Vortex Visualization for Practical Engineering Applications

Monika Jankun-Kelly, *Student Member, IEEE*, Ming Jiang, *Member, IEEE*,
David Thompson, *Member, IEEE*, and Raghu Machiraju, *Member, IEEE*

Abstract—In order to understand complex vortical flows in large data sets, we must be able to detect and visualize vortices in an automated fashion. In this paper, we present a feature-based vortex detection and visualization technique that is appropriate for large computational fluid dynamics data sets computed on unstructured meshes. In particular, we focus on the application of this technique to visualization of the flow over a serrated wing and the flow field around a spinning missile with dithering canards. We have developed a core line extraction technique based on the observation that vortex cores coincide with local extrema in certain scalar fields. We also have developed a novel technique to handle complex vortex topology that is based on k-means clustering. These techniques facilitate visualization of vortices in simulation data that may not be optimally resolved or sampled. Results are included that highlight the strengths and weaknesses of our approach. We conclude by describing how our approach can be improved to enhance robustness and expand its range of applicability.

Index Terms—Vortex detection, vortex visualization, feature mining.

1 Introduction

As computing power continues to increase, large scale computational fluid dynamics (CFD) simulations of complex vortical flows about realistic configurations are becoming routine. In this paper, we consider two such examples. The first case considered is the flow around the serrated wing [9] configuration shown in Figure 1. Wing planforms such as these have been suggested as candidates for highly maneuverable configurations. The second case is the flow field around a spinning missile with dithering canards [4] shown in Figure 2. The vortical flows associated with these two configurations are very complex in comparison to those often used in the visualization literature. Further, the data sets derived from these simulations are large and may preclude the use of many commonly used visualization techniques. For example, the spinning missile data set contains more than nine million nodes [4] and requires 360 time steps to describe a single rotation of the missile.

Feature-based techniques provide viable methods to visualize large scale simulations of this type [22, 28, 31]. These methods extract features (in this case vortices) and define them in terms of feature-based descriptors. However, many existing techniques, when applied to large scale data sets with complex physics, like those described above, produce unsatisfactory results. Simulations of geometrically complex configurations are often performed on unstructured meshes. This impacts the complexity of extraction algorithms and potentially may impact solution accuracy. Further, artifacts due to the discrete nature of the solution may taint the accuracy of gradient based computations. Finally, in unsteady cases, the flow field may contain features that continuously change attributes (e.g., sense of rotation), appear and disappear, merge and split.

- *Monika Jankun-Kelly is a Graduate Research Assistant at the Computational Simulation and Design Center, Mississippi State University, E-mail: mjk@simcenter.msstate.edu.*
- *Ming Jiang is a Postdoctoral Researcher at the Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, E-mail: jiang4@llnl.gov.*
- *David Thompson is an Associate Professor in the Department of Aerospace Engineering at Mississippi State University, E-mail: dst@simcenter.msstate.edu.*
- *Raghu Machiraju is an Associate Professor in the Department of Computer Science and Engineering at The Ohio State University, E-mail: raghu@cse.ohio-state.edu.*

Manuscript received 31 March 2006; accepted 1 August 2006; posted online 6 November 2006.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

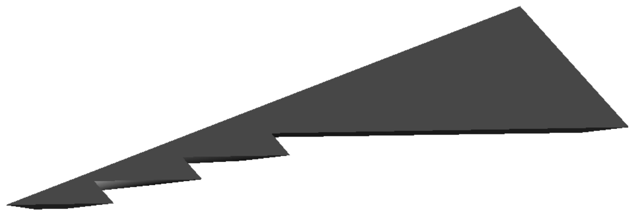
Fig. 1. Serrated wing geometry: The serrations create vortical flow that enhances vehicle maneuverability.

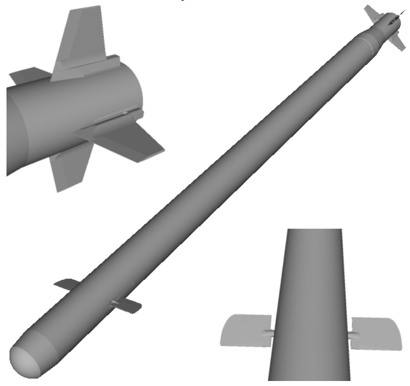
Fig. 2. Missile geometry: Canted tail fins (upper left) cause missile to spin about its longitudinal axis. Canards (lower right) rotate synchronously about axis passing through the missile body to provide pitch and yaw control.

In this paper, we describe a process for developing efficient feature-based visualizations of vortex dominated flow fields. Our approach can be thought of as a generalization of the work of Banks and Singer [1] and is closely related to the approach described by Stegmaier et al. [29] who realized an efficient browsing tool that separates vortices detected using a scalar field [14]. Stegmaier et al. extract core lines with a predictor/corrector algorithm that utilizes four point Lagrangian interpolation and the "direct search" method of Hooke and Jeeves [11]. Our approach, while enhancing existing methods, introduces two new components. A predictor/corrector algorithm that uses a novel function fitting procedure to locate the extreme values of a scalar field is employed in the core line extraction component. This allows us to extract vortices from simulation data that may not be optimally resolved or sampled. A *k*-means clustering algorithm [19] is utilized to handle complex vortex topologies such as those produced by vortex merging. More specifically, our feature extraction algorithm is composed of the following steps: (1) a feature detection step to identify candidate cells, (2) an aggregation step to combine candidate cells into contiguous structures, (3) a *k*-means clustering algorithm to handle complex vortex topologies, (4) a robust vortex core line extraction algorithm, and (5) a tangential velocity determination of vortex extent. Once found, vortex core lines, extent, and other characteristics are compactly stored and can be visualized as shown in Figure 3. To reiterate, we consider our major contribution to be the development of robust and efficient vortex visualization techniques that can work on realistic engineering data sets.

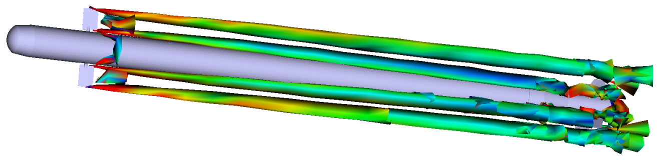
Fig. 3. Vortex dominated flow for spinning missile with dithering canards. The surfaces are employed to illustrate the extent of the vortices. Each surface is shaded according to the local value of the tangential velocity in the plane that contains the swirling motion.

