IDENTIFICATION OF SWIRLING FLOW IN 3-D VECTOR FIELDS

David Sujudi* and Robert Haimes**
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

An algorithm for identifying the center of swirling flow in 3-D discretized vector fields has been developed. The algorithm is based on critical point theory and has been implemented as a visualization tool within pV3, a package for visualizing 3-D transient data. The scheme works with gridding supported by pV3: structured meshes as well as unstructured grids composed of tetrahedra, polytetrahedral strips, hexahedra, pyramids, and/or prism cells. The results have been validated using artificially-generated vector fields and 3-D CFD data.

Introduction

This work is motivated by the need to easily locate vortices in large 3-D transient problems. A tool that will automatically identify such structures is definitely needed to avoid the time-consuming and tedious task of manually examining the data. However, the question of what defines a vortex raises considerable confusion. As a result, various definitions have been proposed by investigators, including, among others, Moin and Kim [1] [2], Villasenor and Vincent [3], Globus, Levit, and Lasinski [4], Banks and Singer [5] [6], and Jeong and Hussain [7]. The reader can refer to [6] for a survey of the schemes that have been developed and [7] for comparisons of some of them

We see the need for a tool that *can help* investigators locate vortices, and yet has a familiar and intuitive interpretation. Due to the confusion on what constitutes a vortex, we relax the condition that what the tool finds must be vortices. However, the scheme must be practical, in terms of computational speed and usability. We must also be able to easily integrate it into a publicly-available visualization package, particularly pV3 [8], a scientific visualization software designed for co-processing and distributed computation (i.e., the user can visualize the solution as it is being computed on one or more computers).

We believe that a tool that identifies the center of swirling flows satisfies the above need and criteria. Investigators have been using swirling flow as one of the means to locate vortices in 3-D discretized vector fields. Swirling flows in a 3-D field are usually identified by studying vector fields that are mapped onto planar cuts or by seeding streamlines. These procedures can be very laborious, especially for large and complex flows. The scheme presented here allows automatic identification of the center of swirling flows in 3-D vector fields.

The algorithm for implementing this tool is based on critical-point theory. As will be described below, the scheme works on a cell by cell basis, lending itself to parallel processing, and is flexible enough to work with the various types of grids supported by pV3. By employing this method, we have avoided the need for curve integration (which is needed for visualization tools such as streamlines and particle paths). Curve integration is a serial operation and can not readily take advantage of pV3's distributed computing capabilities. And, as described in [9], integrations across a distributed environment involve passing information between machines and other additional complexities which further reduce efficiency.

In the next sections, we will describe the theoretical background of this algorithm and how it is implemented. The results of the scheme on exact artificially-generated data as well as CFD data will be shown and its performance will be discussed. Our algorithm will also be compared (using artificially-generated data) against a tool developed by Globus, Levit,and Lasinski [4].

Theory

Critical points are defined as points where the streamline slope is indeterminate and the velocity is zero relative to an appropriate observer [10].

---

* Graduate Research Assistant, Member AIAA
** Principal Research Engineer, Member AIAA

Copyright © 1995 by the American Institute of Aeronautics and Astronautics, Inc.
All rights reserved.

According to critical point theory, the eigenvalues and eigenvectors of the rate-of-deformation tensor, $\partial u_i/\partial x_j$ (we'll call this matrix A), evaluated at a critical point defines the flow pattern about that point. Specifically, if A has one real and a pair of complex-conjugate eigenvalues the flow forms a spiral-saddle pattern, as illustrated in Figure 1. The eigenvector corresponding to the real eigenvalue points in the direction about which the flow spirals, and consequently, the plane normal to this eigenvector defines the plane on which the flow spirals. For a complete description of all other possible trajectories the reader can refer to [10] or [11].

Since we intuitively identify the pattern in Figure 1 as swirling flow, we can use the above method to find the center of swirling flows located at critical points. However, there are obviously swirling flows whose center is not at a critical point. Fortunately, a similar method can be applied in these cases.

