

Automatic Tip Vortex Core Profiling for Numerical Flow Simulations of Rotorcraft in Hover

David L. Kao^{*} and Neal M. Chaderjian[†]
NASA Ames Research Center, Moffett Field, CA 94035

An automated approach is presented that extracts visual and quantitative data from vortex cores produced by Navier-Stokes simulations of rotorcraft in hover mode. This approach extracts contiguous rotor tip vortex-core trajectories, cross-flow velocity profiles, and vortex-core diameter variation with wake age (azimuth angle). This automated approach is faster and more accurate than a conventional manual approach. Moreover, this new approach allows for an efficient way to quantitatively compare vortex-core profiles from different flow simulations, e.g., grid resolution studies, and validate computed results with experimental data.

I. Introduction

Computational Fluid Dynamic (CFD) simulation of rotorcraft aeromechanics is a challenging problem. These multidisciplinary applications combine moving body aerodynamics, structural dynamics of deforming rotor blades, and flight dynamics with controls. Moreover, multiple time and length scales result in smaller time steps and highly refined grids, which require large multi-core compute systems with long simulation times. There is a considerable research effort and interest in improving the numerical simulation accuracy of rotor blade vortices and their wakes, using the time-dependent, Reynolds-averaged, Navier-Stokes (RANS) equations.¹ A key element in understanding and validating these numerical simulations is the ability to extract visual and quantitative data and compare the results with experimental measurements.

Although there are several commercial software packages that provide vortex core extraction and visualization, most of them lack the capability to provide an automated flow analysis of the extracted vortex cores. Often, users need to perform some ad hoc steps for an in-depth vortex-core analysis. For example, in order to plot the cross-flow velocity profile of a vortex core, the user needs to locate the vortex-core center, which can be determined by examining the contours of a scalar field (e.g., vorticity magnitude) on a cutting plane. Next, the user would mark a line across the vortex core to plot the velocity component perpendicular to the line. As the flow field becomes more complex, e.g., several rotor blades, descending flight, blade/wake and wake/wake interactions, the ability to manually extract quantitative data is formidable.

The goal of the present work is to develop an automated approach to analyze vortex cores from rotorcraft simulations while the vehicle is in hover. The analysis includes the visualization of vortex-core lines, plots of the cross-flow velocity component from the vortex core, and a plot of the vortex-core diameter versus wake age. The grid and solutions files are assumed to be overset structured grids.

The present approach consists of four main steps: automatic vortex-core extraction (Section II.A), continuous vortex-core line generation (Section II.B), extraction of the cross-flow velocity profile (Section II.C), and extraction of the vortex-core diameter as a function of wake age (Section II.D). This new approach was demonstrated with two grid resolution studies. The results from these two case studies are presented in Section III. Future work and concluding remarks are made in Section IV.

II. Approach

The current approach relies on several algorithms described in the following subsections. First, a vortex-core line algorithm is used to extract individual vortex-core lines from the given flow field. Each vortex-core line segment lies in a specific grid cell. Depending on the complexity of the flow field, a large number of vortex line segments may be created from the vortex-core extraction algorithm. An algorithm is presented to connect these vortex line

^{*} Researcher, NASA Advanced Supercomputing Division, M/S N258-5, AIAA Member.

[†] Research Scientist, NASA Advanced Supercomputing Division, M/S N258-2, Associate Fellow AIAA.

segments into continuous vortex-core lines. Next, a straightforward approach is used to extract the cross-flow velocity profiles at several wake ages along the vortex core. The wake age is defined as the azimuthal angle along the vortex core. Finally, a plot of the vortex-core diameter versus wake age is created using an algorithm that computes the vortex-core diameter at several wake age locations.

A. Vortex-Core Extraction

There has been considerable work in flow feature extraction algorithms, especially for vortex-core detection. Most of these algorithms can be classified into two major categories: region-based and line-based.² In the region-based algorithms, a scalar field and a threshold value are specified to identify a spatial region where the vortex exists. Several scalar quantities have been proposed, including $\lambda-2$ ³ and the Q-Criterion.⁴ Although it is straightforward to implement region-based algorithms, the selection of the threshold value is not always straightforward. Line-based algorithms are often preferred because a user-specified criterion is not required. In this paper, a line-based algorithm is used to locate the vortex cores that begin at the rotor blade tips.

Sujudi and Haimes⁵ developed a *tetrahedral cell-based* algorithm to identify the center of a swirling flow in a given 3D vector field. Their algorithm is based on critical point theory, in which an eigenvalue analysis is performed on the Velocity Gradient Tensor (VGT) at critical points to classify the local flow topology. Given a CFD grid, which could be either structured or unstructured, their algorithm decomposes each grid cell into tetrahedral cells. Then, the algorithm processes each tetrahedral cell independently to determine if it contains a vortex-core line segment. This is achieved by performing the following steps for each tetrahedral cell:

- (1) compute ∇V , the gradient of the velocity field V ,
- (2) find the eigenvalues of ∇V ,
- (3) if ∇V has one real and a pair of complex-conjugate eigenvalues, then perform the following steps to locate the vortex-core line; otherwise, proceed to the next tetrahedral cell,
 - (3.1) for each node of the tetrahedron, compute the reduced velocity V_r (crossflow velocity) by subtracting from V the velocity component in the direction of the real eigenvector,
 - (3.2) compute line I_r where the reduced velocity is zero,
 - (3.3) if I_r intersects the tetrahedron twice, i.e., two faces, then the cell contains a center of local swirling flow and the vortex-core line segment is formed by the two intersection points.

The Sujudi/Haimes cell-based algorithm generates a set of individual short line segments that collectively represent vortex-core lines. Because each tetrahedral cell has its own VGT, the vortex-core line segments in general do not match at common cell faces. Hence, the resulting vortex-core lines do not appear contiguous. Figure 1a depicts the non-contiguous vortex-core lines computed for an isolated V22 Osprey rotor/simplified-hub using this cell-based algorithm. To address this issue, a *face-based* approach was proposed by Haimes and Kenwright.⁶ This approach first computes ∇V at each grid node. Then, the following steps are performed for each face of the tetrahedral cell:

- (1) average the node VGTs from the face's three vertices to produce ∇V for that face. This guarantees that the same VGT is used on the common face shared by two adjacent tetrahedral cells,
- (2) find the eigenvalues of ∇V ,
- (3) if ∇V has one real and a pair of complex-conjugate eigenvalues, then compute the reduced velocity V_r (crossflow velocity) by subtracting from V the velocity component in the direction of the real eigenvector,
- (4) compute the line I_r where the reduced velocity is zero,
- (5) if I_r intersects the face, then a vortex-core line cuts through the face. Save the intersection point.

Next, for each tetrahedral cell, if two of its faces contain intersection points, then save the line that passes through these two points. The face-based algorithm generates a set of vortex-core line segments whose end points match at adjoining cell faces. Figure 1b depicts vortex-core lines computed using the face-based approach. Compared to the vortex-core lines shown in Fig. 1a, where the lines do not match at the adjoining cell faces, the face-based results in Fig. 1b look more distinct. However, because there is no connectivity information, each vortex line segment is drawn independently, and the vortex lines are still non-contiguous. Figures 2a and 2b respectively show a close-up view the vortex-core lines generated from the cell-based and face-based approaches.

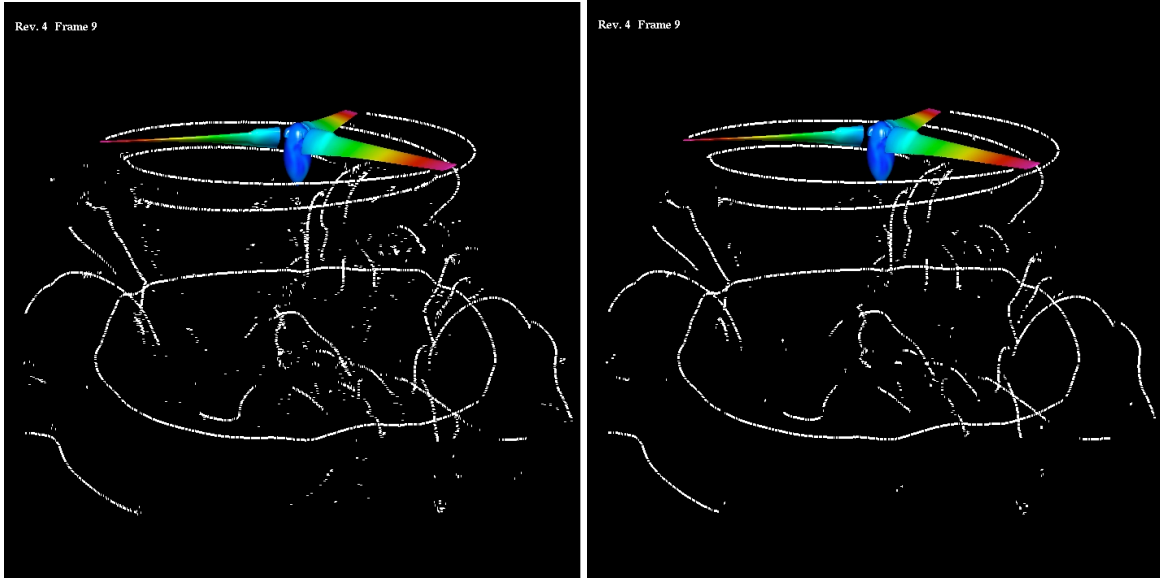


Figure 1. Vortex cores extracted using (a) cell-based approach and (b) face-based approach.