1.1 Related Work

Comprehensive overviews of existing vortex detection algorithms are given in [18, 26], along with useful taxonomies to help guide the selection of algorithms appropriate for various situations. One categorization is based upon the representation of the vortex. A line-based algorithm extracts a vortex core line while a region-based algorithm extracts a region in which a vortex is located. In general, line-based algorithms can more precisely locate vortex cores, whereas region-based algorithms can be computationally less expensive.

Banks and Singer [1] developed a line-based algorithm that employs a predictor/corrector sequence based on the assumptions that the vortex core is a vorticity line and that pressure attains a minimum in the core. Sujudi and Haimes [30] described a line-based method that extracts the vortex core by locating points that satisfy the following two conditions: (1) the velocity gradient tensor has a pair of complex eigenvalues and (2) the velocity in the plane perpendicular to the real eigenvector is zero. Roth and Peikert [24, 27] proposed a different approach for detecting core lines using the parallel vectors operator. Their algorithm detects parallel alignment between the velocity vector and the jerk vector, where jerk is the time derivative of the vector acceleration.

Region-based algorithms are also prevalent in the vortex detection literature. Levy et al. [21] developed a method based on the assumption that a vortex core is located in a region where the normalized helicity approaches ± 1. Berdahl and Thompson [3] assume that two of the eigenvalues of the velocity gradient tensor are a complex conjugate pair in regions of swirling flow. A swirl parameter is defined at each point in the domain using the magnitude of the imaginary part of the conjugate pair and the velocity in the plane perpendicular to the real eigenvector. Jeong and Hussain [14] defined a vortex based on the symmetric deformation tensor S and the antisymmetric spin tensor Ω. According to Jeong and Hussain, if λ_2, the second largest eigenvalue of $S^2 + \Omega^2$, is negative at a point, that point is contained within a vortex. Jiang et al. [16] proposed a method for extracting vortex core regions based on ideas from combinatorial topology. In this approach, a combinatorial labeling scheme based on Sperner's Lemma is applied to the velocity vector field in order to identify centers of swirling flows.

Techniques for improving detection results have been presented for both types of algorithms. Bauer and Peikert [2] proposed a preprocessing step for the parallel vectors operator [26] to reduce the number of artifacts that occur due to the computation of higher order derivatives. Jiang et al. [17] proposed a postprocessing step for automatically verifying the results from the detection algorithm. However, many of the existing methods are still not robust enough to deal with complex simulation data. The work presented by Stegmaier et al. [29] attempted to address this problem. They combined the method of Jeong and Hussain [14] with that of Banks and Singer [1]. The former provides the seed points for growing a skeleton using the framework of the latter. The initial location of the seed points is improved upon by a local search.

Several approaches for computing the physical extent of a vortex have been presented in the literature. In order to determine the extent of the cross section of a vortex, Banks and Singer [1] proposed an approach to sample pressure and vorticity along radial lines emanating from the detected core line, until user defined thresholds are exceeded. Roth [26] proposed a similar approach using a fraction of the value at the core position for each cross section as the threshold. Recently, Garth et al. [8] proposed an improvement to the above approach by approximating the physical extent without the need for user defined thresholds. In this case, the sample values are tangential velocities on the perpendicular plane, and the termination criterion is defined implicitly as the local maximum.

2 Issues Associated with Unstructured Meshes

In this section, we describe some of the challenges associated with visualizing flow fields simulated on large scale unstructured meshes. These challenges can be classified loosely into two categories: geometry/mesh issues and simulation issues.

2.1 Geometry/Mesh Complexity

For high Reynolds number flows such as those considered here, very large velocity gradients occur in the region near the body surface where the no-slip condition is satisfied. This region, typically referred to as the boundary layer, requires resolution in the direction normal to the wall that is several orders of magnitude smaller than the resolution required along the wall. Therefore, it is much more efficient to discretize the region containing the boundary layer using highly anisotropic cells such as prisms instead of isotropic tetrahedral cells. Away from the boundary layer, where there is no preferred direction, isotropic tetrahedral cells provide a flexible mechanism for discretizing the computational domain. Meshes containing prismatic and tetrahedral cells are sometimes referred to as hybrid meshes. Hybrid meshes may also contain pyramids as transitional cells. Thus, a vortex detection algorithm should be appropriate for any number of cell types and not be restricted to tetrahedral cells or cells that can readily be decomposed into tetrahedral cells.

Cells in an unstructured hybrid mesh may have any number of neighbors in any position. Although we know the number of face neighbors, e.g., four for a tetrahedral cell, we cannot know the valence of a given node. We cannot obtain the neighbors of a cell merely by adjusting the cell index, as we could with a structured grid. This is a complicating factor for any vortex detection algorithm. Mesh cell connectivity must be explicitly stored and any computation that involves a cell and its neighbors must be able to deal with an arbitrary number of neighbors in arbitrary positions.

The connectivity problem is further complicated for the spinning missile with dithering canards because there is relative motion between components of the configuration. There are two basic approaches that can be employed for simulating relative motion: mesh deformation, which maintains the mesh connectivity, or mesh reconnection. For the missile geometry, Blades and Marcum [4] employed a local reconnection strategy, which results in the mesh connectivity changing with every time step.

2.2 Volume Search

Volume search, finding the cell containing a given point, is a common task employed in numerous components of our vortex detection algorithm. It can be broken down into two subtasks: locating a starting node near the point and searching from the starting node for the cell containing the given point. The first task is accomplished through the use of optimizing data structures, such as an octree or *k-d* tree. Langbein et al. [20] employ a *k-d* tree approach for very large unstructured meshes in their work. We use a balanced octree to efficiently store nodes. Once a starting node near the given point is found, the search for the desired cell containing the given point can start. We use the A* best first search algorithm [10] to search for the cell in the direction to the given point. An unstructured mesh presents additional challenges relative to a body-fitted structured grid because internal boundaries appear as holes in the computational domain while the structured grid typically has no interior holes when represented in its native parametric space. If a boundary, a concavity or hole, is encountered, the search algorithm attempts to skirt around it. If necessary, the algorithm backtracks. The combination of optimizing data structure and A* best first search makes for a very efficient volume search algorithm. Furthermore, the search can be confined to a local neighborhood by limiting the number of cells traversed. This limitation prevents exhaustive searching of all cells when the given point is outside the volume or in a hole, and thus not in any cell.

