AIAA 99-0155

AN ADAPTIVE, FAST, PARALLEL, VORTEX METHOD FOR NUMERICAL SIMULATIONS OF TURBULENT SEPARATED FLOWS

Athanassios A. Dimas *
Krispin Technologies, Inc.
Rockville, MD

Peter S. Bernard †
University of Maryland
College Park, MD

Jacob Krispin‡
Krispin Technologies, Inc.
Rockville, MD

Abstract

An adaptive, fast, parallel, vortex method for the numerical simulation of incompressible turbulent flow is presented. The algorithm is based on: (1) a vortex creation model, which imitates the physical vortex self-replication process in the vicinity of solid boundaries, (2) the use of convecting, stretching and diffusing vortex filaments, (3) a hairpin removal and reconnection mechanism to limit the number and scale of vortical structures to those that are dynamically essential, and (4) a massively parallel and adaptive implementation of the Fast Multipole Method to greatly reduce the velocity computation time compared to the direct calculation by the Biot-Savart law. Velocity computation timings on distributed memory machines using Message Passing Library and Message Passing Interface show a net gain in numerical efficiency for calculations with more than approximately 4,000 vortex elements, which grows to a two-order reduction for computations with 10^6 vortex elements, and full scalability of the parallel algorithm. Results (vortex filament structure and surface pressure distribution) are presented from the numerical simulation of the turbulent separated flow past a 6:1 prolate spheroid in uniform stream at 0° and 20° angle of attack.

*Senior Research Scientist, AIAA Member
†Professor, Department of Mechanical Engineering
‡President, AIAA Member

Introduction

Vortex methods, wherein a flow field is represented by a collection of convecting and diffusing vortex elements, exhibit several advantages over traditional grid-based methods for the numerical simulation of turbulent flow.[1,2] In particular, they have negligible if any numerical diffusion so that high Reynolds number turbulence effects can be represented with high numerical accuracy and fidelity, and they are grid-free so that they can be easily applied to engineering flows in complex geometries. Moreover, they are naturally adaptive since the computational elements occupy only the relatively small support of the flow field where the vorticity is significant. Finally, vortex methods offer a direct means for modeling some of the most important dynamical processes of turbulent flow; the energy cascade driven by vortex stretching and the quasi-streamwise vortex regeneration process near boundaries.[3]

The direct implementation of vortex methods, however, presents two problems that in the past have limited their appeal in practical engineering applications. First, vortex methods are very honest in tracking the dynamics of individual vortical structures as they stretch and fold, but, unless special steps are taken to confine this tendency to just those scales that are critical to the prediction of the flow field, the number of vortices can grow to impractical levels. Second, the computational time associated with the nominally $O(N_v^2)$ operations implicit in the Biot-Savart law evaluation of velocities due to N_v vortex elements, an essential aspect of their for-

mulation and a source of their unique advantages, can grow beyond reasonable levels for engineering use.

To address the first problem, a unique vortex method has been devised[2] which limits the number and scale of vortical structures to just those which are dynamically essential based upon recent advances in understanding turbulent boundary layer physics[3,4,5] and the statistical mechanics of vortex systems.[6] In particular, it mimics the processes of near-wall vortex self-replication, vortex reconnection and subgrid renormalization via hairpin removal,[7] to prevent the formation of small scale disturbances. In effect, the approach has the character of a large eddy simulation (LES) which is both more efficient than grid-based LES while providing a more realistic accounting of the subgrid physics than traditional diffusive models.

To address the second problem, fast computation of the velocity field is attained by using the Fast Multipole Method (FMM).[8,9,10] A number of recent studies have used the FMM to do calculations with very large numbers of vortex elements. In particular, two-dimensional calculations of viscous flow using more than a million vortices have been reported.[11] The next level of improvement is the parallelization of the FMM, and recent studies have presented parallel, non-adaptive, two-dimensional FMM algorithms.[12,13] In our implementation, a parallel, adaptive, three-dimensional FMM algorithm, which was developed on distributed memory machines using both Message Passing Interface (MPI) and Message Passing Library (MPL). In three dimensions, FMM combines the velocity field induced by collections of nearby vortices into a single local expansion in spherical harmonics. By shifting the center of the expansion to the neighborhood of other collections of vortices, an efficient velocity field calculation can be achieved. With use of an oct-tree structure to efficiently group vortices into near and well separated boxes, an $O(N_v)$ algorithm results, which is parallelized by distributing these boxes amongst the processors.

