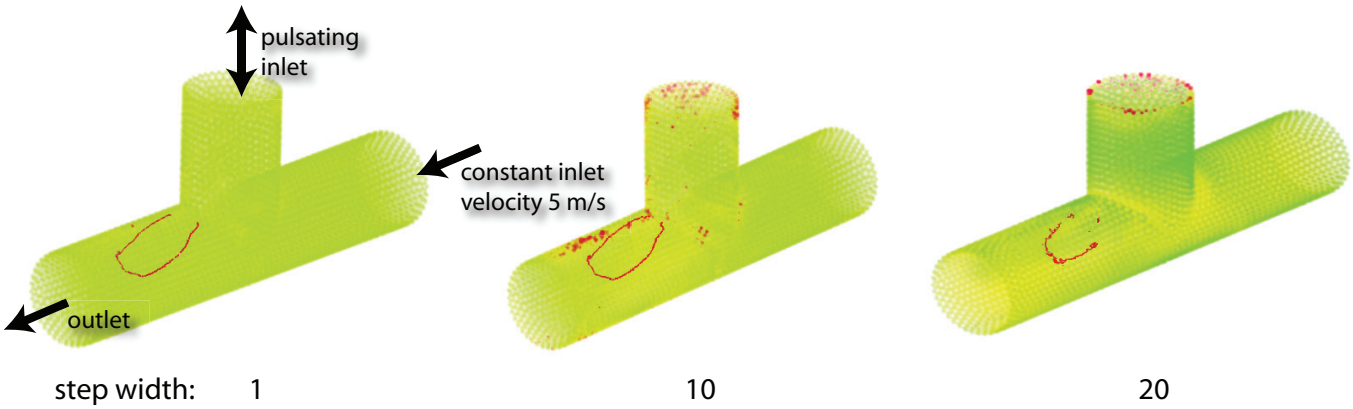

Fig. 11. Impact of time derivative estimation. The different step widths are measured in 1000^{-1} sec. The vortex core lines for stepwidths 1 to 10 do not differ visibly. Color is mapped to the difference between the time derivative for step width 1, and the respective step width (for step width 20 we have changed the color mapping by one order of magnitude).

in general. Since we could demonstrate that vortex core extraction algorithms have to include the temporal developments of the flow, it can be expected that similar results can be achieved for other flow features as well. Therefore we expect to see significant similar results in this direction in the future.

Based on the insight that it is necessary to include the time-derivative information into the feature extraction process we proposed a natural extension of the feature extraction process to unsteady flow data. By changing the underlying geometry from a streamline to a pathline based approach we can generalize existing feature extraction algorithms to unsteady flow data in a way that does not change their behavior on steady flows. We presented an algorithm that follows this approach extending parallel vectors operator criteria. Due to the consistent extension of the approach the algorithms change in a natural way and (given an implementation of the parallel vectors operator) the extension can be implemented quickly. The additional computation cost amounts to computing finite differences to estimate the time derivatives, therefore the difference to the original parallel vectors implementation is small.

We could confirm on real world data that the extracted vortices can differ significantly in position from the method of Sujudi and Haimes and in the large majority of the cases the extracted corelines are the same or better than those we got with the standard methods.

We conclude that for unsteady data the modified version of the algorithm of Sujudi and Haimes is the default choice. The higher order method generally performs very similar to the method of Sujudi and Haimes but it intensifies numerical issues. Also it requires additional computation. Therefore, only if after inspection of the data the results of the unsteady version of Sujudi and Haimes does not perform as expected, we suggest to switch to the modified higher order method.

REFERENCES

[1] D. C. Banks and B. A. Singer. A predictor-corrector technique for visualizing unsteady flow. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):151–163, 1995.

[2] D. Bauer and R. Peikert. Vortex tracking in scale-space. In *Proceedings of the 4th Joint IEEE VGTC - EUROGRAPHICS Symposium on Visualization (VisSym 2002)*, pages 140–147, 2002.