At a non-critical point with the necessary eigenvalue combination (i.e., one real and a pair of complex conjugates) the velocity in the direction of the eigenvector corresponding to the real eigenvalue is subtracted. The invariance of the eigenvectors' directions with respect to a Galilean transformation ensures that the resulting flow will have the same principal directions. We'll call the resulting velocity vector the reduced velocity. If the reduced velocity is zero, then the point must be at the center of the swirling flow. A similar statement was also made by Vollmers, Kreplin, and Meier [12].

Therefore, to find a point at the center of a local swirling flow, we look for a point whose rate-of-deformation tensor has one real and a pair of complex-conjugate eigenvalues and whose reduced velocity is zero.

Implementation

pV3 accepts structured and/or unstructured grids (containing any combination of tetrahedra, polytetrahedra strips, hexahedra, pyramids, and prism cells). In the interest of efficiency, we have decided to use only tetrahedral cells, with all other cell types reduced to 2 or more tetrahedra. Figure 2 shows how various types of cells are decomposed into tetrahedral cells.

This approach enables us to use a simple linear interpolation for the velocity, avoiding the more complex, and inherently more costly, interpolation required by other types of cells (such as bilinear interpolation for hexahedra). A tetrahedron has 4 node points, sufficient to solve for the four coefficients of a 3-D linear interpolant. More importantly, linear velocity interpolation produces a constant rate-of-deformation tensor within the entire tetrahedral cell. Consequently, the straight forward algorithm described below can be employed, which otherwise would not have been possible.

The algorithm proceeds one tetrahedral cell at a time, and can be summarized as follows (it is assumed that a velocity vector is available at each node):

1. Linearly interpolate the velocity within the cell.
2. Compute the rate-of-deformation tensor A. Since a linear interpolation of the velocity within the cell can be written as

$$u_i = C_i + \frac{\partial u_i}{\partial x}\Delta x + \frac{\partial u_i}{\partial y}\Delta y + \frac{\partial u_i}{\partial z}\Delta z \qquad (1)$$

   then A can be constructed from the coefficients of the linear interpolation function of the velocity vector.
3. Find the eigenvalues of A. Processing continues only if A has one real (λ_R) and a pair of complex-conjugate eigenvalues (λ_C).
4. At each node of the tetrahedron, subtract the velocity component in the direction of the eigenvector corresponding to λ_R. This is equivalent to projecting the velocity onto the plane normal to the eigenvector belonging to λ_R, and can be expressed as

$$\vec{w} = \vec{u} - (\vec{u}\cdot\vec{n})\vec{n} \qquad (2)$$

   where $\vec{n}$ is the normalized eigenvector corresponding to λ_R, and w is the reduced velocity.
5. Linearly interpolate each component of the reduced velocity to obtain

$$w_i = a_i + b_i x + c_i y + d_i z \qquad (3)$$
$$i = 1, 2, 3$$

6. To find the center, we set w_i in equation (3) to zero. Since the reduced velocity lies in a plane, it has only 2 degrees of freedom. Thus, only 2 of the 3 equations in equation (3) are independent. Any 2 can be chosen as long as their coefficients are not all zero. Now we have

$$0 = a_i + b_i x + c_i y + d_i z \qquad (4)$$
$$i = 1, 2$$

which are the equations of 2 planes, whose solution (the intersection of 2 planes) is a line.

7. If this line intersects the cell at more than 1 point, then the cell contains a center of a local swirling flow. The center is defined by the line segment formed by the 2 intersection points.

Since the 2 intersection points lie on the line found in step 6, the reduced velocity at those points must be zero. This suggests a different (but equivalent) and more efficient way to finding the center. This approach renders steps 5, 6, and 7 unnecessary and replaces them with a new step 5:

5. For each of the tetrahedron's face, determine if there is exactly 1 point on the face where the reduced velocity is zero. If at the end there are exactly 2 distinct points, then the cell contains a center, which is defined by those 2 points.

Both approaches have been tried on our test cases with identical results. Therefore, the second approach is implemented.