Results are presented from timings of the FMM velocity computation and simulations of the turbulent separated flow past a 6:1 prolate spheroid at 0° and 20° angle of attack in a uniform stream.

Vortex Method Formulation

The evolution of the vorticity field, $\mathbf{\Omega}(\mathbf{x}, t)$, in three-dimensional flow is governed by the transport equation

$$\frac{\partial \mathbf{\Omega}}{\partial t} + (\nabla \mathbf{\Omega})\mathbf{u} = (\nabla \mathbf{u})\mathbf{\Omega} + \frac{1}{R}\nabla^2 \mathbf{\Omega} \qquad (1)$$

where $\mathbf{u}$ is the velocity field and R is the Reynolds number. In a vortex method, numerical solutions to (1) are obtained in the form of collections of N_v vortex elements covering the support of the vorticity field.[1] Depending on the particular details of the method, these may be in the form of smoothed point vortices (i.e., blobs), filaments, sheets or other configurations. In the vortex method presented in this paper, the flow field is represented by a collection of thin rectangular vortex sheets, which form layers that cover any solid surfaces present, and convecting, stretching and diffusing vortex filaments, which are present in the rest of the flow domain. Layers of smoothed vortex sheets are used close to solid walls due to their efficient performance in capturing wall-normal viscous diffusion.[2,14] Vortex filaments are natural elements to use in modeling the vortical structures – coherent or otherwise – which play a major role in the dynamics of turbulent flow.[3,15] In view of the linear dependence of (1) on vorticity, there is little difficulty in accommodating the coexistence of sheets and tubes near solid walls.

The vorticity equation for the ith element can be written in Lagrangian form as

$$\frac{d\mathbf{\Omega}_i}{dt} = (\nabla \mathbf{u})\mathbf{\Omega}_i + \frac{1}{R}\nabla^2 \mathbf{\Omega}_i \qquad (2)$$

where d/dt is the material derivative operator. A variety of fractional step methods can be employed in solving (2) in conjunction with a Runge-Kutta discretization of the kinematic equation

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}(\mathbf{x}_i(t), t) \qquad (3)$$

governing the position $\mathbf{x}_i(t)$ of the ith vortex element. In the case of sheets, it is convenient to regrid after every time step, thus allowing the vortex stretching and diffusion terms in (2) to be represented through second order finite differences.[2] The filament part of the calculation is adapted directly from standard grid-free schemes using collections of tubes.[7] Each filament (which may be composed of a number of straight-line segments) is assigned a constant circulation along its length. The end points, $\mathbf{x}_i(t)$, of each segment are moved in time according to the kinematic equation (3), which accounts for both the stretching and convection terms.

Following the approach of Chorin,[6,7] folded vortex segments forming hairpins are removed when their interior angle is less than a critical angle. Similarly, vortex tubes of equal and opposite circulation

are reconnected if they come into close proximity. These steps are tantamount to a physically based subgrid renormalization limiting resolution to just the larger scales. The importance of hairpin removal to the success of the method cannot be overstated: without this renormalization, 3D vortex methods tend to meticulously track the cascade of energy to small dissipative scales until the sheer number of vortical elements overwhelms the computation.[7,11] Depending on the application and available computer time, the criteria in the hairpin removal can be adjusted to allow more or less of the small scales to be resolved.

Vorticity diffusion is modeled by two different vortex destruction models.[2] In the first, vortices are eliminated if they become overly stretched, i.e., are formed from a large number of filament segments. Such older structures tend to be chaotic and their removal paves the way for relatively young, more coherently organized vortices to dominate the flow. A second scheme is to allow vortices which are not being sufficiently stretched to lose an appropriate amount of circulation at each time step, specifically, that amount which would diffuse beyond their core radius. Vortices whose circulation drops below a threshold are then eliminated. Calculations performed to date suggest that both of these approaches are viable.