2.3 Flow Field Resolution

One major challenge associated with detection of features in simulation data is the quality of the simulation. Typically, the numerical dissipation inherent in the discrete approximation increases as cell size increases. This is especially problematic for the vortices trailing from the canard tips on the spinning missile, which may traverse long distances through relatively coarse regions of the mesh and may experience significant numerical dissipation. We address this problem by using a core line extraction algorithm that is designed to work for a vortex that varies in strength along its axis. This is in contrast to region-based approaches that typically require isosurfacing to visualize the vortex [18]. A second issue is the discrete nature of the solution. Unless a vortex core happens to fall on a cell center, its location may be difficult to detect. We address this problem using techniques from statistical estimation and function fitting. We exploit an assumed variation of the field values near the vortex core to obtain subcell resolution thereby enhancing the estimate of the location of the vortex core. Finally, in practical engineering applications, some features may not be resolved adequately. For example, the solution on a given mesh may not be adequate to accurately capture the physically complex process of vortex merging. In these instances, the detection algorithm is challenged to maintain the identities of the distinct vortices. We address this problem by using the *k*-means clustering algorithm to separate the merging vortices [19]. These techniques are described in detail in the section that follows.

3 VORTEX DETECTION AND VISUALIZATION

The algorithm that we employ for vortex core line extraction is an enhancement of the Banks and Singer [1] algorithm. Our algorithm exploits the fact that the local extrema of other scalar fields (and not just pressure) can also be employed to locate vortices. Additionally, we utilize a novel clustering algorithm (see Section 3.4) to handle complex vortex topologies. Further, we use a statistical function fitting approach in a predictor/corrector step to extract the core line as a series of local extreme values (minima or maxima as appropriate for the field) as described in Section 3.5.4. We employ the vortical flow about the serrated wing, shown in Figure 1, to illustrate the functions of the various components of our algorithm. Additional results for the spinning missile with dithering canards, shown in Figure 2, are included in Section 4.

3.1 Identification of Vortex Regions

We first identify regions that may contain a vortex core using a region-based technique. We compute the necessary scalar field only in regions where the velocity gradient tensor has complex eigenvalues [25], which is a necessary but not sufficient condition for swirling flow. The computed scalar field should be one that has a line-type extremum in the core of a vortex. The swirl parameter [3] satisfies the necessary conditions because it is a scalar field that has a line-type maximum

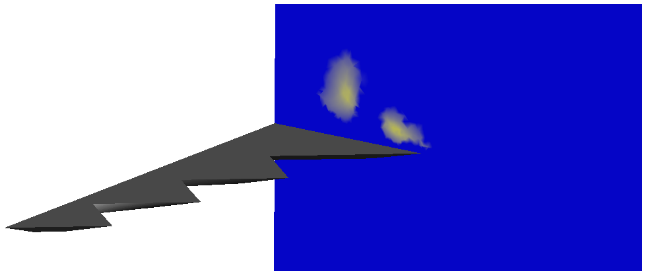

Fig. 4. Compute a scalar field whose local extrema coincide with vortex core lines. The swirl parameter (shown) is such a field. The computation is performed only in cells where the velocity gradient tensor has complex eigenvalues which indicate the presence of swirling flow.

near the core of a vortex. Figure 4 shows a slice of the swirl parameter field. The two vortices that intersect the slicing plane appear as two local maxima. Very low values of the swirl parameter (fifth percentile or less) indicate lack of swirling flow. A fifth percentile threshold can be used to remove false positives due to very low valued local maxima. Other scalar fields such as vorticity magnitude [6] (maximum) or λ_2 [14] (minimum) may be used instead of the swirl parameter. Compared to the swirl parameter, vorticity magnitude is less computationally expensive and has the advantage of Galilean invariance, but is more prone to false positives in shear regions [6]. The Galilean invariant λ_2 field, which is negative in regions of swirling flow, was developed to detect vortices in *incompressible* flows [14]. Therefore, it is not appropriate for detecting vortices in the compressible, supersonic flow field associated with the spinning missile with dithering canards. We use the swirl parameter in the results shown in Section 4.

3.2 Vortex Detection with Local Extrema Method (LEM)

The essential step of our detection algorithm is based on the observation that a vortex core is coincident with local extrema in pertinent scalar fields in the plane normal to the vortex core line. Thus, a vortex core line can be extracted from an appropriate scalar field by locating local extrema in a properly oriented plane and connecting the location of the extreme values with line segments to form a curve.

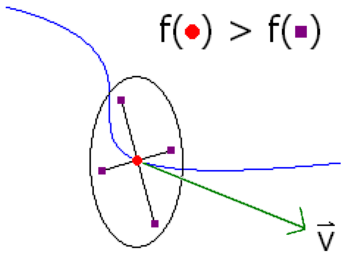

Fig. 5. Stencil employed for the local extrema method (maximum). The stencil plane is perpendicular to the swirl vector **v**. Note: The maxima of the scalar field **f** coincide with vortex cores. **f(circle)** and **f(square)** are scalar values. The **circle** point lies in the cell under consideration. The **square** points lie in a local neighborhood outside the cell.

The local extrema method (LEM) determines the locations of local extrema of a scalar field f. Figure 5 depicts the stencil employed to determine where local extrema of a scalar field f occur. The central point of the stencil lies in the cell currently under consideration, while the other four points lie outside the cell along perpendicular rays in a planar local neighborhood. The vector **v** is tangent to the curve and normal to the local neighborhood. Either vorticity or the real eigenvector of the velocity gradient tensor can be used as the vector **v** because they are both an approximate tangent to the vortex core line and the local extremum curve coincident with it. We used the real eigenvector, which is computationally more expensive, but can be a better approximation near boundaries. Determining a stencil by ray tracing through an unstructured mesh is nontrivial. When f is greatest at the central stencil point, the current cell lies on a local maximum curve. The local minima of a scalar field f can be similarly found. The cells containing local extrema are marked by the LEM as vortex core candidate cells.

We emphasize that the LEM alone should not be considered as yet another vortex detection technique. Rather, the LEM post processes scalar field data obtained from region-based vortex detection algorithms and locates the highly anisotropic regions around vortex core lines. Thus, the LEM can be thought of as the underlying algorithm for the original Banks and Singer algorithm [1].

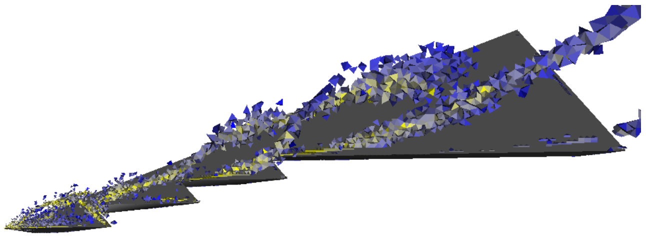
Fig. 6. Candidate cells are found using the local extrema method (LEM) whose input is the scalar field from Figure 4.