Results and Comparisons

The algorithm is first tested on artificially-generated vector fields where the location of the center of the swirling flow is known exactly. The field is defined by

$$v_x = y - y_c \,,\; v_y = x - x_c \,,\; v_z = f(z) \qquad (5)$$

Note that this vector field has circular streamlines (in the x-y plane) around a central axis whose location is defined by x_c and y_c. The magnitude of the vector is equal to the distance from the central axis. The field is discretized using an 11 x 11 x 11 node structured grid. The results for various values of x_c, y_c, and functional forms of v_z (including constant, linear, and exponential) are studied and determined to be correct. A sample result (with $x_c = 2.2\,\Delta x$, $y_c = 1.5\,\Delta y$, and $v_z = 1$) is shown in Figure 3. A streamline is also shown in this figure to provide a sense of the swirling vector field.

Further tests are done using data from 3-D calculations of flow past a tapered cylinder [13] and of flow over an F-117 fighter at an angle of attack [14]. The tapered-cylinder calculation employs structured grid, while the F-117 case uses unstructured grid composed of tetrahedra. The size of these data sets and the time needed to find the swirl flow centers are summarized in table 1.

Table 1 Size of Test Cases

Case	Number of Nodes	Number of Cells	Number of Tetrahedral Cells*	Calculation Time (sec.)**
Cylinder	131072	123039	738234	34
F-117	48518	240122	240122	16

* After decomposition (if needed) of original cells.
** On SGI Indigo2 with MIPS R4400 150 Mhz CPU.

Results are shown in Figures 4 and 5. To indicate the existence swirling flow, streamlines have been spawned near the centers found by the algorithm. These results indicate that the large coherent structures found by the algorithm do indeed correspond to centers of swirling flow. However, the algorithm does not find all the swirling flow in the tapered cylinder data. Missing are a few swirling flow structures further downstream of the cylinder, which are found by studying the vector field more closely. We believe the size of the grid cells might be a factor. The cells are larger away from the cylinder, reducing the accuracy in the calculation of the rate-of-deformation tensor (and consequently the reduced velocity). Another possible cause is the algorithm's sensitivity to the strength of the swirl flow. As shown in Figures 4b and 4c, the structures are very coherent for strong swirls (i.e., the swirl velocity is larger than or comparable to the normal velocity). However, the structures start to break up as the swirl weakens, and further downstream, where the swirl flows are very weak, the algorithm finds no coherent structures.

In the case of the F-117 data, the structures are less coherent than in the tapered cylinder. The tetrahedral grid used in this data is very irregularly sized, and is rather coarse. Comparison between the streamlines in Figures 4c and 5 also shows that the swirl in the F117 data is noticeably weaker. Both of these factors might contribute to the incoherency in the structures.

We have also compared our results with that of FAST's vortex-core finder [4] [15]. FAST's finder defines a vortex core by integrating from a critical point in the direction of the eigenvector corresponding to the only real eigenvalue of the rate-of-deformation tensor. For this comparison, we use 3 artificially-generated data sets, each bounded by a cube containing 3 randomly-placed vortices. The data generator is developed by D. Asimov at NASA Ames Research Center.

The comparisons are shown in Figures 6a to 6f. Despite the lack of any 3-dimensional cues, the curves

in these figures do exist in 3-D space, and each pair of figures are taken from the same view point. A high degree of similarities are found in each case except for the middle curves in each data set, where FAST produces curves that are either longer and/or has different orientation. Closer inspection of the data shows that pV3's results are the correct ones, while FAST's curve integrations veer away from the core.

Conclusion

An algorithm to automatically locate the center of swirling flow in 3-D vector fields has been developed and implemented as part of the pV3 visualization package. By employing cell-by-cell processing and using only tetrahedral cells (and transforming other cell types to tetrahedra), the scheme can take advantage of pV3's distributed environment and the simplification from using linear interpolation. Although we believe the strength of the swirl flow and the coarseness of the grid can affect the degree of accuracy and coherency of the results, tests using artificially-generated vector fields and 2 different CFD data have shown that the coherent structures found by the algorithm are indeed centers of swirling flow. Comparisons with FAST vortex-core finder also show high degrees of similarities in the results.