Providing for each new generation of vortex tubes as they are produced in a turbulent flow is also a critical aspect of the numerical scheme. The algorithm must at the same time be both sensitive to the physical process by which new structures appear, yet not so unconstrained as to allow for the formation of impossibly large numbers of new vortices. A successful means of accommodating these conditions has been developed,[2] where sheets ejecting past the end of the sheet domain are reformulated as tubes if their vorticity is above a threshold. This models the process by which parent quasi-streamwise vortices provoke the creation of counter-rotating offspring through ejection and reorientation of vorticity produced at the boundary in fulfillment of the no-slip condition.

The velocity field is evaluated as a sum of contributions from individual vortical elements using the Biot-Savart law.[1] For the ith tube segment, the smoothed velocity field induced at a point $\mathbf{x}$ is computed from

$$-\frac{\Gamma_i}{4\pi}\,\frac{\mathbf{r}_i \times \mathbf{s}_i}{\mid \mathbf{r}_i \mid^3}\,\phi(r/\sigma) \qquad (4)$$

where Γ_i is the circulation, $\mathbf{r}_i = \mathbf{x}_i - \mathbf{x}$, $r = \mid \mathbf{r} \mid$, $\phi(r) = 1 - \left(1 - \frac{3}{2}r^3\right) e^{-r^3}$ is a second order smoothing function, $\mathbf{s}_i$ is the axial vector along the segment and σ is a scaling parameter. In the case of sheets, the velocity field is determined from a smoothed Biot-Savart integral.[2] Then, the velocity field due to a collection of sheets and filaments is a sum of the individual contributions. Following standard practice, the non-penetration boundary condition is enforced by adding a grid-free potential flow solution belonging to a surface source distribution. Boundary values of the tangential vorticity components are determined by evaluation of finite difference approximations to their definitions in terms of velocity derivatives, and using the no-slip condition.

Adaptive Parallel FMM

The FMM[8,9,10] is a fast summation scheme for calculating velocities induced on N_f field points by N_v vortices. The direct implementation of the Biot-Savart law results into an operation count of $O(N_v N_f)$. In most implementations of vortex methods, including ours, N_f is of the same order with N_v, hence the operations in the direct velocity computation are of $O(N_v^2)$. The serial FMM implementation results into an $O(N_v)$ algorithm, while further reduction of the computational cost is achieved by the efficient parallelization of the FMM algorithm.

The FMM is based on combining the velocity fields induced by collections of nearby vortices into a single local expansion in spherical harmonics. By shifting the center of the expansion to the neighborhood of other collections of vortices, a fast velocity field calculation can be effected. To facilitate the procedure of grouping vortices and field points, the flow domain is partitioned into cubic boxes. The number and size of these boxes, and, consequently, the number of vortices per box are parameters that control the efficiency of the FMM. In this paper, an adaptive FMM is employed, where the domain is partitioned into boxes of equal size but empty boxes are discarded. The partitioning of the flow domain into source boxes, S, and field boxes, F, of equal size is achieved by the following procedure:

- Level 1: The entire flow domain is enclosed inside a cubic box
- Level 2: The level 1 box is divided into 8 boxes (2 boxes per direction, $2^3 = 8$) of equal size
- Level 3: Each non-empty box of the previous level is divided into 8 boxes of equal size
- .....

- Level n: Each non-empty box of the previous level is divided into 8 boxes of equal size

The FMM reduces the complexity of the $O(N_v^2)$ algorithm by using multipole source expansions to compute the influences (field expansions) of the vortices on the field points, as well as the property that these expansions can be translated. In particular, multipole source expansions are first used to represent the influence of vortices within each source box by a truncated series of spherical harmonics. This multipole source expansion can be thought of as the equivalent of a single vortex at the center of S whose far field influence is approximately the same as the influence of each individual vortex in S. A further reduction in computational complexity is achieved by not applying the source expansion to each individual field point in F, but instead to the center of F. This is achieved by shifting the effect of source boxes onto field boxes via truncated expansions in spherical harmonics. Finally, the resulting field expansion centered in F can be translated to all the field points in F, using another truncated series, thus yielding the velocity at each field point. The truncation order, N, is the same for all these series, and it is the parameter that controls the accuracy of the FMM in evaluating the velocity field in comparison to the direct computation.