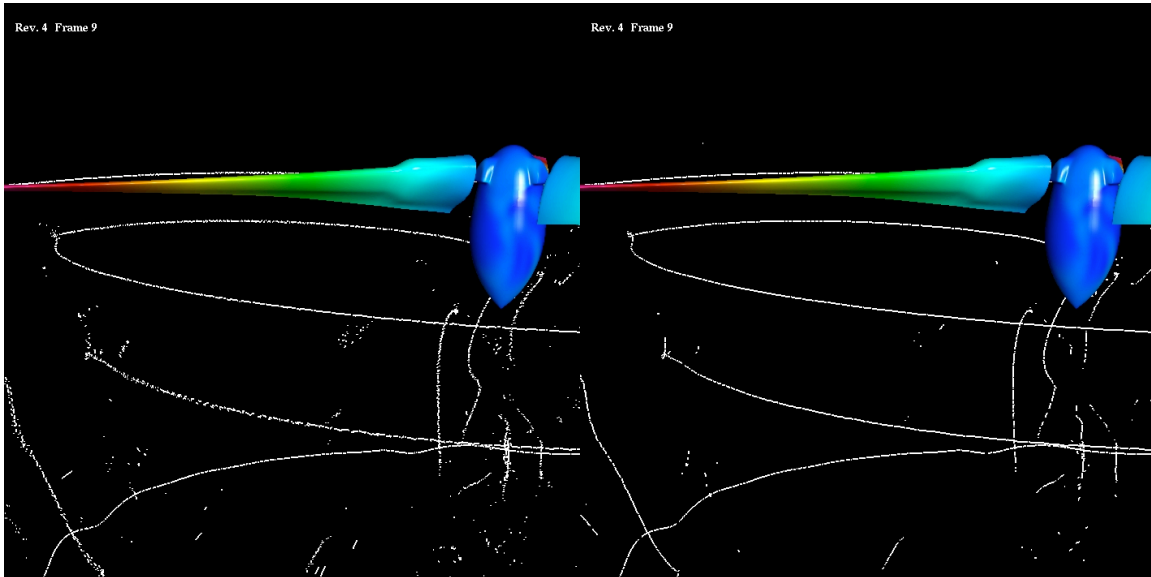


Figure 2. Close-up view of vortex core extracted using (a) cell-based approach and (b) face-based approach.

B. Connected Vortex-Core Lines

The output of the face-based approach described in the previous section is a list of non-connected line segments, and the connectivity information regarding neighboring line segments is unknown. It is possible to connect all line segments that originate from the same rotor blade tip. This would render an image that is more easily viewed when the blade tip vortices undergo several azimuthal rotations (large wake ages). These connected vortex lines also provide a straightforward means to extract and analyze visual and quantitative data for a specific rotor blade.

During rendering, only vortex-core lines that meet a minimum length criterion are displayed. This provides a vortex-core image that is less cluttered with short line segments, see Figs. 1-2.

In order to create a set of connected vortex-core lines, it is currently assumed that the input grids are structured overset grids with hexahedral cells. This simplifies the joining of vortex line segments into a contiguous vortex-core line. However, in principle, an unstructured grid system can be used if the grid-cell connectivity information is available, e.g., from the flow solver.

In the current method, the tetrahedral decomposition described by Kenwright and Lane⁷ is used. Each hexahedral cell in a structured grid is decomposed into five tetrahedral cells. This decomposition is performed prior to applying the face-based vortex-core extraction algorithm. A hexahedron can be divided into either five or six tetrahedra. The five tetrahedra decomposition is used to minimize storage requirements and improve computational efficiency.

There are two possible ways to divide a hexahedron into five tetrahedra, see Fig. 3. In order for adjacent tetrahedral cells to have matching faces, the hexahedral parent cells must be divided differently. Given the node indices (i,j,k) of a hexahedral cell, if the sum of the indices $(i+j+k)$ is even, then an even configuration is used; otherwise, an odd configuration is used. The use of this decision sum ensures that the correct configuration is used so that adjacent cell faces match.

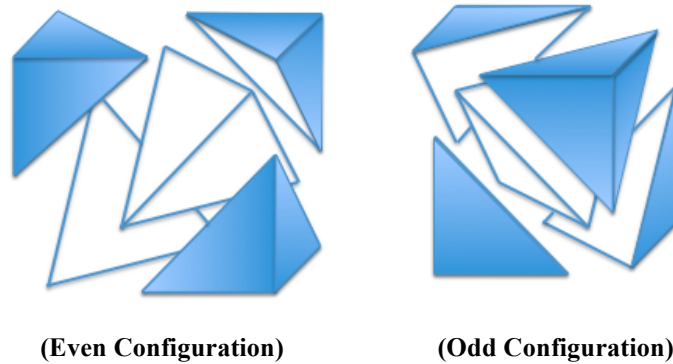


Figure 3. A hexahedron can be divided into a minimum of 5 tetrahedra. For a hexahedron node $(i+j+k)$, whose indices sum to an even number, the even configuration is used. Otherwise, the indices sum to an odd number and the odd configuration is used.

The approach for generating connected vortex-core lines is now described. The input is a set of unconnected line segments and the output is a set of connected lines. Each unconnected line segment lies entirely inside a tetrahedron and its end points reside on the faces of the tetrahedron, see Fig. 4. The following information is stored for each vortex-core line segment: the positions of the two endpoints, the node indices of the tetrahedron containing the line segment, the tetrahedron number and the face number of each endpoint. Given a line segment, the task is to determine the neighboring line segment (if any) at each endpoint. Hence, for each endpoint, it is necessary to determine the node indices of its neighboring tetrahedron and the matching face number of that tetrahedron. Then the neighboring tetrahedron is checked to see if it contains a vortex-core line segment.

Two lookup tables are created to facilitate the process of finding adjacent tetrahedral cells. Table 1 corresponds to an even tetrahedral decomposition and Table 2 to an odd tetrahedral decomposition. In the lookup tables, the first column is the tetrahedron number, the second column is the face number, and the third column is the vertex indices of the face. Since (i,j,k) is the node index of the current hexahedral cell, then the neighboring hexahedral cell is found by changes to the i , j , and k indices found in column 4. Lastly, the tetrahedral number and the face number of the neighboring hexahedron are listed in columns 5 and 6. Tables 1 and 2 correspond to all even and odd configurations, respectively. They only need to be created once for the entire process.

Figure 4 shows an example of a vortex line segment that lies in tetrahedron #1 (of an even configuration decomposition) and its one endpoint resides on face #2 (vertices 1, 7, and 3). This information is highlighted in yellow in Table 1. Column 4 of Table 1 is used to determine the node indices of the neighboring hexahedron, given by $(i+1,j,k)$ in this example. Within the neighboring hexahedron, the vortex line will intersect tetrahedron #2 on face #0 (of an odd configuration). Table 2 contains information for the neighboring odd hexahedron decomposition. Data specific to this example are once again highlighted in yellow.

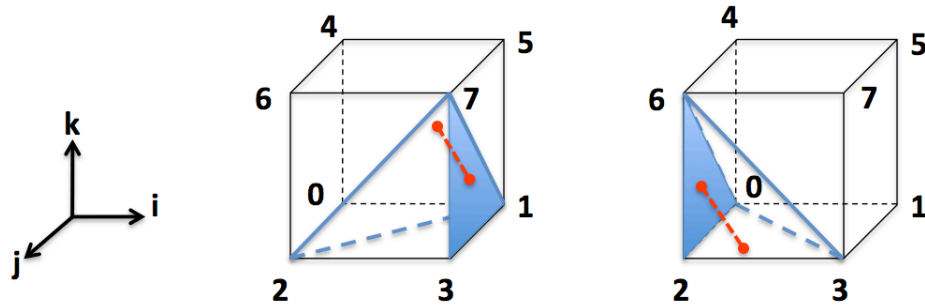


Figure 4. A vortex line segment in tetrahedron #1 (vertices 2, 3, 1 and 7), face #2 (even configuration) connects to another vortex line segment in tetrahedron #2 (vertices 2, 3, 0 and 6), face #0 (odd configuration). The common faces are shaded.