Acknowledgements

The authors would like to thank Al Globus and Creon Levit of NASA Ames for their inputs and suggestions, as well as their eigen solver. Dan Asimov of NASA Ames provided the data-set generator used for generating the comparison data.

This work was funded by NASA Ames Research Center (Tom Woodrow and Michael J. Gerald-Yamasaki, technical monitors) and United Technologies Research Center (Dave Edwards, technical monitor).

References

[1] P. Moin and J. Kim, "The Structure of the Vorticity Field in Turbulent Channel Flow. Part 1. Analysis of Instantaneous Fields and Statistical Correlations," J. Fluid Mech. 155, pp. 441, 1985.

[2] J. Kim and P. Moin, "The Structure of the Vorticity Field in Turbulent Channel Flow. Part 1. Study of Ensemble-Averaged Fields," J. Fluid Mech. 162, pp. 339, 1986.

[3] J. Villasenor and A. Vincent, "An Algorithm for Space Recognition and Time Tracking of Vorticity Tubes in Turbulence," CVGIP: Image Understanding 55:1, pp. 27, 1992.

[4] A. Globus, C. Levit, and T. Lasinski, "A Tool for Visualizing the Topology of Three-Dimensional Vector Fields," Report RNR-91-017, NAS Applied Research Office, NASA Ames Research Center, 1991.

[5] D. C. Banks and B. A. Singer, "A Predictor-Corrector Scheme for Vortex Identification," ICASE Report NO. 94-11, NASA CR-194882, 1994.

[6] D. C. Banks and B. A. Singer, "Vortex Tubes in Turbulent Flows: Identification, Representation , Reconstruction," ICASE Report No. 94-22, NASA CR-194900, 1994.

[7] J. Jeong and F. Hussain, "On the Identification of a Vortex," J. Fluid Mech. 285, pp. 69, 1995.

[8] Robert Haimes, "pV3: A Distributed System for Large-Scale Unsteady CFD Visualization," AIAA Paper 94-0321, 1994.

[9] D. D. Sujudi and R. Haimes,"Integration of Particle Paths and Streamlines in a Spatially-Decomposed Computation," Parallel CFD Conference, Pasadena, CA, June 26 - 28, 1995.

[10] M.. S. Chong, A. E. Perry, and B. J. Cantwell, "A General Classification of Three-Dimensional Flow Fields," Phys. Fluids A, vol. 2, pp. 765-777, May 1990.

[11] R. H. Abraham and C. D. Shaw, *Dynamics: The Geometry of Behavior*, parts 1-4, Ariel Press, Santa Cruz, CA., 1984.

[12] H. Vollmers, H. P. Kreplin, H. U. Meier, "Separation and Vortical-Type Flow around a Prolate Spheroid. Evaluation of Relevant Parameters," AGARD Conference Proceedings No. 342.

[13] Dennis Jespersen and Creon Levit, "Numerical Simulation of Flow Past a Tapered Cylinder," AIAA Paper 91-0751, 1991.

[14] S. A. Vermeersch, "Investigation of the F117A Vortical Flow Characteristics," Master's Thesis, Massachussetts Institute of Technology, Department of Aeronautics and Astronautics, May 1993.

[15] G. V. Bancroft, F. J. Merrit, T. C. Plessel, P. G. Kelaita, R. K. McCabe, and A. Globus,"FAST: A Multi-processing Environment for Visualization of Computational Fluid Dynamics", Proceedings of Visualization '90, San Francisco, CA, Oct. 1990.

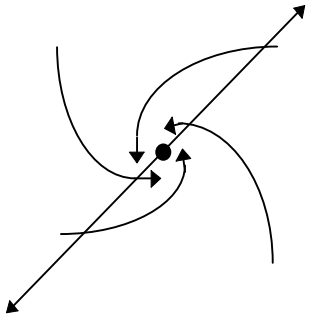

Figure 1 Flow pattern at a critical point whose rate-of-deformation tensor has 1 real and a pair of complex-conjugate eigenvalues.