The above FMM procedure is used to evaluate the influence of vortices that are well-separated from the field points. For each field point, the vortices that are well-separated are defined as the vortices that do not belong to the same box as the field point or any of the surrounding 26 boxes that have a common face/edge/vertex. A two-dimensional projection of a typical box partition is shown in figure 1. The 8 dark gray boxes surrounding the black field-point box contain all the near-field vortices. Vortices in these boxes are not well-separated from the field points in the black box and their influence is evaluated using the direct computation. The rest of the boxes (light gray and white) are well-separated from the field-point box. The influences of the light gray boxes are evaluated at this level, while the influences of the white boxes are evaluated from the influences of previous levels.

The parallel implementation of this procedure is based on the following algorithm:

- Distribute non-empty boxes and all necessary data (e.g., coordinates of vortices and field-points within each box) amongst the N_p processors.
- On each processor; compute multipole source expansions in boxes S.
- Perform total exchange of multipole source expansion coefficients between processors.
- On each processor; do
  - Compute multipole field expansions centered in boxes F.
  - Translate field expansions centered in box F to all field points in F.

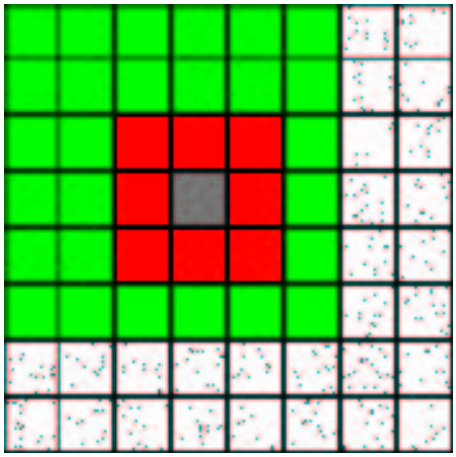

Figure 1: Typical box partition for the FMM implementation.

FMM Performance Results

The vortex field used to obtain the FMM velocity computation timings is an instantaneous realization obtained during the simulation of a turbulent channel flow with periodic boundary conditions along the streawise and spanwise directions.[2] The complete set consists of a basic set of 14,441 vortices and its surrounding 80 periodic images for a total of about 1,170,000 vortices. In these tests, the field points are identical to the vortex locations, therefore, $N_f = N_v$. The serial computations were performed on a Pentium Pro 200 Linux workstation. The parallel computations were performed on the Arctic machine of the SGI/Cray Corporation, where we had access to 64 Origin-2000 nodes using MPI, and the Quad machine of the Argonne National Laboratory, where we had access to 64 IBM-SP1 nodes using MPI and MPL.

For the serial FMM, detailed benchmark comparisons have been presented of CPU run times and truncation errors between the FMM and the direct, $O(N_v N_f)$, computation of the velocity field induced

by N_v vortices on N_f field points.[16] The truncation error, ϵ_u, in the velocity computation is defined as:

$$\epsilon_u = \left[\frac{\sum_{i=1}^{N_v} \left(\mid \mathbf{u} \mid_i - \mid \mathbf{u_e} \mid_i \right)^2}{\sum_{i=1}^{N_v} \left(\mid \mathbf{u_e} \mid_i \right)^2} \right]^{\frac{1}{2}} \tag{5}$$

where $\mid \mathbf{u} \mid$ is the magnitude of the velocity vector evaluated using FMM, and $\mid \mathbf{u_e} \mid$ is the magnitude of the exact velocity vector evaluated by the direct computation.