Current Tetrahedron			Neighboring Tetrahedron		
Tet #	Face #	Vertex Indices	Changes to Hexahedral Node Indices	Tet #	Face #
0	0	0, 1, 2	0, 0, -1	3	2
	1	1, 2, 4	0, 0, 0	4	0
	2	2, 4, 0	-1, 0, 0	1	0
	3	4, 0, 1	0, -1, 0	2	3
1	0	3, 2, 1	0, 0, -1	0	2
	1	2, 1, 7	0, 0, 0	4	3
	2	1, 7, 3	1, 0, 0	2	0
	3	7, 3, 2	0, 1, 0	1	3
2	0	5, 4, 7	0, 0, 1	1	2
	1	4, 7, 1	0, 0, 0	4	2
	2	7, 1, 5	1, 0, 0	3	0
	3	1, 5, 4	0, -1, 0	0	3
3	0	6, 7, 4	0, 0, 1	2	2
	1	7, 4, 2	0, 0, 0	4	1
	2	4, 2, 6	-1, 0, 0	0	0
	3	2, 6, 7	0, 1, 0	3	3
4	0	1, 2, 4	0, 0, 0	0	1
	1	2, 4, 7	0, 0, 0	3	1
	2	4, 7, 1	0, 0, 0	2	1
	3	7, 1, 2	0, 0, 0	1	1

Table 1. Neighbor lookup table for a tetrahedron in an even configuration.

Current Tetrahedron			Neighboring Tetrahedron		
Tet #	Face #	Vertex Indices	Changes to Hexahedral Node Indices	Tet #	Face #
0	0	7, 3, 5	1, 0, 0	3	2
	1	3, 5, 6	0, 0, 0	4	0
	2	5, 6, 7	0, 0, 1	1	0
	3	6, 7, 3	0, 1, 0	2	3
1	0	1, 5, 3	1, 0, 0	0	2
	1	5, 3, 0	0, 0, 0	4	3
	2	3, 0, 1	0, 0, -1	2	0
	3	0, 1, 5	0, -1, 0	1	3
2	0	2, 6, 0	-1, 0, 0	1	2
	1	6, 0, 3	0, 0, 0	4	2
	2	0, 3, 2	0, 0, -1	3	0
	3	3, 2, 6	0, 1, 0	0	3
3	0	4, 0, 6	-1, 0, 0	2	2
	1	0, 6, 5	0, 0, 0	4	1
	2	6, 5, 4	0, 0, 1	0	0
	3	5, 4, 0	0, -1, 0	3	3
4	0	3, 5, 6	0, 0, 0	0	1
	1	5, 6, 0	0, 0, 0	3	1
	2	6, 0, 3	0, 0, 0	2	1
	3	0, 3, 5	0, 0, 0	1	1

Table 2. Neighbor lookup table for a tetrahedron in an odd configuration.

To create a set of connected lines, the endpoints of each vortex-core line segment are checked from the set of ‘unconnected’ line segments. If there is a neighboring line segment, then the current endpoint is saved and the neighboring tetrahedral cell is next examined. This process is repeated until there are no more neighboring line segments. Since there are two endpoints for each line segment, the left and right children are kept track of during the traversal. A list of connected vortex-core lines is obtained after all the vortex-core line segments are traversed. Using this technique, the length of each contiguous vortex-core line is stored in memory. Figures 5a and 5b depict the connected vortex-core lines that span at least 30 tetrahedral cells and 100 tetrahedral cells, respectively. Compared to the vortex-core lines shown in Figs. 1 and 2, the connected vortex-core lines appear more distinct and less cluttered. Quantitative data can now be extracted from these contiguous vortex-core lines.

C. Cross-flow Velocity Profile

An automated technique for extracting the cross-flow velocity profile of the vortex-core line is now described. This technique is demonstrated by using a numerical simulation of a V22 Osprey isolated rotor in hover. The V22 isolated rotor consists of three rotor blades and a spinner hub. The cross-flow velocity component can be computed as follows:

- (1) Determine the intersections of the vortex-core lines with the $y=0$ constant plane. The vortex-core lines may intersect the $y=0$ plane at one or more intersection points. Figure 6a shows an oblique view from above the rotor of the vortex-core lines, the $y=0$ constant plane, and the intersections (represented by yellow boxes) of the vortex-core lines with the $y=0$ plane.

- (2) These intersection points are sorted in descending z order (assuming rotation about the z -axis) so that the vortex cores near the rotor blade are selected first. These intersections are also grouped according to the sign of their x coordinates. This information is used to assign a wake age (azimuthal angle) for the intersection points. Depending on how many azimuthal rotations are traversed by a vortex core, there may be none or several intersection points selected for velocity profile analysis.
- (3) After the corresponding intersection points have been selected for profiling, the w (cross-flow) component of the velocity along the x -direction is stored. Figure 6b shows a plot of the cross-flow velocity component for the vortex core located at the right most intersection point (wake age at 20°). The green vertical line shown in Fig. 6b represents the location of the vortex center. The red line and symbols represent the cross-flow velocity profile of the vortex.

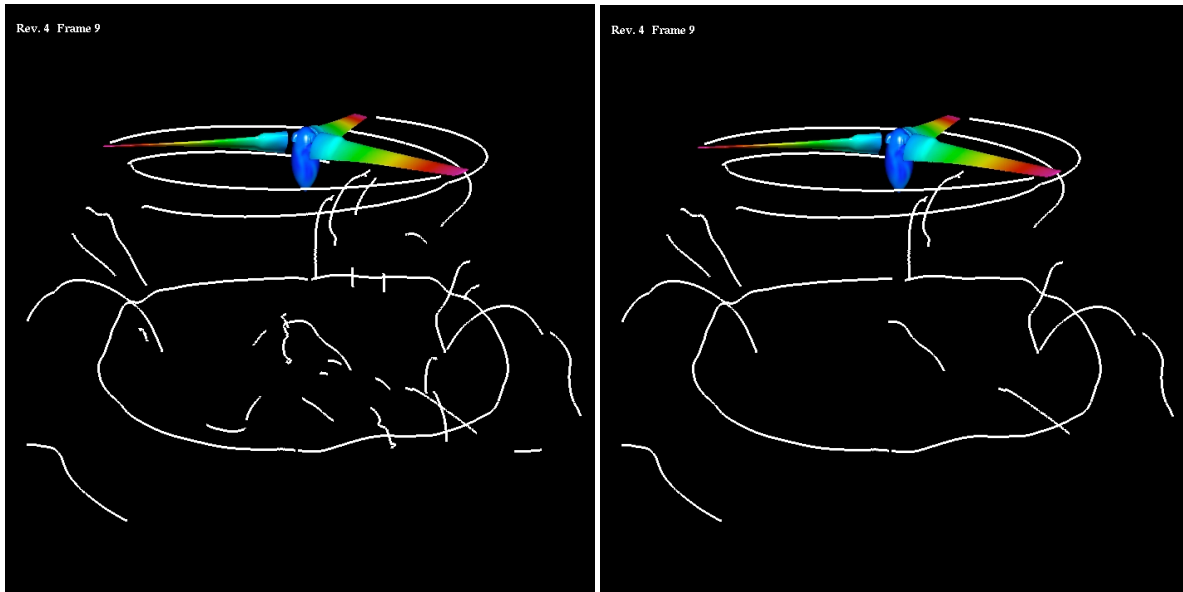


Figure 5. Vortex cores extracted using the face-based approach with the line connection algorithm. Left image depicts and vortex-core lines with (a) a minimum span of 30 tetrahedral cells and (b) a minimum span of 100 tetrahedral cells.

D. Vortex-Core Diameter

Once the cross-flow velocity components of a vortex are known, the vortex-core diameter can be determined. This is often defined as the distance between the local minimum and local maximum cross-flow velocity magnitude. Figure 7, an enlarged view of Fig. 6b, shows how the cross-flow velocity varies with distance from the vortex center. The blue “X” to the left of the green line is the x -position of the local minimum cross-flow velocity, and the blue “X” to the right of the green line is the local maximum cross-flow velocity. The vortex-core diameter is shown as the blue line connecting the two blue “X” marks.