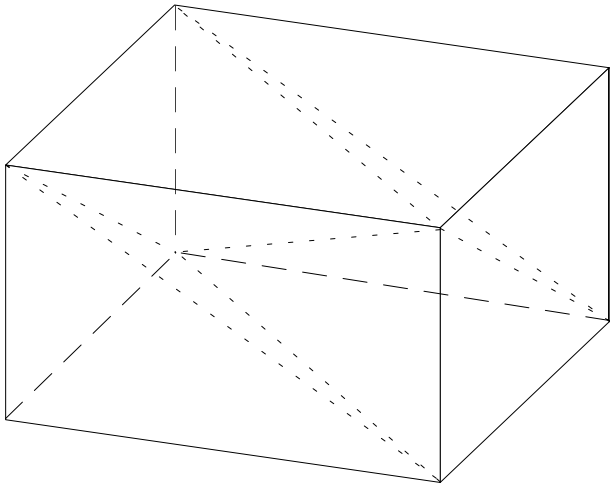

Figure 2a A hexahedron (or structured-grid cell) divided into 6 tetrahedra.

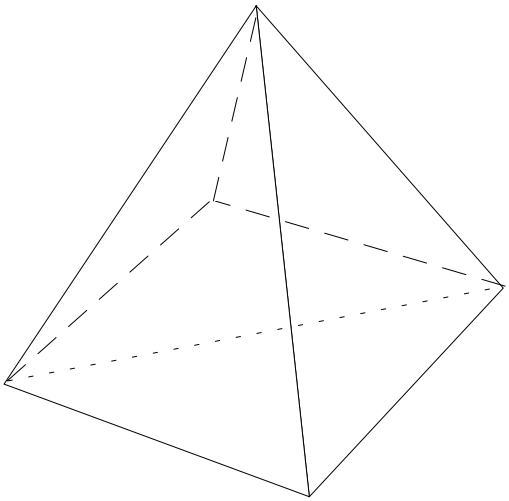

Figure 2c A pyramid cell divided into 2 tetrahedra.

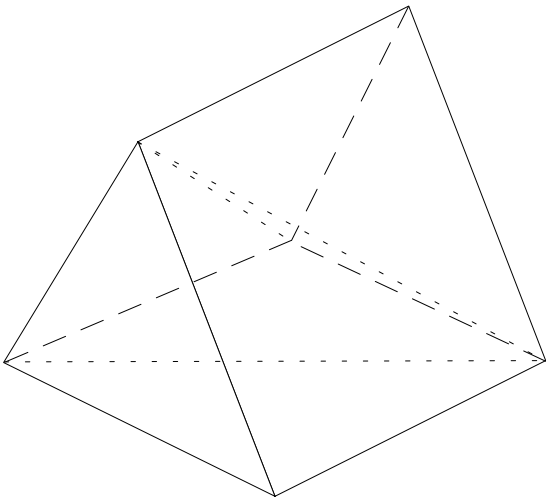

Figure 2b A prism cell divided into 3 tetrahedra.

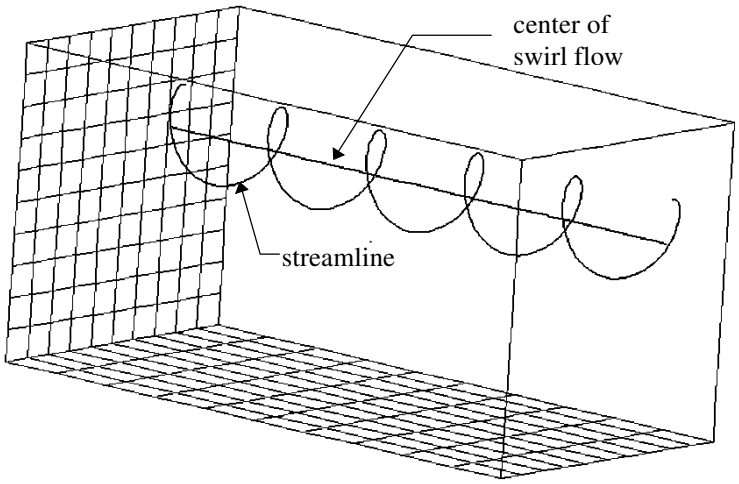

Figure 3 A sample result of an artificially-generated test case.