[3] R. Bürger, P. Muigg, M. Ilcìk, H. Doleisch, and H. Hauser. Integrating local feature detectors in the interactive visual analysis of flow simulation data. In *Proceedings of the 9th Joint IEEE VGTC - EUROGRAPHICS Symposium on Visualization (VisSym 2007)*, pages 171–178, 2007.

[4] M. S. Chong, A. E. Perry, and B. J. Cantwell. A general classification of three-dimensional flow fields. *Physics of Fluids Archive*, 2:765–777, 1990.

[5] C. Garth, G.-S. Li, X. Tricoche, C. D. Hansen, and H. Hagen. Visualization of coherent structures in transient flows. In *Proceedings of TopoInVis 2007: Topology-Based Methods in Visualization*, 2007. (to appear at Springer).

[6] C. Garth, X. Tricoche, and G. Scheuermann. Tracking of vector field singularities in unstructured 3d time-dependent datasets. In *Proceedings IEEE Visualization 2004*, pages 329–336, 2004.

[7] G. Haller. An objective definition of a vortex. *Journal of Fluid Mechanics*, 525:1–26, 2005.

[8] B. Hua, J. McWilliams, and P. Klein. Lagrangian accelerations in geostrophic turbulence. *Journal of Fluid Mechanics*, 366:87–108, 1998.

[9] J. C. R. Hunt, A. A. Wray, and P. Moin. Eddies, stream and convergence zones in turbulent flows. Technical report, Center for Turbulence Research Report CTR-S88, 1988.

[10] F. Hussain. Coherent structures - reality and myth. *Physics of Fluids*, 26:2816–2850, 1983.

[11] M. Jankun-Kelly, M. Jiang, D. Thompson, and R. Machiraju. Vortex visualization for practical engineering applications. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):957–964, 2006.

[12] J. Jeong and F. Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285:69–84, 1995.

[13] Y. Levy, D. Degani, and A. Seginer. Graphical visualization of vortical flows by means of helicity. *AIAA Journal*, 28:1347–1352, 1990.

[14] D. J. Mavriplis. Revisiting the least- squares procedure for gradient reconstruction on unstructured meshes. In *Proc. of the 16th AIAA Computational Fluid Dynamics Conference*, 2003.

[15] A. Okubo. Horizontal dispersion of floatable trajectories in the vincinity of velocity singularities such as convergencies. *Deep Sea. Res.*, 17:445–454, 1970.

[16] R. Peikert and M. Roth. The 'parallel vectors' operator: a vector field visualization primitive. In *Proceedings IEEE Visualization 1999*, pages 263–270, 1999.

[17] R. Peikert and F. Sadlo. Flow topology beyond skeletons: Visualization of features in recirculating flow. In *Proceedings of TopoInVis 2007: Topology-Based Methods in Visualization*, 2007. (to appear at Springer).

[18] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. The state of the art in flow visualisation: Feature extraction and tracking. *Computer Graphics Forum*, 22(4):775–792, 2003.

[19] F. Reinders, F. H. Post, and H. J. W. Spoelder. Visualization of time-dependent data with feature tracking and event detection. *The Visual Computer*, 17(1):55–71, 2001.

[20] M. Roth and R. Peikert. Flow visualization for turbomachinery design. In *Proceedings IEEE Visualization 1996*, pages 381–384, 1996.

[21] M. Roth and R. Peikert. A higher-order method for finding vortex core lines. In *Proceedings IEEE Visualization 1998*, pages 143–150, 1998.

[22] F. Sadlo and R. Peikert. Visualizing lagrangian coherent structures: A comparison to vector field topology. In *Proceedings of TopoInVis 2007: Topology-Based Methods in Visualization*, 2007. (to appear at Springer).

[23] S. Stegmaier, U. Rist, and T. Ertl. Opening the Can of Worms: An Exploration Tool for Vortical Flows. In *Proceedings IEEE Visualization 2005*, pages 463–470, 2005.