This method of automatically computing the cross-flow velocity and the vortex-core diameter is applied at each yellow intersection point shown in Fig. 6a. Once these vortex diameters have been determined, they can be plotted to show how the vortex-core diameter varies with wake age. A common practice is to plot the normalized vortex-core diameter, where the actual vortex-core diameter is divided by C_{tip} (the chord length of the rotor blade tip), versus wake age.⁸

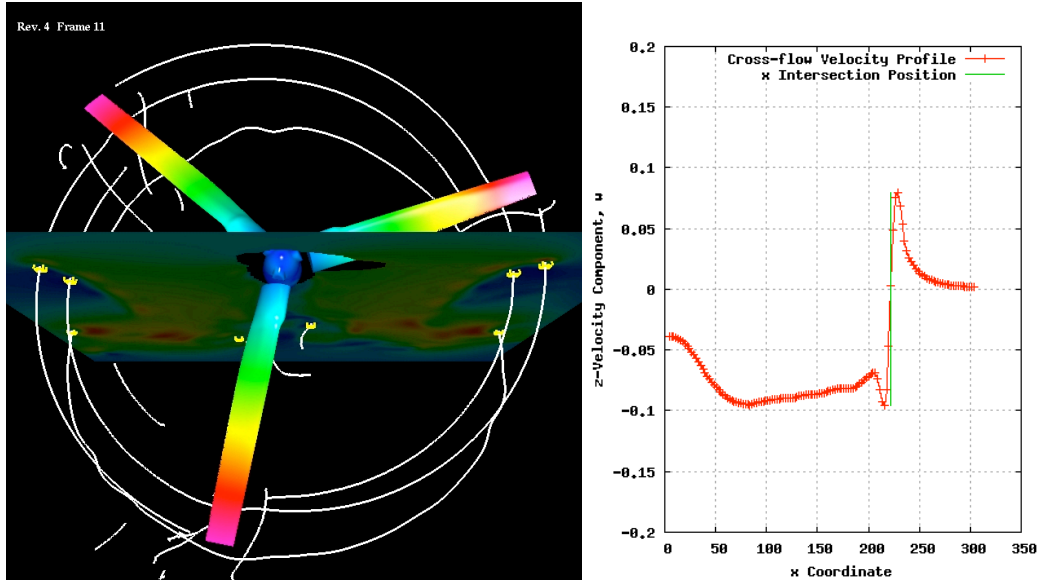


Figure 6. (a) Intersection points (yellow cubes) of the vortex-core lines with the $y=0$ plane. (b) The cross-flow velocity profile for a wake age of 20^0 (the intersection point is represented by the right most yellow cube). The green vertical line indicates the center of the vortex.

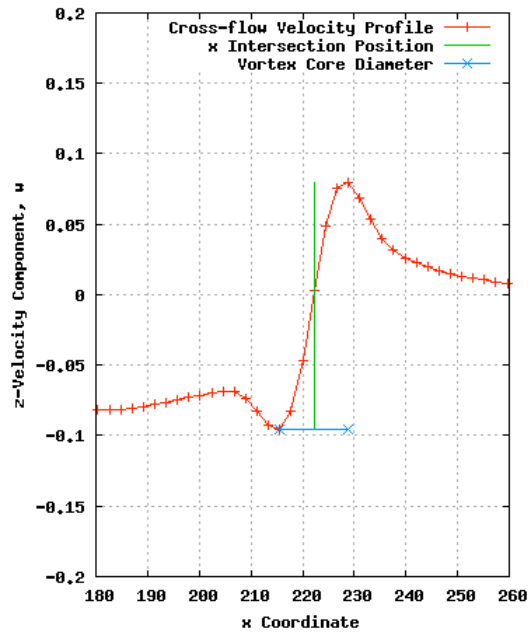


Figure 7. A close-up view of the cross-flow velocity profile for a wake age of 20^0 . The vertical green line represents the vortex center. The vortex-core diameter is denoted by the blue line connecting the two blue crosses.

III. Results

The Vortex-Core Profiling (VCP) algorithm described in the previous section is now applied to two Navier-Stokes grid resolution studies by Ahmad⁹ and Holst and Pulliam⁸ for an isolated V22 TRAM¹⁰ rotor with simplified hub in hover.

In the first case study, body-fitted, curvilinear coordinate, overset grids are attached to the three V22 rotor blades, which are imbedded in a uniformly spaced Cartesian background grid. This background grid captures the vortex wake generated by the spinning rotor blades. Successively courser Cartesian overset “brick” grids are added to this background grid system to rapidly expand the computational domain to a far-field.

Three different background grid resolutions are shown in Fig. 8. They correspond to a uniform spacing of 4.4, 2.2, and 1.1 inches, respectively. The middle resolution is 10% of the rotor-tip chord, a typical choice in CFD rotorcraft simulations, and approximately the size of the vortex-core diameter. The total number of grid points for the three grid systems are: 10 million, 26 million, and 143 million grid points, respectively.

Figure 9 shows vortex-core lines extracted using the non-contiguous cell-based approach and the new contiguous face-based approach on the coarsest background grid ($\Delta s=4.4$ inches). Notice that the rotor-blade tip vortices dissipate quickly due to the coarse background grid resolution. The same information is displayed in Fig. 10, where the finest background resolution is used ($\Delta s=1.1$ inches). Notice that with the finer grid resolution, the tip vortices persist for more than one full revolution (longer wake age). Furthermore, the connected face-based approach produces an image that is less cluttered and easier to follow vortical features than the non-contiguous cell-based approach.

Figure 11 shows the cross-flow velocity profiles extracted at several wake ages for different background grid resolutions. The minimum cross-flow velocity peak precedes the maximum cross-flow velocity peak on the right side of the $y=0$ plane (where $x>0$), see Fig. 11a. The situation is reversed on the other side of the $y=0$ plane (where $x<0$), see Fig. 11b. This occurs because the plots are made from the same viewpoint, looking towards the $y=0$ plane. It is also possible to display these plots such that the minimum cross-flow velocity peak always precedes the maximum peak. This would represent a viewpoint that traverses along the vortex line, so that the vortex always has a counterclockwise sense of rotation. Notice that as the grid resolution is decreased from $\Delta s=1.1$, to 2.2, and 4.4 inches, the number of cross-flow velocity profiles detected decreases from 6, to 4, and 1 respectively. This is consistent with the courser grids having greater vortex diffusion, resulting in shorter vortex-core lines. This diffusion also causes the cross-flow velocity peaks to diminish in magnitude with coarser grid resolution.

Figure 12 shows the normalized vortex-core diameter variation with wake age as extracted using the VCP algorithm for two background grid resolutions ($\Delta s=1.1$ and 2.2 inches). Two sets of experimental data are also shown in the figure. Notice that both CFD resolutions show the general trend of increasing vortex diameter with wake age. This trend reflects both the physical and numerical dissipative processes causing the vorticity in the vortex to diffuse. A surprising dip in vortex diameter size at a wake age of approximately 200° is indicated in both the finer CFD grid resolution and the experimental data.

The VCP algorithm was also applied to another grid resolution study.⁸ In this study, two types of vortex-wake grid refinement were used. The first approach used uniform grid spacing in the background grid, like the previous example. Figure 13 shows a view of the baseline grid system (no grid adaption) at the coordinate plane $z=0$ with the background grid spacing of $\Delta S_1=2.2$ inches (10% of the rotor blade tip-chord), which had 120 million grid points. Two additional uniform grid refinement were reported in Ref. 8, i.e., $\Delta S_2=\Delta S_1/2=1.1$ inches (164 million grid points) and $\Delta S_3=\Delta S_1/4=0.55$ inches (486 million grid points).

The second approach used in Ref. 8 was grid adaption. Various levels of refined Cartesian grids were imbedded into the background grid to better resolve the rotor blade vortices in a more efficient manner. For example, a three-level adaption means the background grid spacing is $\Delta S_1=2.2$ inches, and additional Cartesian block grids are imbedded into the background grid with grid spacings of $\Delta S_2=\Delta S_1/2$, $\Delta S_3=\Delta S_1/4$, and $\Delta S_4=\Delta S_1/8$ (448 million grid points in all). This finest level of grid adaption means that approximately 8 grid cells span a typical rotor blade vortex-core diameter. Note also that the adapted grid system (448 million grid points) had twice the resolution of the finest uniform grid system (486 million grid points) at a reduced computational cost, due to fewer grid points.

Figures 14 and 15 show the extracted cell-based non-contiguous vortex cores (left) and the new face-based contiguous vortex cores (right), for the uniform refinement case using the baseline background grid spacing ΔS_1 and ΔS_3 (a uniform grid refinement: $\Delta S_3=\Delta S_1/4$), respectively. As expected, the extracted vortex cores are longer (greater wake age) for the more finer refinement case than the non-refined baseline grid case. Once again, the new contiguous algorithm provides a more distinct, easier to follow image, with less clutter from short line segments.

Figure 16 shows the cross-flow velocity profiles at several wake ages for the uniform grid refinement cases. As before, finer grids produced higher cross-flow velocity peaks, smaller vortex-core diameters, and less diffused vorticity within the vortex.

Figure 17 shows a plot of the normalized vortex-core diameter versus wake age. The first three lines correspond to the automated VCP algorithm extraction of vortex-core size from the uniform grid refinement study. Notice that as the grid is refined in the vortex wake region, the vortex-core diameter growth with wake age is diminished, and in better agreement with the experimental data. The last two lines in Fig. 17 correspond to the three-level grid adaption solution. This represents the highest level of grid refinement reported in Ref. 8, and is in close agreement with the measured data. The vortex-core diameters were computed using both the automated VCP algorithm and manually. Notice that both results are virtually identical. The automated approach greatly reduces the user workload.