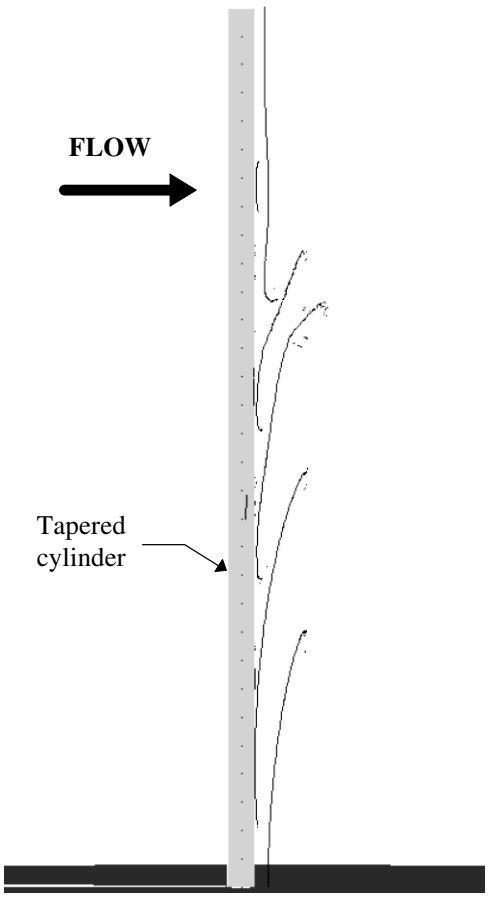

Figure 4a Flow past a tapered cylinder. Shown are swirl flow centers found by the algorithm.

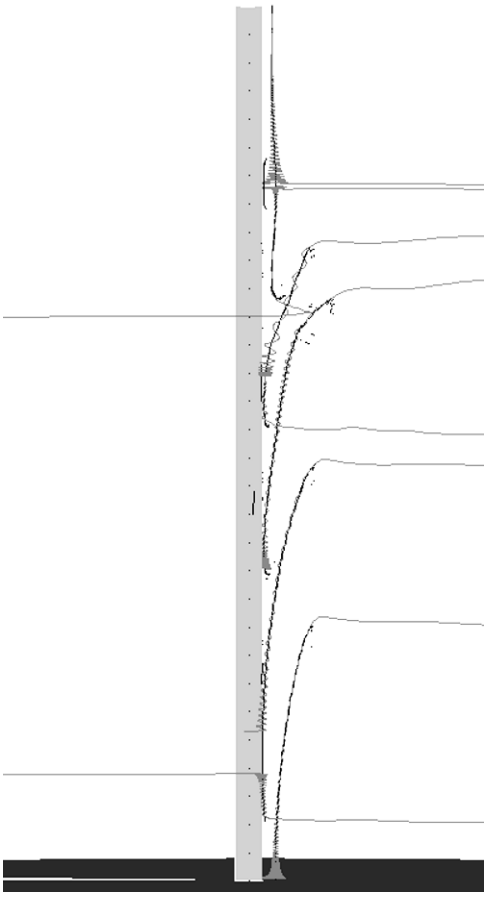

Figure 4b flow past a tapered cylinder. Swirl flow centers and streamlines are shown.