[24] D. Sujudi and R. Haimes. Identification of swirling flow in 3D vector fields. Technical Report AIAA-95-1715, American Institute of Aeronautics and Astronautics, 1995.

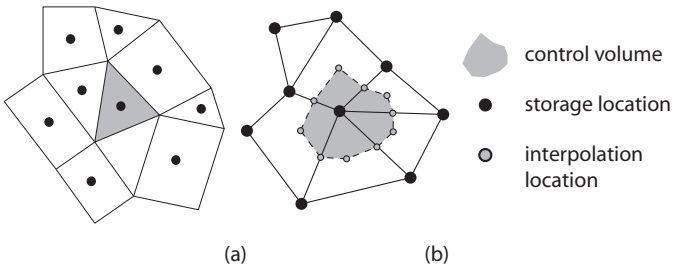

Fig. 12. Control volume variants used for numerical solution for CFD. (a) Cell-centered volume representation. (b) Vertex-centered volume representation. The segments surround the median dual control volume, i.e., the positions inside the cells are computed using the center of gravity for each cell.

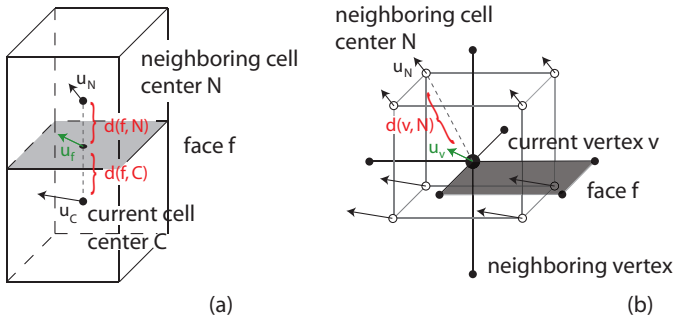

Fig. 13. Velocity extimation for different mesh types. (a) Estimating face velocities from cell centered data can be done by inverse distance weighting of the adjacent cell velocities. (b) Estimating vertex velocities from cell centered data can be done by inverse distance weighting of the surrounding cell velocities

[25] H. Theisel and H.-P. Seidel. Feature flow fields. In *Proceedings of the 5th Joint IEEE VGTC - EUROGRAPHICS Symposium on Visualization (VisSym 2003)*, 2003.

[26] 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 *Proceedings IEEE Visualization 2004*, pages 321–328, 2004.

[27] X. Tricoche, C. Garth, G. Kindlmann, E. Deines, G. Scheuermann, M. Ruetten, and C. Hansen. Visualization of intricate flow structures for vortex breakdown analysis. In *Proceedings IEEE Visualization 2004*, pages 187–194, 2004.

[28] T. Weinkauf, J. Sahner, H. Theisel, and H.-C. Hege. Cores of swirling particle motion in unsteady flows. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1759–1766, 2007.

[29] J. Weiss. The dynamics of enstrophy transfer in two-dimensional hydrodynamics. *Physica D Nonlinear Phenomena*, 48:273–294, 1991.

[30] Homepage of Arsenal Research. See URL `http://www.arsenal.ac.at`.

[31] Homepage of the SimVis Visualization Framework. See URL `http://www.simvis.at`.

APPENDIX

A ALGORITHM DETAILS

The vortex core line extraction process consists of three stages:

1. estimate velocities at vertices and faces (Subsection A.1)
2. reconstruct gradients at vertices and faces using estimated velocities (Subsection A.2)
3. for each cell subdivide into tetrahedra and use reconstructed gradients to find vortex core positions (Subsection A.3).

Depending on the type of simulation data storage can be either vertex or cell centered (see Figure 12). In the third vortex core line extraction step we need gradients at the nodes of the grid, and the gradient reconstruction step varies slightly for the two storage types.

A.1 Velocity Estimation

To reconstruct velocities we use a standard inverse geometric weighted interpolation scheme.