Several combinations of number of vortices, N_v, number of boxes, N_b, and truncation order of FMM expansions, N, were used. Typical results are presented in figure 2 and table 1. In figure 2, CPU run times are shown for the direct and the FMM computations; the latter for $N = 6$ and three different configurations with 64, 512 and 4096 boxes. The CPU run times for the FMM computations consist of the far-field contribution, which is dominated by an $O(N_b^2 N^4)$ operation, and the near-field direct computation, which is an $O(N_v^2/N_b)$ operation. It is shown that in the FMM computations, for low N_v values, the CPU run time is dominated by the far-field computation which is independent of N_v, while for high N_v values, the CPU run time is dominated by the $O(N_v^2)$ character of the near-field computation. Also, for low N_v values the FMM computation is more expensive than the direct computation but there is always a critical number of vortices beyond which the FMM computation becomes faster. The higher the number of boxes, N_b, or the higher the truncation order, N, the higher the critical number of vortices. As the number of boxes, though, increases, the FMM computation becomes even faster since the near-field computation is limited to a smaller number of vortices. For 10^6 vortices the FMM with $N = 6$ achieves a two-order of magnitude reduction in the computation time in comparison to the direct computation. For the number of vortices, N_v, and number of boxes, N_b, considered in this paper, the FMM truncation error depends strongly on the truncation order, N, and is more or less independent of N_v and N_b. In table 1 the typical order of magnitude of the FMM truncation errors are shown.

The algorithm of the parallel FMM takes advantage of the operations count dependencies on N_v and N_b to produce an efficient parallelization. Here, the time-consuming operation is the source box to field box computation. The most efficient way to perform this operation in parallel is to equally distribute non-empty boxes amongst the processors. The major advantage of this scheme is the reduced amount of information that has to be communicated between processors given the fact that, for a typical FMM computation, the number of boxes is, in general, about two orders of magnitude smaller than the number of vortices. The influence of the near-field vortices, on the other hand, is parallelized by equally distributing the field points amongst the processors according to the box to which they belong. For the parallel FMM velocity computation, a typical plot of CPU run time as a function of the number of processors, N_p, up to 64, for the configuration with 512 boxes and $N = 6$ is shown in figure 3. In this case, the overall computational time is dominated by the far-field FMM computation, while the cost of the near-field direct computation is negligible. The far-field computation is fully scalable for all three cases presented. The near-field computation is not fully scalable because its computation time is so small that it is overwhelmed by the overhead communication time. Similar behavior is observed for several other cases studied, and the conclusion is that by requiring an average vortex density of about 100 vortices per box, the velocity computational time is dominated by the FMM far-field computation and it is fully scalable.

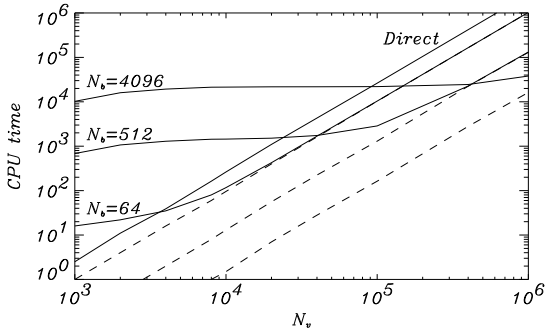

Figure 2: CPU run times in seconds as a function of number of vortices. The solid lines correspond to the $O(N_v^2)$ direct and the FMM computations; the latter for three different number of boxes, N_b. The dashed lines correspond to the near-field CPU run time of the corresponding FMM computations.

	$N = 4$	$N = 6$	$N = 8$
ϵ_u	10^{-3}	10^{-4}	10^{-5}

Table 1: FMM truncation error order of magnitude.

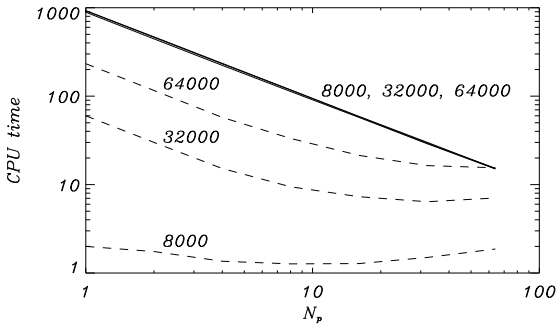

Figure 3: CPU run times in seconds as a function of number of processors for three configurations (N_v = 8000, 32000, 64000) with $N_b = 512$ and $N = 6$. The solid lines correspond to the far-field FMM computation, while the dashed lines correspond to the near-field direct computations.