IV. Summary

A Vortex Core Profiling (VCP) algorithm has been presented to automatically identify and visually display continuous vortex-core lines associated with rotor blade tips under hover conditions. This is an improvement over a previous vortex core detection algorithm that extracts non-contiguous vortex-core line segments. The resulting continuous vortex-core lines are easier to view, and the image is less cluttered with short line segments. The VCP algorithm also extracts vortex cross-flow velocity profiles and vortex-core diameter variation with wake age. This provides a tool to routinely compare CFD results with measured experimental data.

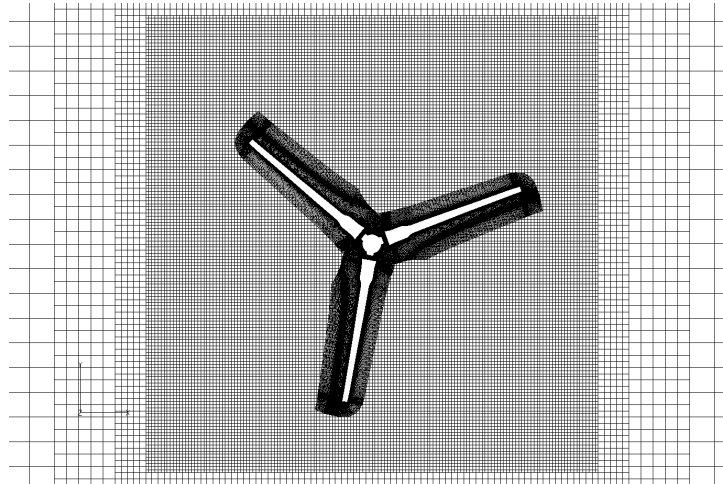
The automated VCP algorithm was demonstrated on several CFD simulations of an isolated V22 Tram rotor in hover with a simplified hub. The vortex-core profiling tool provided useful insight into grid resolution studies reported in the literature. The vortex-core diameter growth with wake age computed by the automated method was virtually identical to manually obtained results, and compared well with experimental data. This approach currently works only for structured overset grids, but it could be extended to unstructured grid simulations.

Acknowledgments

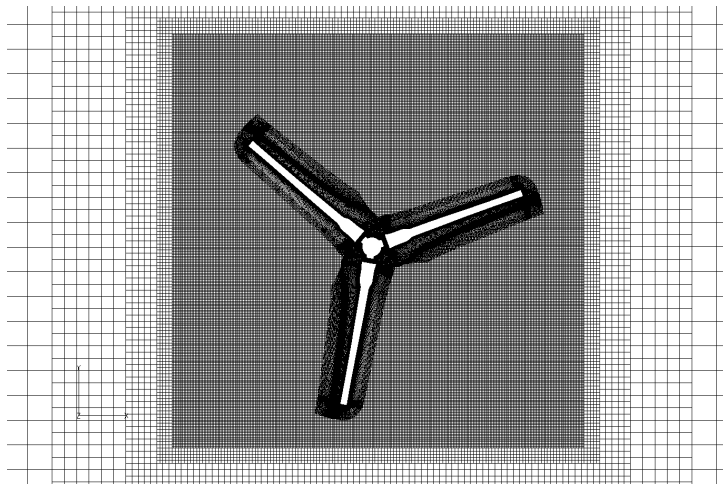
The authors thank the Subsonic Rotary Wing (SRW) Project of NASA's Fundamental Aeronautics Program for supporting this work, and to Terry Holst, Tom Pulliam, and Jasim Ahmad for constructive discussions during the development of the automated approach presented in this paper. The first author thanks Terry Holst for several helpful suggestions during the conception of this research.

References

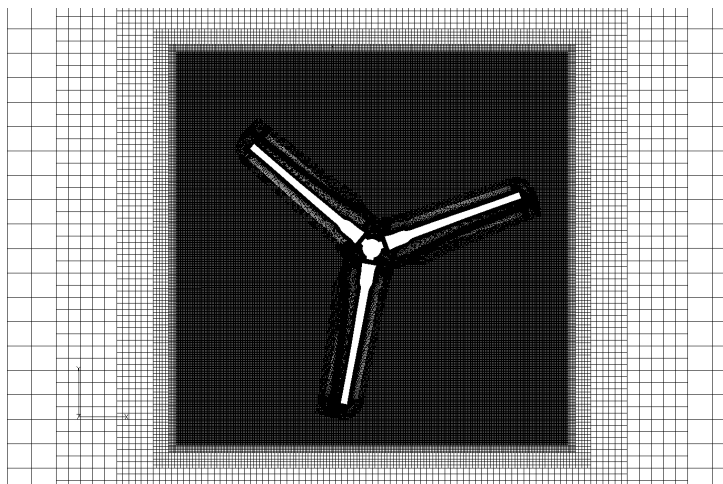
- ¹Yamauchi, G. and Young, L., "A Status of NASA Rotorcraft Research," NASA/TP-2009-215369, September 2009, pp. 149-196.
- ²Jiang, M., Machiraju, R., and Thompson, D., "Detection and Visualization of Vortices," in Visualization Handbook, pp. 295-309, C. Hansen and C. Johnson, eds, Academic Press, 2005.
- ³Jeong, J. and Hussain, F., "On the Identification of a Vortex," *J. Fluid Mechanics*, 285:69-94, 1995.
- ⁴Hunt, J.C., Wray, A., and Moin, P., "Eddies, Streams, and Convergence Zones in Turbulent Flows," Report CTR-S88, Center for Turbulence Research, 1988.
- ⁵Sujudi, D. and Haimes, R., "Identification of Swirling Flow in 3-D Vector Fields," 12th AIAA Computational Fluid Dynamics Conference, AIAA Paper 95-1715, 1995.
- ⁶Haimes, R. and Kenwright, D., "On Velocity Gradient Tensor and Fluid Feature Extraction", 14th AIAA Computational Fluid Dynamics Conference, AIAA Paper 99-3288, June 1999.
- ⁷Kenwright, D. and Lane, D., "Interactive Time-Dependent Particle Tracing using Tetrahedral Decomposition," *IEEE Trans. Visualization and Computer Graphics*, Vol. 2, No. 2, pp. 120-129, 1996.
- ⁸Holst, T. and Pulliam, T., "Overset Solution Adaptive Grid Approach Applied to Hovering Rotorcraft Flows", 27th AIAA Applied Aerodynamics Conference, AIAA Paper 09-3519, June 2009.
- ⁹Ahmad, J., Private communication, Jasim.U.Ahmad@nasa.gov, NASA Ames Research Center, Moffett Field, CA, 94035.
- ¹⁰Young, L. "Tilt Rotor Aeroacoustic Model (TRAM): A New Rotorcraft Research Facility," Heli Japan 98: AHS International Meeting on Advanced Rotorcraft Technology and Disaster Relief, Gifu, Japan, April 1998.
- ¹¹McAlister, K. and Heineck, J., "Measurements of the Early Development of Trailing Vorticity from a Rotor," NASA/TP-2002-211848, AFDD/TR-02-A001, May 2002.
- ¹²Ramasamy, M., Johnson, B. and Leishman, J.G., "Understanding the Aerodynamic Efficiency of a Hovering Micro-Rotor," *J. of the American Helicopter Society*, Vol. 53, No. 4, Oct. 2008, pp. 412-428.



a) Background grid spacing, $\Delta s=4.4$ inches



b) Background grid spacing, $\Delta s=2.2$ inches



c) Background grid spacing, $\Delta s=1.1$ inches

Figure 8. Top view of three uniformly spaced background grid resolutions.