For estimating face velocities from cell centers, we define the distance between the center of a cell and one of its faces as the distance between the cell center and the center of gravity of the face. The velocity at face u_f is computed as

$$u_f := \alpha u_C + (1-\alpha) u_N$$

where C and N are the two cells adjacent to the face f. Here the weighting geometric factor α can be computed as $\alpha := \frac{d(f,N)}{d(f,C)+d(f,N)}$, where $d(.,.)$ denotes the Euclidean distance (see Figure 13 (a)). When working with vertex-centered volume representation, the velocity at a face can be computed by taking the average of the surrounding vertices.

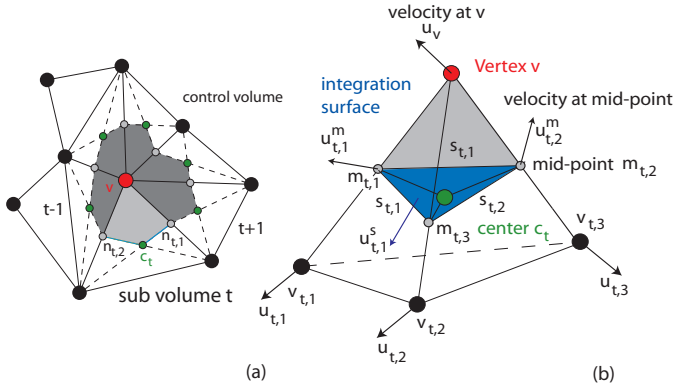

Fig. 14. Gradient estimation at a vertex (red) using the Green-Gauss theorem requires to estimate cell center velocities and mid-point velocities. (a) The surrounding surface uses cell centers (green) and mid-points (gray). (b) In this detail illustration of the lighter gray section from (a) we see the full configuration for a single surrounding tetrahedron.

The velocity at a node v can be computed from the surrounding cell centers by using the cell values of the surrounding cells. Again the weight is taken as the inverse of the distance of the node from the cell center. Let N_C be the number of cells surrounding v, C_i the center of the i-th neighboring cell, and u_{C_i} its velocity vector. Then we can compute the velocity u_v at v as $u_v := \sum_{i=0}^{N_C} u_{C_i}\alpha_i$ where the weight of the i-th cell is $\alpha_i := d(v,C_i)^{-1}$ (see Figure 13 (b)). The velocity at a cell center from the surrounding vertices for vertex centered grids can be computed by taking the inverse distance weighted average of vertices of the cell.

A.2 Gradient Reconstruction

In addition to the velocity values at the vertices to the cell we also need the velocity gradients. To compute the gradients we suggest to use the Green-Gauss reconstruction method which works with velocity values from the faces of the cell. The least-squares linear reconstruction method can be used when no connectivity values are present (for example when working with point clouds).

A.2.1 Green-Gauss Linear Reconstruction

Let Ω be a volume (a cell of the mesh for cell centerd representation or the median control volume for vertex centered representations), $S = \partial\Omega$ the bounding surface of Ω, φ some scalar function defined on Ω, and $\nabla\varphi$ the derivative of φ. Then the Green-Gauss theorem states that the surface integral of the scalar function φ times the normal vector

of the surface over the surface S is equal to the volume integral of the gradient $\nabla\varphi$ over the volume Ω:

$$\int_\Omega \nabla\varphi d\Omega = \int_S \varphi \mathbf{n} dS.$$

To compute the derivative at the center of the control volume we assume that $\nabla\varphi$ is constant over the control volume and the volume integral over $\nabla\varphi$ reduces to the volume of Ω times ∇u:

$$|\Omega|\nabla\varphi \approx \int_\Omega \nabla\varphi d\Omega = \int_S \varphi \mathbf{n} dS.$$

Finally, we can approximate the integral over the bounding surface using face values. That is

$$\nabla\varphi = \frac{1}{|\Omega|} \sum_{faces} \varphi \cdot area(face_i) \cdot n_f$$

where $area(triangle)$ is the surface area of a triangle.