Prolate Spheroid Flow

The vortex method described in the previous sections is used to simulate the turbulent separated flow past a 6:1 prolate spheroid at 0° and 20° angle of attack and Reynolds number $4.2 \cdot 10^6$. In both cases, the surface sheet mesh is $64 \times 16 \times 6$ in the axial, azimuthal and normal directions, respectively.

Computations have been run approximately until $t \approx 3$ assuming an impulsive start to the spheroid. Typical examples of the computed vortex filament structure of these flows are shown in figures 4, 5 and 6. In figure 4, it is shown the vortex filament structure of the 0° angle of attack flow at the end of the simulation. In figures 5 and 6, it is shown, from two different views, the time evolution of the vortex filament structure for the 20° angle of attack flow. It should be noted that there is also considerable levels of vorticity (not depicted) in the vortex sheets adjacent to the surface. At the end of the simulations, each solution has approximately 10^5 vortex elements and the computational cost is about 100 minute-nodes per time step on an Origin-2000.

At this high Reynolds number, the wakes are relatively narrow compared to flows at lower Reynolds numbers where large elaborate vortices are produced by the impulsive start. This illustrates one of the great strengths of the vortex method: the computational domain consists of just the relatively small support of the vorticity field, instead of the large volume where the velocity field varies. Another interesting feature of the wakes is the way in which the computed vortex filaments are seen to combine into larger structures such as are commonly observed in turbulent wakes. A careful examination of the time evolution of the computed flows shows the breakdown of the initial vorticity field into turbulence. In the case of flow at 20° angle of attack, the process unfolds first with the appearance of counter-rotating axial vortices on the leeward side of the prolate spheroid. These are formed as a result of the oblique flow over the surface which converges on the central axis. The calculations reveal the appearance of an oscillation in the vortex pair, which grows in amplitude until the vortices interact forming a larger structure. Rapid development of the turbulent wake follows as the initial structure convects off the body. Other attractive features of these solutions include sharp separation off the rear tip and persistence of the wake vortices as they shed off the body, which highlight the advantage of the non-diffusive character of vortex methods. In figure 7, it is shown the pressure distribution on the spheroid surface for the flow at 20° angle of attack. It is characterized by the turbulence and separation effect on the upper surface of the spheroid and a smoother variation on the lower surface.

Conclusion

An adaptive, fast, parallel vortex method was presented for the numerical simulation of turbulent flows in complex geometries. The implementation of a three-dimensional, fully-scalable FMM algorithm reduces dramatically the velocity computational time, thus permitting to perform vortex method computations with up to 10^6 vortex elements within reasonable time limits. This efficiency combined with the adaptive, grid-free, non-diffusive character of the vortex methodology presents an attractive tool in modeling turbulent flow.

Acknowledgements

Financial support by the Department of Energy under Award No. DE-FG02-97ER82413 is greatly appreciated. Parallel computation time was provided by the SGI/Cray Corporation and the High Performance Computing Research Facility, Mathematics and Computer Science Division, Argonne National Laboratory.

References

[1] Puckett, E. G., "Vortex Methods: An Introduction and Survey of Selected Research Topics," In *Incompressible Computational Fluid Dynamics: Trends and Advances*, M. D. Gunzburger and R. A. Nicolaides, ed., Cambridge University Press, 1993, pp. 335-407.

[2] Bernard, P. S., "Toward a Vortex Method Simulation of Non-Equilibrium Turbulent Flows," Proc. *ICASE/LaRC/AFOSR Symposium on Modeling Complex Turbulent Flows*, Kluwer Pub. Inc., 1998, to appear.

[3] Bernard, P. S., Thomas, J. M., and Handler, R. A., "Vortex Dynamics and the Production of Reynolds Stress," *J. Fluid Mech.*, Vol. 253, 1993, pp. 385-419.

[4] Brooke, J. W., and Hanratty, T. J., " Origin of Turbulence-Producing Eddies in a Channel Flow," *Phys. Fluids A*, Vol. 5, 1993, pp. 1011-1022.