The results from the LEM shown in Figure 6 clearly suggest the nature of the vortical flow. The drawback of the LEM is its inability to determine whether the underlying cause of a local extremum is a vortex or some other feature, such as a strong shear region. This limitation is common to many vortex detection methods and can be addressed by feature level verification. The advantages of the LEM are generality, ease of use, and the use of functional comparisons rather than numerical gradients, which may be inaccurate due to discretization errors. For the functional comparisons, a stencil with a radius of three or four cells can be used to find extreme values in noisy, non-smooth data. The stencil radius is a topological measure (number of cells), not a spatial one (distance).

3.3 Aggregation of Candidate Cells

Although a visualization of LEM output appears as long slender regions, the output is only a list of individual candidate cells. Since we desire to work at the feature level rather than the cell level, we group candidate cells into contiguous aggregates. We can remove false positives occupying small regions by unmarking cells in aggregates that contain less than a certain number of cells.

Aggregation groups contiguous candidate cells. Naturally, cell connectivity is needed for aggregation and efficient storage of connectivity becomes an issue. When dealing with an unstructured mesh, the question of which cells are considered contiguous for the purpose of aggregation is nontrivial. Typically, aggregates formed using face, edge, and node neighbors are *rougher* than those formed without using node neighbors, therefore we do not consider node neighbors contiguous while aggregating.

The aggregates formed for the serrated wing are shown in Figure 7. Note that the purple aggregate includes two initially distinct vortices that merged. The clustering algorithm described in the next section is employed to identify the topology of the merged vortices.

3.4 Clustering for Vortex Topology Identification

One of the problems with existing vortex detection algorithms is that they have difficulty distinguishing vortices that are in the process of merging. We would like to be able to distinguish the individual vortices that are in the process of merging from the resulting merged vortex. The serrated wing data set exhibits this phenomenon, which is shown in Figure 7. There are two separate vortices arising from the first and second serration, as numbered from the left. They start to merge together slightly to the right of the third serration, forming a Y-shaped aggregate, which is colored purple. This topology presents challenges for the core line extraction algorithm because only one vortex core line is found per aggregate. Branching aggregates need to be subdivided into component non-branching aggregates before core line extraction can be applied.

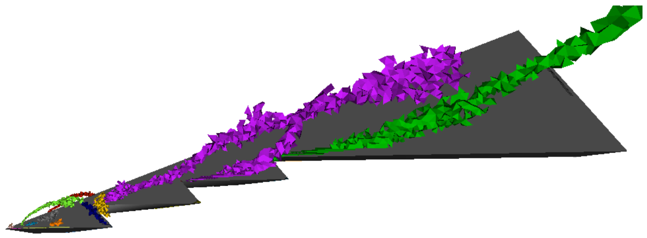
Fig. 7. Contiguous candidate cells are aggregated. Aggregates with less than a specified minimum number of cells are removed.

In order to correctly identify the vortex topology, we developed a clustering algorithm to identify if and where two vortices merge – in other words, where the branching of the aggregate occurs. This clustering algorithm is a variation of the k-means clustering algorithm [19, 23], designed specifically for unstructured meshes. The idea is to decompose the entire aggregate into small segments (clusters) and identify if and where a branching segment occurs. To create the initial set of clusters, we apply the standard region-growing algorithm for unstructured meshes, but we only grow each cluster to a fixed neighborhood size. Each neighborhood iteration consists of adding the connected neighbors that are within the aggregate to the cluster. For the serrated wing data set, we used a neighborhood size of four.

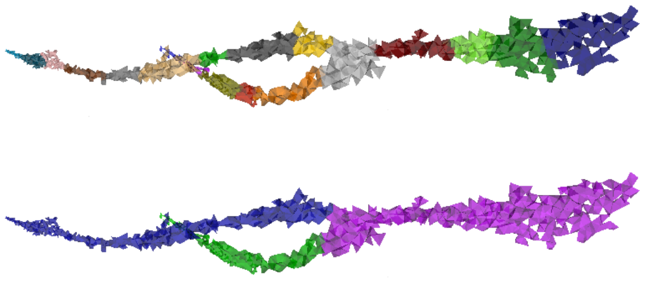
Fig. 8. Zoomed in view of the merging vortices. The top image shows the clusters from the modified k-means algorithm. The bottom image shows the merged clusters.

Once the entire aggregate is decomposed, we winnow out the tiny clusters by redistributing the cells within the tiny clusters to the closest connected cluster. For the serrated wing data set, we used a minimum cluster size of 16. Next, we compute the centroids of all the remaining clusters. At this point, we apply the standard k-means clustering algorithm [19, 23], with the only difference being that we do not use any prescribed value for k. The algorithm terminates once the same set of cells has been identified to be the centroids. Figure 8 shows the individual clusters resulting from our algorithm for the merging vortices. After the clusters have been computed, we proceed to identify if and where a branching cluster occurs. In this case, a *simple* cluster should have at most two neighboring clusters and only one if it is at the end. A *non-simple*, or branching, cluster will have more than two neighboring clusters. We identify branching clusters and merge the clusters

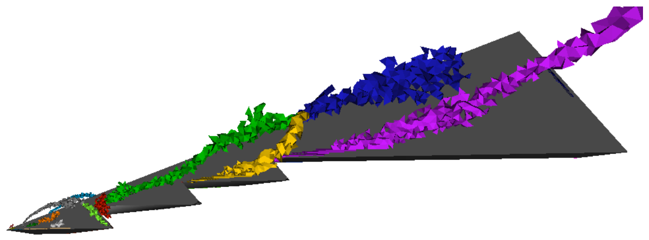
Fig. 9. Aggregates after clustering. Notice that the single purple aggregate shown in Figure 7 has been divided into three constituent aggregates.

in the branching regions separately. Figure 9 shows the result from the clustering and merging step of our algorithm. Notice that we have correctly distinguished the merging vortices from the merged vortex.

3.5 Predictor/Corrector Core Line Extraction

The LEM identifies highly anisotropic regions that contain curve shaped extrema of the selected scalar field. These were split into non-branching aggregates through k-means clustering. However, these are still aggregates of candidate vortex core cells, rather than lines. To extract the line in each aggregate, we pick an initial seed point, then trace forward and backward from the seed point.