To compute the derivative at a vertex we can use the control volume depicted in Figure 14 and get

$$|\Omega_C|\nabla u \approx \sum_{t=0}^{N_t(v)} \sum_{i=1}^{3} area(s_{t,i}) \cdot u^s_{t,i}.$$

Here $N_t(v)$ is the number of tetrahedra at vertex v. Here we are using an interpolated velocity vector at the mid-points $u^m_{t,i} := \frac{1}{2}(u_v + u_{t,i})$ and the velocity at the cell center to construct the surface velocity $u^s_{t,i} := \frac{1}{3}(u^m_{t,i} + u^m_{t,i+1} + u_c)$ (with $u^m_{t,4} := u^m_{t,1}$). See Figure 14 for an illustration.

A.2.2 Least-Squares Linear Reconstruction

Here the gradient is estimated by fitting a hyperplane to the cell such that the difference between the extrapolated value for the surrounding cells and the present values of the surrounding cells are minimized.

For each edge of the resulting mesh incident to the vertex v_0, an edge projected gradient constraint equation is constructed using inverse distance weights α_i for each edge:

$$\alpha_k(\nabla u) \cdot (x_k - x_0) = \alpha_k(\varphi_k - \varphi_0).$$

The gradient construction is obtained by solving a least-squares optimization problem to minimize the sum of the distances between the estimated values and the vertex values. This approach implicitly smoothes the data and can improve the results when working with noisy data.

Which weighting scheme works best is still an open question. Mavriplis [14] stresses that the minimization problem will be much better conditioned when using inverse distance weighting. On the other hand when the mesh is irregularly sampled and on one side of a cell we have a large number of small triangles and on the other side just a few larger triangles this can lead to a gross misrepresentation of small triangles. Therefore we use unweighted direct neighbors for estimating the gradient at a cell by default and only change this procedure when necessary.

A.3 Pseudocode

```
INPUT: unstructured grid
OUTPUT: array of line segments

lines *core = new lines();
//pre-processing
cells *c = grid->getVertexCells();

//check each cell for vortex-core
foreach cell ∈ c {
    //quick: check for two face intersections
    vector<tri> *tris = cell.getTriFaces();
    result = checkFaces(tris);

    if(result.size() == 2)
        core->add(result);

    //fallback: check all tets
    else {
        //get tetrahedralization
        tets *t = cell.getTets();
        foreach tet ∈ t
            core->add( checkFaces(tet) );
    }
}
return core;
```

Fig. 15. General algorithm outline. The method 'getTriFaces()' returns a vector of triangulated faces for a cell. The method 'checkFaces()' applies the parallel vectors operator to each face in a list and returns a list of points where v and w are parallel. The method 'getTets()' returns a tetrahedralization of a cell using face centers and the cell center as additional points.

```
INPUT: triangle (t1,t2,t3),
       flow (v1,v2,v3),
       accel (w1,w2,w3)
OUTPUT: parallel positions

//compute increments
mat incrV = (v2-v1, v3-v1, v1);
mat incrW = (w2-w1, w3-w1, w1);

//find parallel positions
if(det(incrV) != 0) {
    mat inv = incrV.inverse();
    mat sol = inv * incrW;
    vector *eig = sol.realEigenV();
    vector *pos;

    foreach e ∈ eig {
        float s = e->x/e->z;
        float t = e->y/e->z;
        if(s>=0 && t>=0 && s+t<=1)
            pos.add(t1+s*t2+t*t3);
    }
    return pos;
}
```

Fig. 16. Pseudo code of the parallel vectors operator on a triangle. If the determinant of incrV is zero, also incrW has to be checked. The method 'inverse()' returns the inverse matrix. The method 'realEigenV()' returns a list of eigenvectors of a matrix having real eigenvalues.