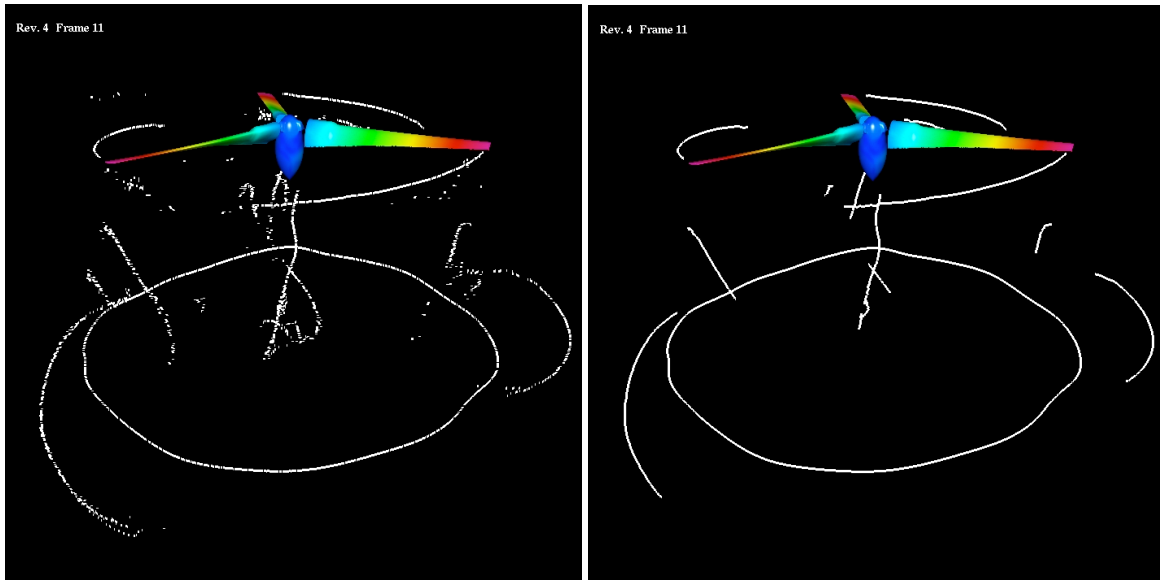


Figure 9. Vortex cores extracted using the non-contiguous cell-based approach (left), and the contiguous face-based approach (right). The background grid resolution is $\Delta s=4.4$ inches.

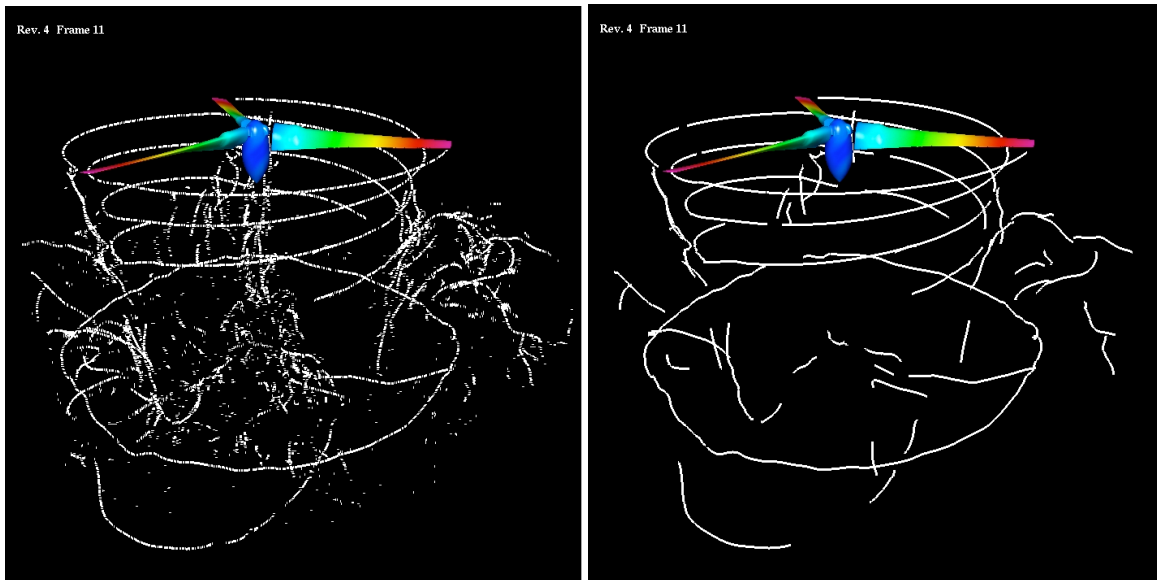
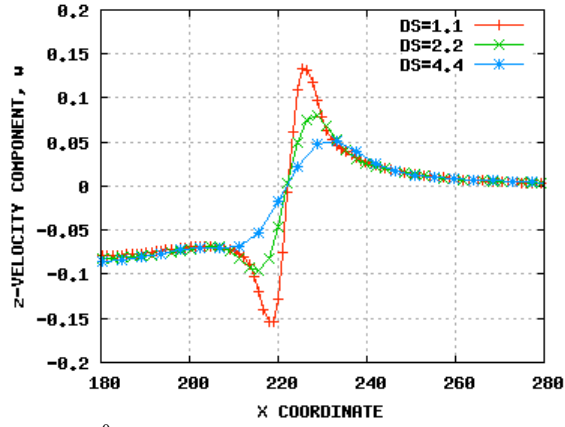
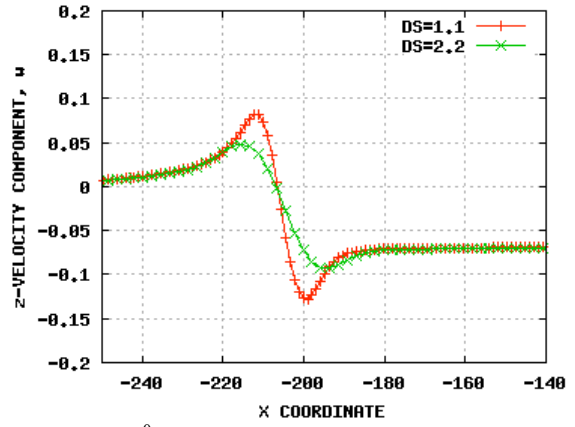


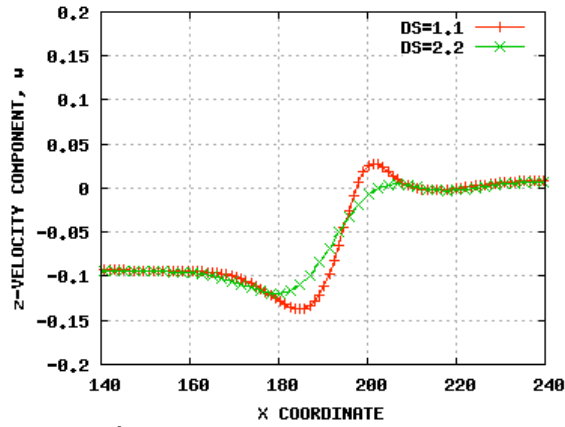
Figure 10. Vortex cores extracted using the non-contiguous cell-based approach (left), and the contiguous face-based approach (right). The background grid resolution is $\Delta s=1.1$ inches.



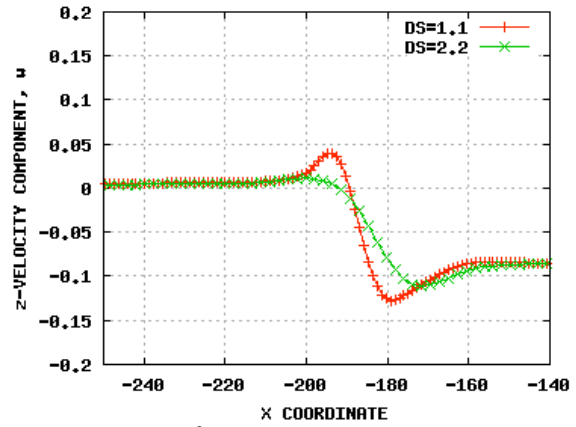
a) $\psi = 20^\circ$



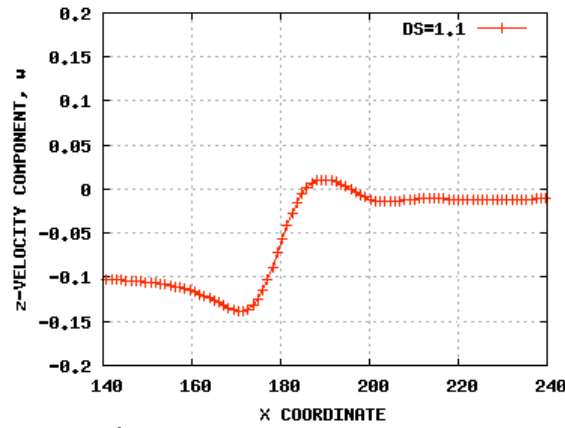
b) $\psi = 80^\circ$



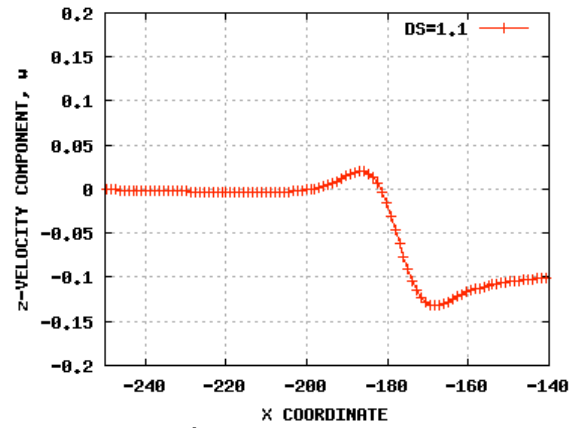
c) $\psi = 140^\circ$



d) $\psi = 200^\circ$



e) $\psi = 260^\circ$



f) $\psi = 320^\circ$

Figure 11. Cross-flow velocity profiles with different background grid resolutions and wake ages (ψ).