3.5.1 Selection of Seed Points

There are several seeding strategies that can be employed. Most commonly, we select the center of the “best” cell intersected by the aggregate’s bisecting plane. The “best” cell can be the one with the largest absolute value of normalized helicity or most extreme value of the scalar field that was used as input to the LEM. The normalized helicity should not be used in cases where the vorticity is not a good approximation of the swirl vector. The bisecting plane approach is not suitable for aggregates whose cell sizes vary widely because it can place the seed point in a coarse region of the mesh where the solution is less accurate. In that case, we can select the cell with the smallest size or the one with the most extreme scalar field value of all the cells in the aggregate. In the results presented here, we have used bisecting plane with normalized helicity seed cell selection. It is highly unlikely that the core line passes exactly through the seed cell center, so we perform the correction step, described in Section 3.5.4, to find the seed point.

3.5.2 Predictor/Corrector Core Line Extraction

Starting from the seed point, we grow the core line. For each cell along the way, we predict which cell we will visit next, and correct the location of the next point along the line. If only the prediction step is used, the curve extraction algorithm is strongly dependent on the choice of the initial seed point. Correction improves the quality of the extracted core line by ensuring that the extracted core line passes through a locally extreme value in each cell.

3.5.3 Local Direction of Core Line Growth

Given a current point on the core line, we wish to find the next point along the core line. We predict where the next point will lie using the swirl vector, which is an approximate tangent to the vortex core line. Either the vorticity vector or the real eigenvector of the velocity gradient tensor can be used as the swirl vector. We have used the real eigenvector. When using the real eigenvector, care must be taken because it can point either forward or backward along the curve. The vorticity vector typically points in the same direction along the curve. An exception to this is when, for external reasons, the vortex reverses direction of rotation along its length. A ray is shot from the current point in the direction of the swirl vector. The intersection of that ray and a cell face is found and is taken to be a predicted point on the curve.

3.5.4 Correction using Function Fitting

Now the correction step is performed. We accomplish the correction via a novel function fitting approach applied to the scalar field in the swirl plane. Our objective is to find the location of the extreme value in the local swirl plane. It is highly unlikely that the extreme value falls exactly on a cell center. This extreme value cannot be found with linear interpolation of the values at the cell centers. If we use a higher-order polynomial interpolation, unrealistic oscillations may occur. What we do know about the field is that it attains a local maximum (or minimum) at the vortex core line in the swirl plane perpendicular to the vortex core line. Therefore, in the local neighborhood of the vortex core line, we can search for the location of the local maximum (or minimum) by fitting a conical function to the data in the swirl plane. We chose a conical function because, like the field in the local neighborhood of the core line, it has a single extreme value.

Function fitting consists of several steps. First, we select the data to be fitted. The data points are the center of the cell in which the core line currently resides, as well as the centers of the neighboring cells (face, edge and node neighbors). We have found that, for CFD solutions, using all the neighbors produces better fits than using only the face neighbors. Next, we take the cell centers and project their locations onto the swirl plane. We have tried two projection methods: using the values at the original cell centers and interpolating the values at the points projected onto the swirl plane. We found that using the values at the original centers works better. Interpolation smears and degrades the data. We can justify using the values at the cell centers as follows. In the small local neighborhood near a vortex core line, the isosurfaces of the scalar field appear as tubes around the vortex core. Thus, the original cell center and its projection onto the swirl plane lie on the same ”isotube” and have the same value of the scalar field.

Now that we have two-dimensional data in the swirl plane, we can fit a conical function to it. Fitting involves moving the center of the cone, the location of the function’s extreme value, to different positions in the local neighborhood, computing the standard deviation of the fitting error, and selecting the location with the smallest standard deviation of the fitting error. Our local neighborhood is the size of a two-dimensional bounding box that contains all the projected data points in the swirl plane. We apply a local rectilinear grid to this neighborhood and place the cone’s center at each of the grid points in turn. After experimentation, we chose a 20×20 grid size because we wanted to use the smallest grid we could to save computation time, while keeping the grid fine enough that the corrected point found with function fitting was an improvement over the uncorrected point.

The local max fitting function we use is given below. Let $f(r)$ be a conical function of the radius r measured from the center of the cone.

$$f(r) \quad = \quad f(0) + m * r$$

The cone’s central value $f(0)$ and the slope m are

$$\begin{aligned} f(0) &= f_{max} \\ m &= -(f_{max} - f_{min})/d \end{aligned}$$

where d is the length of the rectilinear grid diagonal, while f_{min} and f_{max} are

$$\begin{aligned} f_{max} &= s * max \\ f_{min} &= min \end{aligned}$$

where min and max are the extreme values of the projected data and s is a scaling factor. We scale up max because we expect the maximum whose location we seek to be greater than max. We use $s = 1.5$ for $max > 0$. For $max < 0$, we would use $s = 2/3$, but this does not occur in the scalar fields we employ. The local min version of the fitting function differs in three ways. (1) The cone’s central value $f(0)$ is f_{min}. (2) The slope m is positive. (3) The scaling factor s is applied

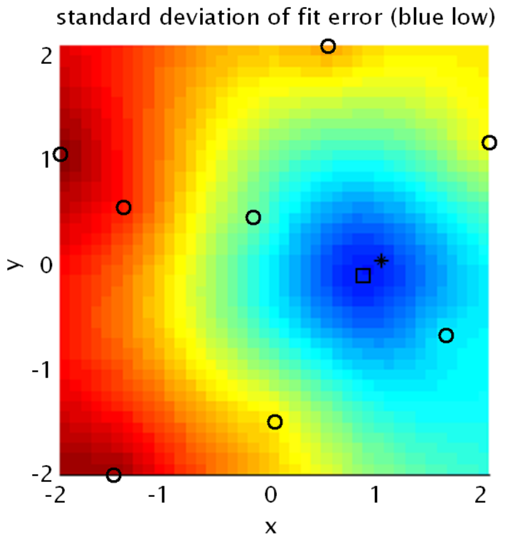

Fig. 10. Locating the maximum through function fitting. Circles represent points where a known function was sampled. The star is the location of the known maximum. A conical fitting function with a maximum at its center is fit to the data. The color indicates the standard deviation of the error (red is high, blue is low). Using this approach, we are able to obtain subcell resolution in the vortex core position computation.

to *min* rather than *max* when computing f_{min} and f_{max}. For $min < 0$, $s = 1.5$. For $min > 0$, $s = 2/3$.

Several correction iterations may be necessary. The corrected point may fall outside the cell containing the predicted point, in which case another correction taking into account the neighbors of this new cell is needed. We correct iteratively until we converge on a cell or complete 10 correction iterations. If there is no convergence, we reject the correction and revert to the predicted point, but this happens only about 1% of the time. In 50% to 70% of cases, convergence on a cell is achieved after just one iteration.