[5] Kaftori, D., Hetsroni, G., and Banerjee, S., "Funnel-Shaped Vortical Structures in Wall Turbulence," *Phys. Fluids*, Vol. 6, 1994, pp. 3035-3050.

[6] Chorin, A. J., *Vorticity and Turbulence*, Springer-Verlag, 1994.

[7] Chorin, A. J., "Hairpin Removal in Vortex Interactions II," *J. Comput. Phys.*, Vol. 107, 1993, pp. 1-9.

[8] Greengard, L., and Rokhlin, V., "A Fast Algorithm for Particle Simulations," *J. Comput. Phys.*, Vol. 73, 1987, pp. 325-348.

[9] Carrier, J., Greengard, L., and Rokhlin, V., "A Fast Adaptive Multipole Algorithm for Particle Simulations," *SIAM J. Sci. Stat. Comput.*, Vol. 9, 1988, pp. 669-686.

[10] Strickland, J. H., and Baty, R. S., "A Three Dimensional Fast Solver for Arbitrary Vorton Distributions," *Technical Report SAND93-1641*, Sandia National Laboratories, 1993.

[11] Winckelmans, G. S., Salmon, J. K., Leonard, A., and Warren, M. S., "Three-Dimensional Vortex Particle and Panel Methods: Fast Tree-Code Solvers with Active Error Control for Arbitrary Distributions/Geometries," Proc. *Forum on Vortex Methods for Engineering Applications*, Albuquerque, New Mexico, 1995, pp. 23-43.

[12] Greengard, L., and Gropp, W. D., "A Parallel Version of the Fast Multipole Method," *Computers and Mathematics with Applications*, Vol. 20, 1990, pp. 63-71.

[13] Stalzer, M. A., "A Parallel Fast Multipole Method for the Helmholtz Equation," *Parallel Processing Letters*, Vol. 5, 1995, pp. 263-274.

[14] Bernard, P. S., "A Deterministic Vortex Sheet Method for Boundary Layer Flow," *J. Comput. Phys.*, Vol. 117, 1995, pp. 132-145.

[15] Panton, R. L., ed., "Self-Sustaining Mechanisms of Wall Turbulence," *Advances in Fluid Mechanics*, Vol. 15, Computational Mechanics Publications, 1995.

[16] Dimas, A. A., Collins, J. P., and Bernard, P. S., "A Fast, Parallel Vortex Method For Turbulent Flow Simulation," *Proc. ASME Fluids Engineering Division Summer Meeting*, Paper FEDSM-98-5000, Washington, DC, 1998, pp. 1-8.

Figure 4: Typical vortex filament structure of the turbulent separated flow past a 6:1 prolate spheroid at 0° angle of attack and $R = 4.2 \cdot 10^6$. Vortex elements are drawn as cylinders. Side view.