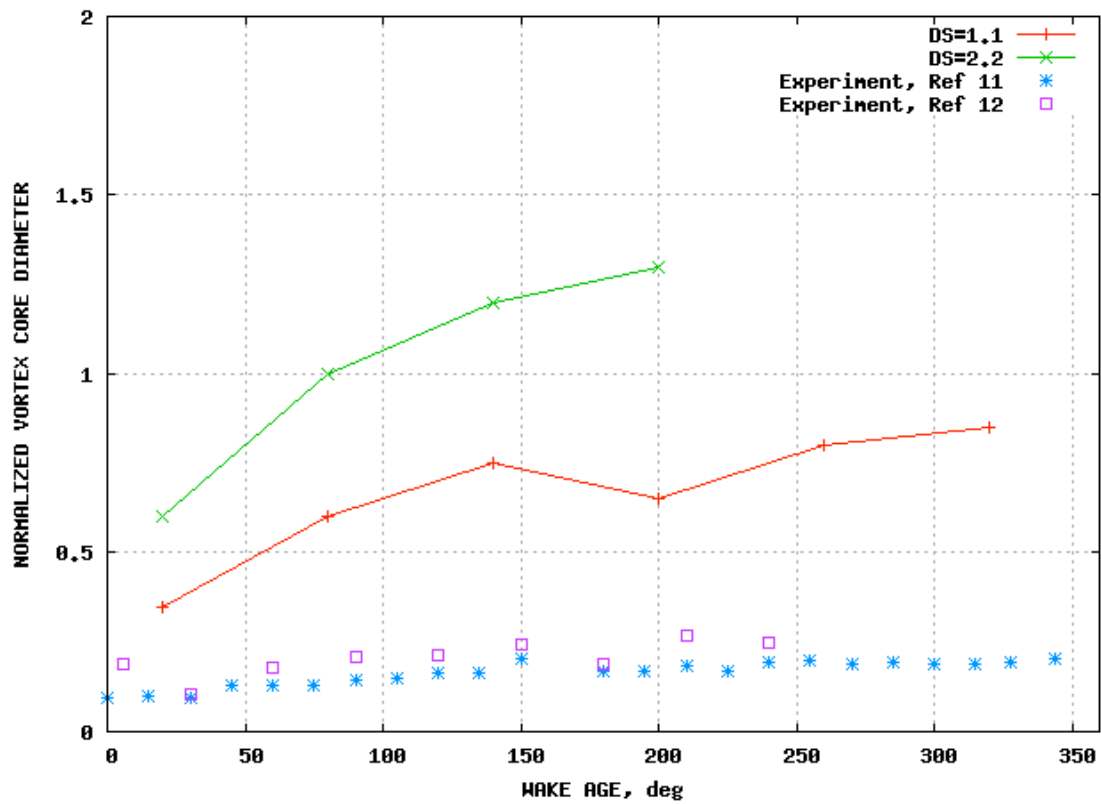


Figure 12. Normalized vortex-core diameter growth with wake age for two CFD background grid resolutions and two sets of experimental data.

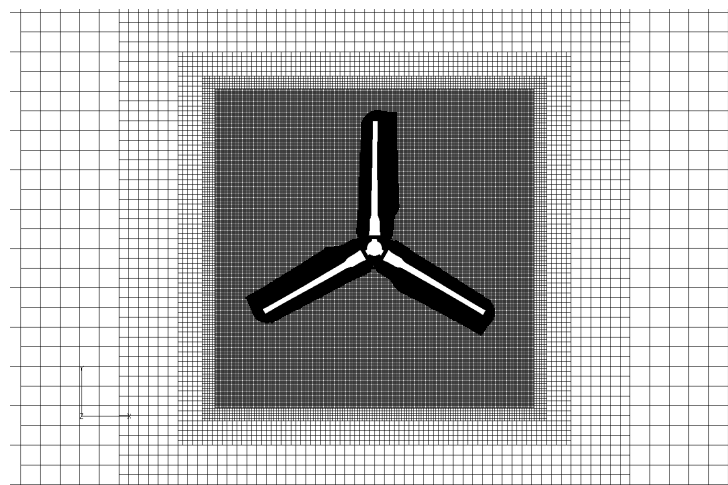


Figure 13. Top view of a uniformly spaced baseline grid (no grid adaptation). $\Delta S_1 = 2.2$ inches (10% of the rotor blade tip).

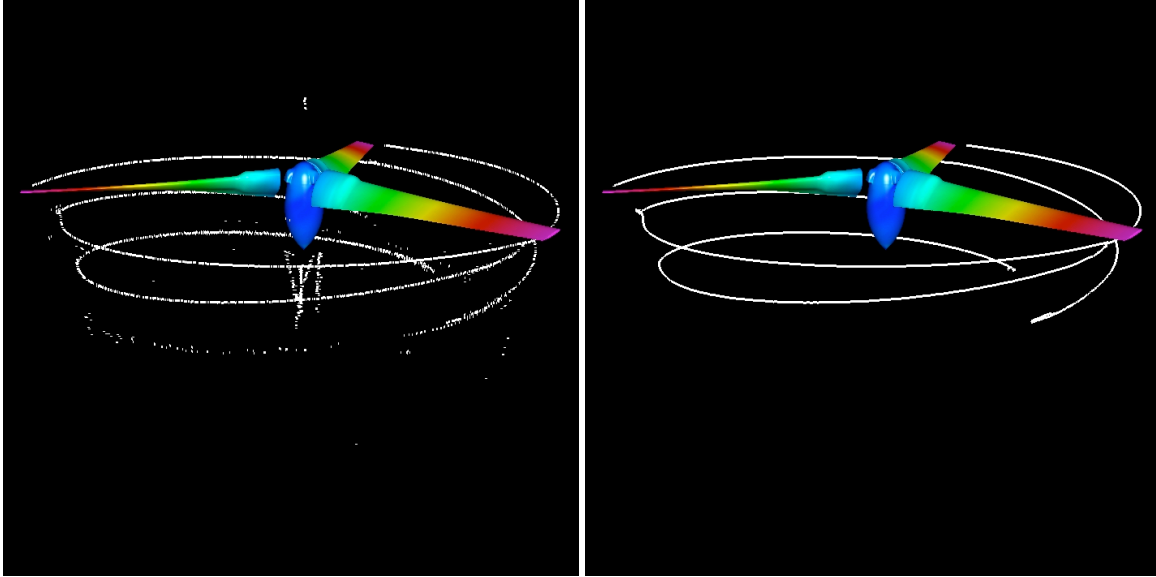


Figure 14. Vortex cores extracted using the non-contiguous cell-based approach (left), and the contiguous face-based approach (right). The background grid resolution is $\Delta S_1=2.2$ inches.

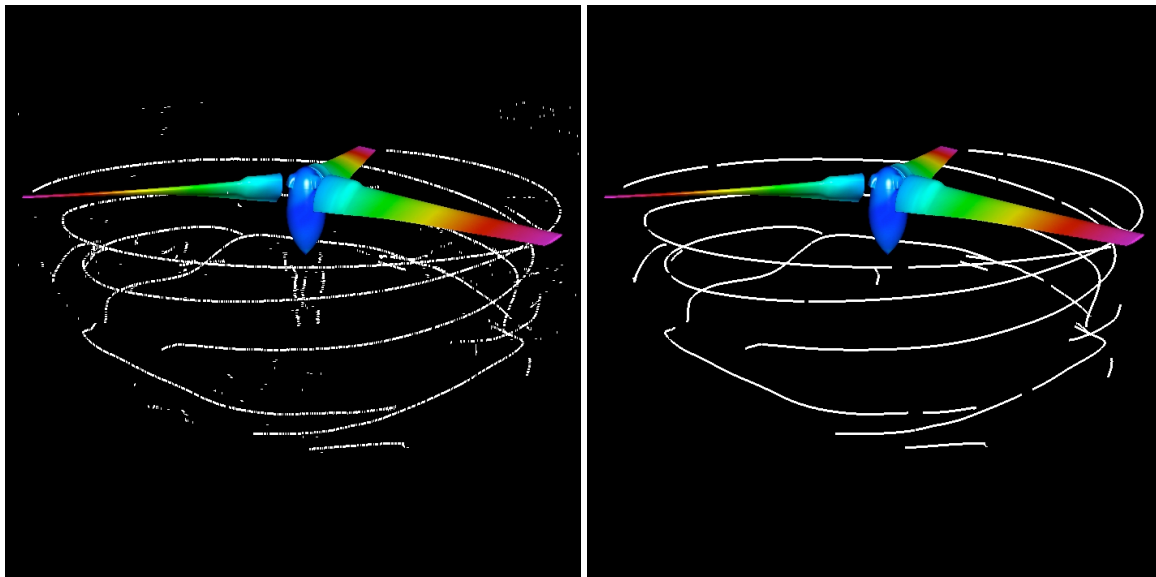
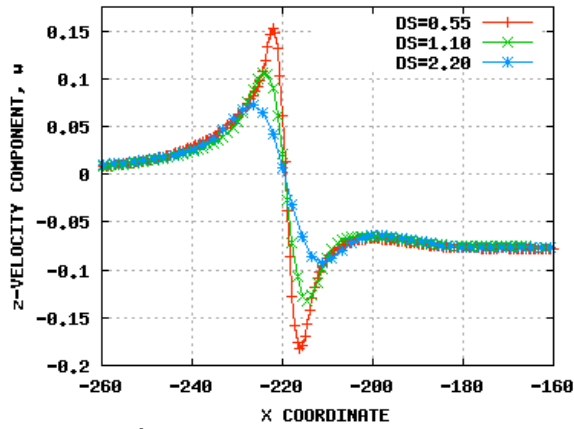
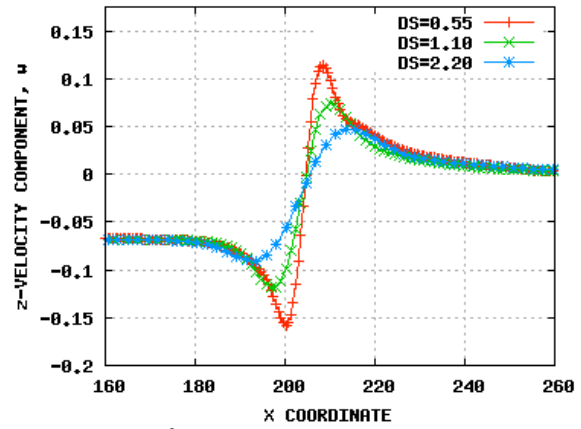