The result of function fitting is illustrated in Figure 10. In this example, we sampled an analytical function for which the location of the maximum value was known. The circles (o) represent points where the function was sampled. The star (*) is the actual maximum location, although there is no data point there. We then applied the conical fitting function. The conical function was centered on each local rectilinear grid point in turn, the standard deviation of the fitting error was computed, then the grid position with the lowest standard deviation was selected as the location of the extreme value. The image in Figure 10 is colored by standard deviation: red is high, blue is low. The square shows where the standard deviation was lowest. While this does not exactly coincide with the actual location of the maximum, it is a better estimate than one of the existing data points would have been and demonstrates that subcell resolution can be achieved.

3.5.5 Core Line Termination

The core line extraction algorithm terminates when it hits an aggregate other than the one in which it started, when it hits a domain boundary, or when it has traversed more than a certain number of contiguous unmarked cells. Core line ends that lie in unmarked cells are removed.

3.5.6 Core Line Smoothing

The core line can be made *smoother* by using the points found thus far as the control points of a low-order b-spline. While this may seem like an unnecessary sacrifice of some of the accuracy obtained by function fitting, it is useful. The correction step in the function fitting algorithm was used to keep the core line from deviating from the line-type local extremum of the scalar field and to achieve subcell resolution. However, when we start with a noisy and/or non-smooth solution, the core line produced is also noisy. The b-spline eliminates some of this noise by smoothing. While we want to smooth out high frequency noise, we do not wish to eliminate the curvature of the core line itself. Thus we use a low-order b-sline rather than a high-order b-spline. The b-spline curve becomes the smoothed core line. The tangent of this smoothed core line can then be used as an approximation of the swirl vector in the remaining steps of the method (extent and other characteristics). The tangent is preferable to the vorticity or the real eigenvector whose vector fields may be noisy.

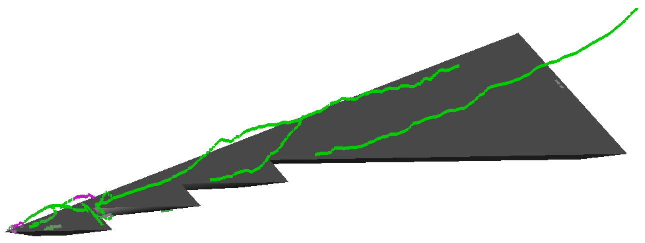

Fig. 11. Vortex core lines are extracted, one line per aggregate.

The core lines extracted are shown in Figure 11. Note that the core lines are continuous and that the topology of the merging vortices is maintained. For this relatively well resolved case, the core line extraction technique works well.

3.6 Vortex Extent: Maximum Tangential Velocity

There are several definitions of vortex extent. We chose the maximum tangential velocity (MTV) definition for simplicity and to avoid ambiguity [8]. We are motivated in this selection by the physical characteristics of an isolated vortex. Figure 12 shows the tangential velocity vs. radius for a wing tip vortex as reported by Dacles-Mariani et al. [5]. Although the influence of the vortex extends beyond this radius, the unambiguous MTV boundary encloses a region that is easily recognizable as a vortex core.

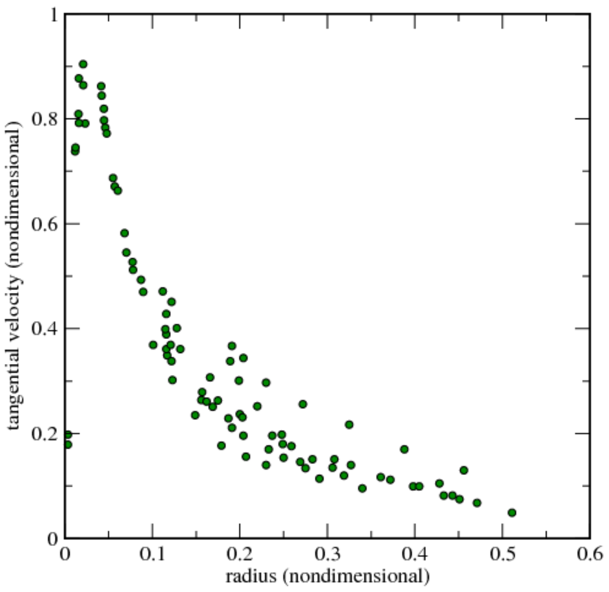

Fig. 12. The vortex core region lies within the radius of maximum tangential velocity: tangential velocity vs. radius for the NACA0012 wingtip vortex, experimental data from Dacles-Mariani et al. [5].

Following Banks and Singer [1], the two dimensional extent curve is found by marching radially outward, independently in each of several directions in the swirl plane, until a maximum of tangential velocity is found. In some cases, such as a vortex near a no-slip boundary, a plateau in the maximum tangential velocity may exist in some directions rather than an extreme value. Detection of these plateaus is described by Jiang et al. [15] and is based on identifying small rates of change in the tangential velocity. Note that vortex extent is affected by vortex-vortex interaction and physical boundaries, thus extent cross sections may not be circular or even elliptical. Vortex-vortex interaction regions pose a challenge to the extent computation algorithm because they may not exhibit the tangential velocity profile shown in Figure 12. It should be emphasized that the MTV surface is not an isosurface of the tangential velocity.

The vortex extent surface is created by repeatedly finding two dimensional extent curves and lofting between them. The extent surfaces are shown in Figure 13. The axial variation in the strength of each vortex is evident by the change in shading on its extent surface.

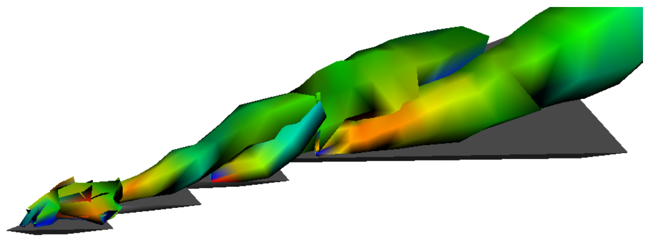
Fig. 13. Vortex extent is found based on the local change in the maximum tangential velocity (MTV). Each extent surface is colored by the local value of the tangential velocity on the extent surface.

3.7 Vortex Visualization

We now have feature level characteristic descriptions of the vortices in the computational domain. The visualizations presented here display geometrical characteristics of vortices such as core line position and vortex extent and kinematical characteristics such as rotation direction and tangential velocity [13]. Since characteristic data files are orders of magnitude smaller than CFD solutions and meshes, these visualizations permit real time interaction.