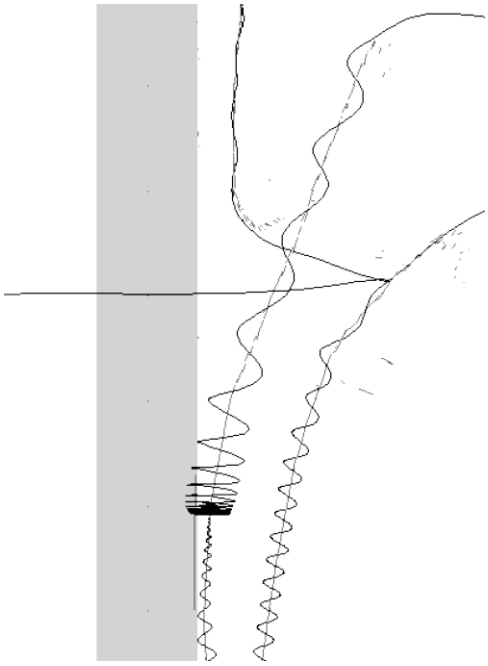

Figure 4c Blow up of figure 4b.

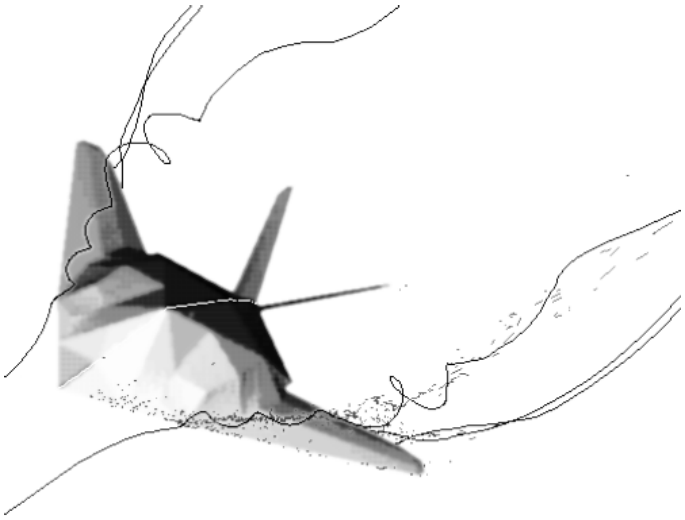

Figure 5 Flow over a F117 fighter.
Swirl flow centers and streamlines are shown. Note: The centers are not mirrored.

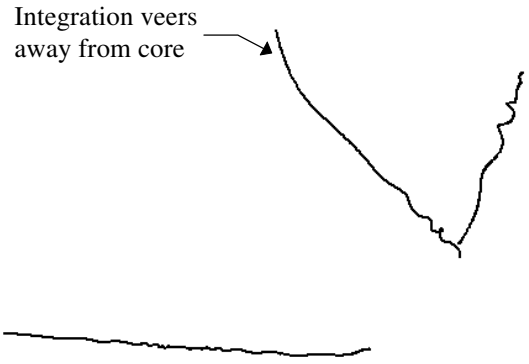

Figure 6a Result of FAST vortex core finder on data set 1.

Figure 6b Result of pV3 swirl flow finder on data set 1.

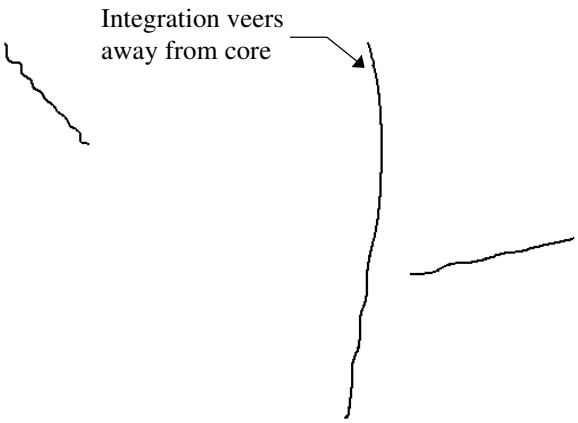

Figure 6c Result of FAST vortex core finder on data set 2.

Figure 6d Result of pV3 swirl flow finder on data set 2.

Figure 6e Result of FAST vortex core finder on data set 3. Note that a streamline has also been spawned here to indicate that the twist at the top of the middle curve is actually outside the vortex.

Figure 6f Result of pV3 swirl flow finder on data set 3.