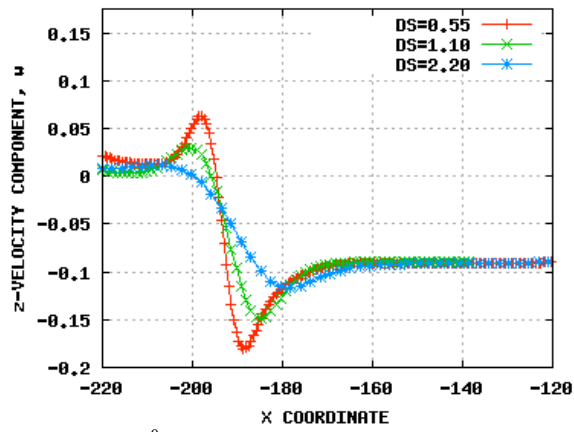
Figure 15. Vortex cores extracted using the non-contiguous cell-based approach (left), and the contiguous face-based approach (right). The grid system is based on a uniform grid refinement of the baseline grid: $\Delta S_3=\Delta S_1/4=0.55$ inches.



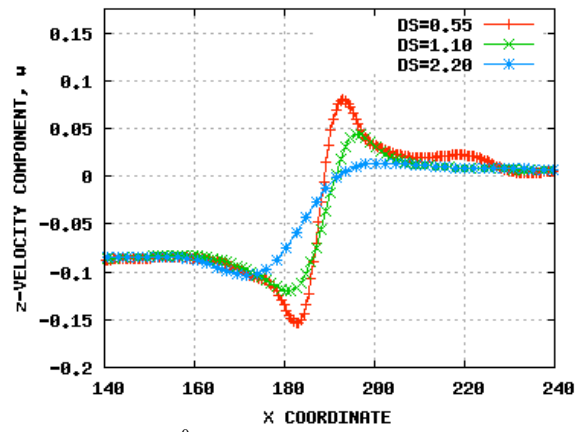
a) $\psi = 30^\circ$



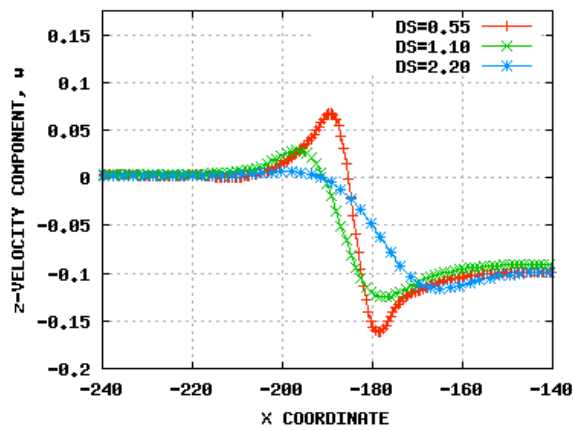
b) $\psi = 90^\circ$



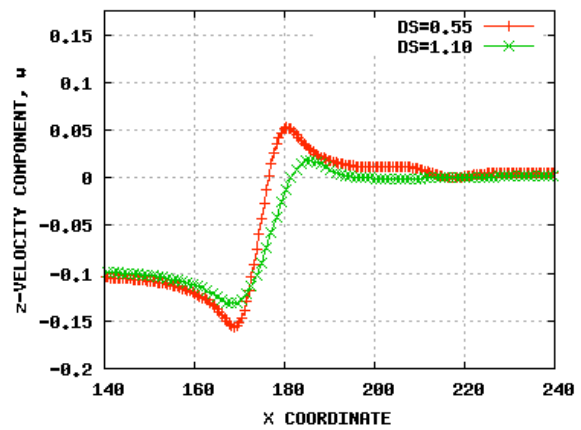
c) $\psi = 150^\circ$



d) $\psi = 210^\circ$



e) $\psi = 270^\circ$



f) $\psi = 330^\circ$

Figure 16. Cross-flow velocity profiles with different background grid resolutions and wake ages (ψ).

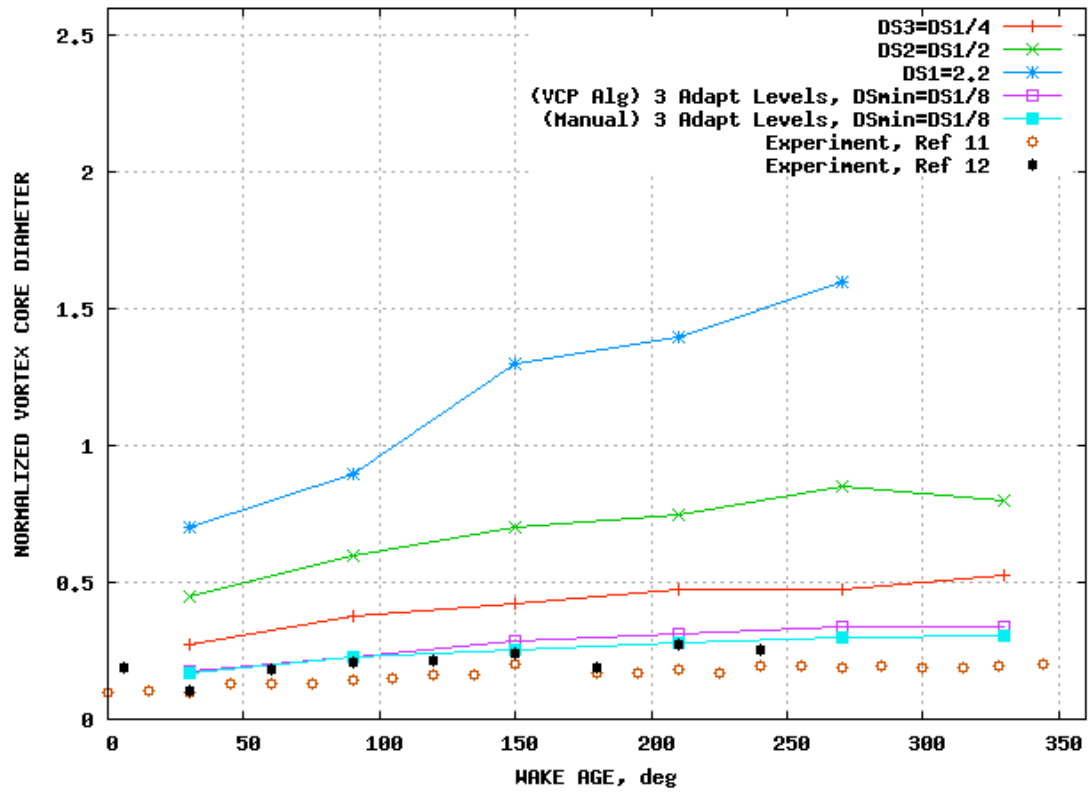


Figure 17. Normalized vortex-core diameter growth with wake age using uniform background grid refinement and solution-adapted results with imbedded Cartesian grids ($\Delta S_2 = \Delta S_1/2$, $\Delta S_3 = \Delta S_1/4$, and $\Delta S_4 = \Delta S_1/8$).