4 Application to Spinning Missile with Dithering Canards

In this section, we present results for the spinning missile with dithering canards. The rolling of the missile coupled with the canard deflections produce lift forces on the canards. In response to the resulting pressure differential, a counter rotating vortex pair is generated that trails from the inboard and outboard tips of each canard. As the missile rotates and the canards dither, the angle of attack seen by a canard varies and the strengths of its trailing vortices change over time. Additionally, the sense of rotation of each vortex also changes. Under certain conditions, one or more of these vortices may impinge on a tail fin.

The specific case we considered employed the “0% command level” canard schedule as described in [4]. The mesh, described in Table 1 was generated using SolidMesh [7]. The flow solution was computed using the U^{2}NCLE flow simulation code [12] for a Mach number of 1.6, an angle of attack of 3.0 degrees, and a Reynolds number of 41.3×10^6 based on body length. The time step chosen for the solution resulted in a rotation of the missile of one degree per time step.

Figure 14 shows a sequence of images that covers 6 degrees of rotation for the missile. The column of images on the left shows the extracted core lines. The core lines are colored by the sign of the helicity, (+) purple and (-) green to indicate the direction of rotation of the vortex relative to the streamwise velocity. The images in Figure 14 were selected because they show that the vortices trailing from the canard tips change rotation direction. The disturbance in the vortical flow due to this reversal of rotation then propagates down the body and eventually interacts with the tail fins. Although not completely connected, and thus unsuitable for characterization by vortex core line length, the quality of the core line is acceptable given the complexity of the flow field. From the image on the top right, we note that the strength of the primary vortex is maintained along the length of the body as indicated by the reasonably constant green shade. We note from the temporal variation that the strength of the vortex near the canard tips is decreasing, initially, and then increasing as the canard deflects further, as indicated by the red shading near the canard tip in the bottom image. An animation showing the canards completing two dither cycles (an approximate roll angle of 40°) is included in the paper directory. The change in the sense of rotation of each of the canard vortices is clearly shown in the animation. Also evident are remnants of poorly resolved vortices that are shed from the canard posts and traverse the length of the body.

nodes	9,262,283
triangular faces	909,090
tetrahedrons	24,619,310
pyramids	32,753
prisms	9,882,451

Table 1. Description of unstructured hybrid mesh employed for CFD simulation.

The detection and characterization process for the spinning missile requires approximately 33 minutes per time step while running in serial mode on a Sun Enterprise V880 with eight processors (750MHz UltraSPARC III) with 2 GB RAM per processor (16 GB RAM total). Run time and memory requirements are difficult to predict for they depend not only on mesh size, but also on the percentage of cells with complex eigenvalues of the velocity gradient tensor and the number of vortices. We have characterized a data set with more nodes (20 million) than the spinning missile, yet it required only 11 minutes.

5 Conclusion

In this paper, we described a technique to visualize vortices in complex flow fields found in practical CFD solutions computed on unstructured meshes. The novel core line extraction technique with function fitting allows us to resolve vortex core lines in solutions on unstructured meshes. The *k*-means clustering of candidate cell aggregates allows us to identify individual vortices in complicated vortex topology. Our visualizations bring to light interesting and complex feature level vortex behavior, such as the merging of two co-rotating vortices along the serrated wing and the change in vortex rotation direction due to the moving geometry of the spinning missile. Our two contributions make such visualizations possible. In the future, we intend to produce improved core lines by reducing the noise in the swirl vector field, improve core line connectivity, adapt Jiang’s feature level vortex verification [15] to unstructured meshes, and improve extent computation in regions where vortices merge.

Acknowledgements

The authors would like to thank NSF (ITR/NGS-0326386) and DoD for supporting this research. Thanks are also due to Eric Blades of the Computational Simulation and Design Center at Mississippi State University for providing the missile data set.

References

[1] D. C. Banks and B. A. Singer. A Predictor-Corrector Technique for Visualizing Unsteady Flow. *IEEE Trans. Visualization and Computer Graphics*, 1(2):151–163, 1995.

[2] D. Bauer and R. Peikert. Vortex Tracking in Scale-Space. In *Joint Eurographics–IEEE TCVG Sym. Visualization*, pages 233–240, 2002.

[3] C. H. Berdahl and D. S. Thompson. Eduction of Swirling Structure using the Velocity Gradient Tensor. *AIAA J.*, 31(1):97–103, 1993.

[4] E. L. Blades and D. L. Marcum. Numerical simulation of a spinning missile with dithering canards using unstructured grids. *J. Spacecraft and Rockets*, 41(2):248–256, 2004.

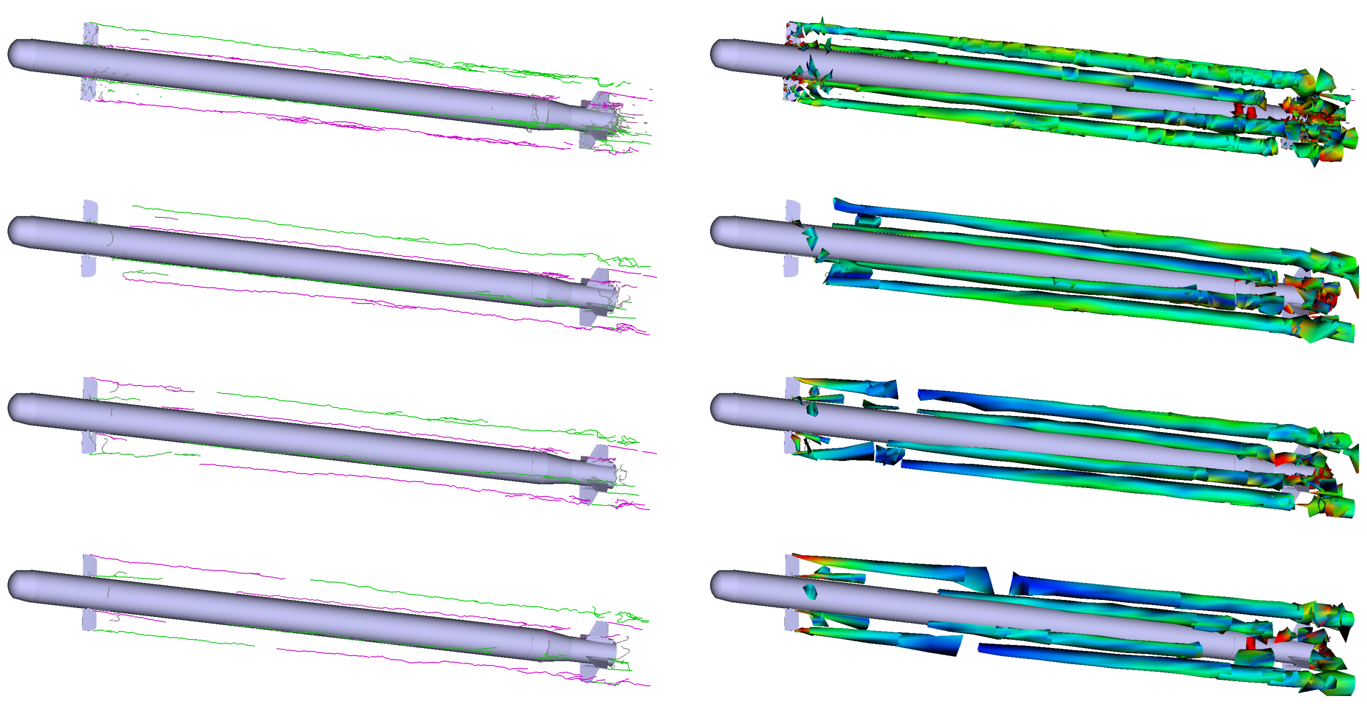

Fig. 14. Core lines and extent for spinning missile with dithering canards for time steps *t*=726, 728, 730, and 732 (top to bottom). In the images on the left, purple indicates a left handed rotation while green indicates a right handed rotation. The right images show vortex extent with surface colored by the local tangential velocity. The scale on the nondimensional tangential velocity is blue (0) to red (0.189).

[5] J. Dacles-Mariani, G. Zilliac, J. Chow, and P. Bradshaw. Numerical/experimental study of a wingtip vortex in the near field. *AIAA J.*, 33(9):1561–1568, Sept. 1995.

[6] V. Fernandez, N. Zabusky, S. Bhat, D. Silver, and S. Chen. Visualization and feature extraction in isotropic navier-stokes turbulence. In *Proc. AVS95 Conference*, 1995.

[7] J. A. Gaither, D. L. Marcum, and B. Mitchell. SolidMesh: A Solid Modeling Approach to Unstructured Grid Generation. In *Proc. International Conference on Numerical Grid Generation in Computational Field Simulations*. Whistler, B.C., Sept. 2000.

[8] C. Garth, X. Tricoche, T. Salzbrunn, and G. Scheuermann. Surface Techniques for Vortex Visualization. In *Joint Eurographics–IEEE TCVG Sym. Visualization*, pages 155–164, 2004.

[9] C. Hammons and D. Thompson. A numerical investigation of novel planforms for micro uavs. In *AIAA 44th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 9-12, 2006*, Jan. 2006.

[10] P. E. Hart, N. J. Nilsson, and B. Raphael. Correction to "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". *SIGART Newsletter*, 37:28–29, 1972.

[11] R. Hooke and T. A. Jeeves. "Direct Search" Solution of Numerical and Statistical Problems. *J. ACM*, 8(2):212–229, 1961.

[12] D. G. Hyams, K. Sreenivas, C. Cheng, W. R. Briley, D. L. Marcum, and D. L. Whitfield. An investigation of parallel implicit solution algorithms for incompressible flows on multielement unstrucutred topologies. In *AIAA paper 2000-0271*, Jan. 2000.

[13] M. Jankun-Kelly, D. Thompson, W. Brewer, M. Jiang, and R. Machiraju. A Multistage Vortex Visualization Algorithm. In *AIAA 44th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 9-12, 2006*, 2006.

[14] J. Jeong and F. Hussain. On the Identification of a Vortex. *J. Fluid Mechanics*, 285:69–94, 1995.

[15] M. Jiang. *A Feature-Based Approach to Visualizing and Mining Simulation Data*. PhD thesis, The Ohio State University, 2005.

[16] M. Jiang, R. Machiraju, and D. S. Thompson. A Novel Approach to Vortex Core Region Detection. In *Joint Eurographics–IEEE TCVG Sym. Visualization*, pages 217–225, 2002.

[17] M. Jiang, R. Machiraju, and D. S. Thompson. Geometric Verification of Swirling Features in Flow Fields. In *IEEE Visualization '02*, pages 307–314, 2002.

[18] M. Jiang, R. Machiraju, and D. S. Thompson. Detection and Visualization of Vortices. In *Visualization Handbook*, pages 287–301. Academic Press, 2004.

[19] T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu. An Efficient k-Means Clustering Algorithm: Analysis and Implementation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.

[20] M. Langbein, G. Scheuermann, and X. Tricoche. An Efficient Point Location Method for Visualization in Large Unstructured Grids. In *Proc. Vision, Modeling, and Visualization '03*, pages 27–36. IOS Press, 2003.

[21] Y. Levy, D. Degani, and A. Seginer. Graphical Visualization of Vortical Flows by Means of Helicity. *AIAA J.*, 28(8):1347–1352, 1990.

[22] K.-L. Ma and V. Interrante. Extracting Feature Lines from 3D Unstructured Grids. In *IEEE Visualization '97*, pages 285–292, 1997.

[23] J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Fifth Berkeley Symp. Math. Statistics and Probability*, volume 1, pages 281–296, 1967.

[24] R. Peikert and M. Roth. The "Parallel Vectors" Operator–A Vector Field Visualization Primitive. In *IEEE Visualization '99*, pages 263–270, 1999.

[25] A. E. Perry and M. S. Chong. A Description of Eddying Motions and Flow Patterns using Critical Point Concepts. *Ann. Rev. Fluid Mechanics*, 19:125–155, 1987.

[26] M. Roth. *Automatic Extraction of Vortex Core Lines and Other Line-Type Features for Scientific Visualization*. PhD thesis, Swiss Federal Institute of Technology Zürich, 2000.

[27] M. Roth and R. Peikert. A Higher-Order Method for Finding Vortex Core Lines. In *IEEE Visualization '98*, pages 143–150, 1998.

[28] D. Silver and X. Wang. Tracking Scalar Features in Unstructured Datasets. In *IEEE Visualization '98*, pages 79–86, 1998.

[29] S. Stegmaier, U. Rist, and T. Ertl. Opening the Can of Worms: An Exploration Tool for Vortical Flows. In *IEEE Visualization '05*, pages 463–470, 2005.

[30] D. Sujudi and R. Haimes. Identification of Swirling Flow in 3D Vector Fields. In *AIAA 12th Computational Fluid Dynamics Conf., Paper 95-1715*, 1995.

[31] T. van Walsum, F. H. Post, D. Silver, and F. J. Post. Feature Extraction and Iconic Visualization. *IEEE Trans. Visualization and Computer Graphics*, 2(2):111–119, 1996.