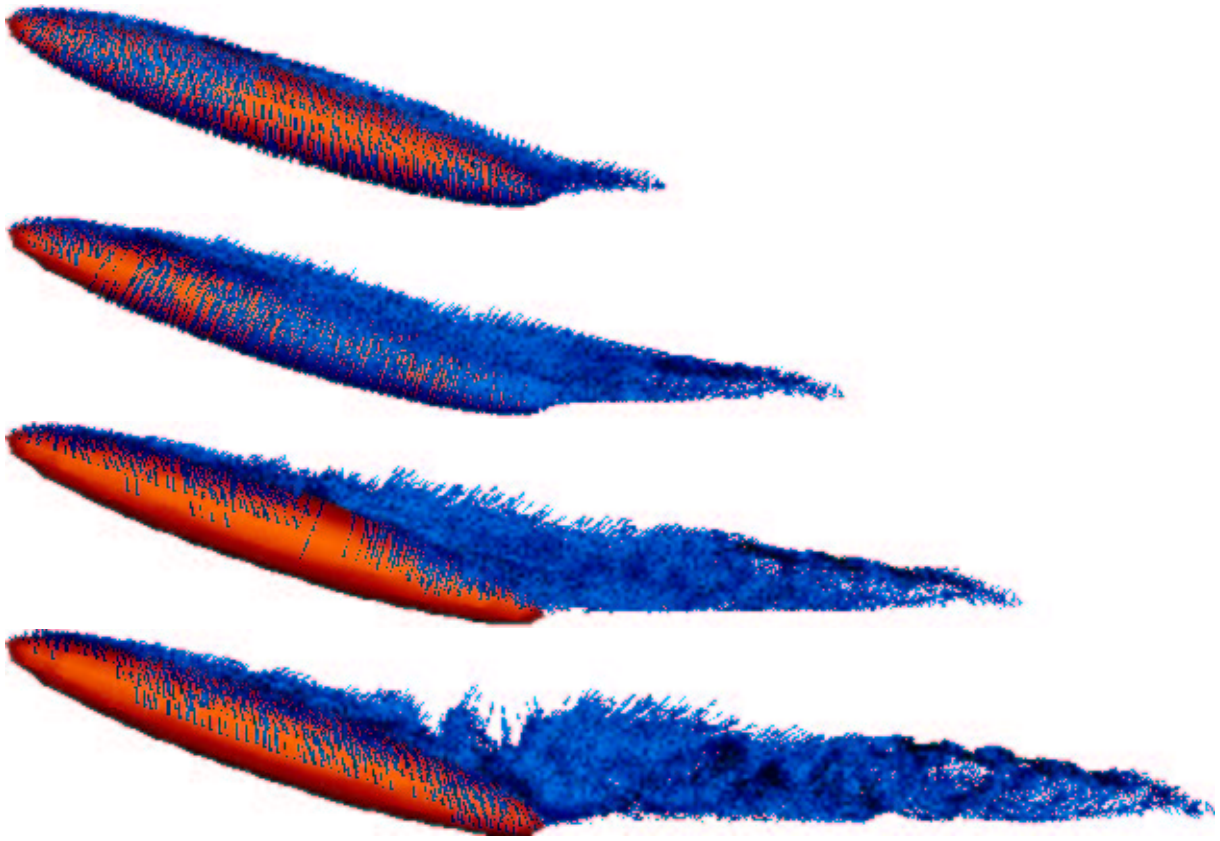

Figure 5: Time evolution (from top to bottom) of the vortex filament structure of the turbulent separated flow past a 6:1 prolate spheroid at 20° angle of attack and $R = 4.2 \cdot 10^6$. Side view.

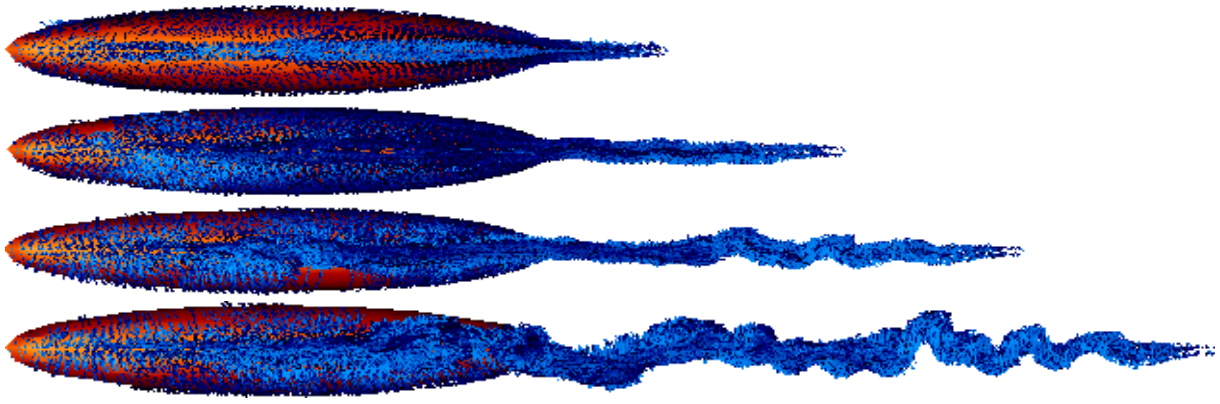

Figure 6: Time evolution (from top to bottom) of the vortex filament structure of the turbulent separated flow past a 6:1 prolate spheroid at 20° angle of attack and $R = 4.2 \cdot 10^6$. Top view.

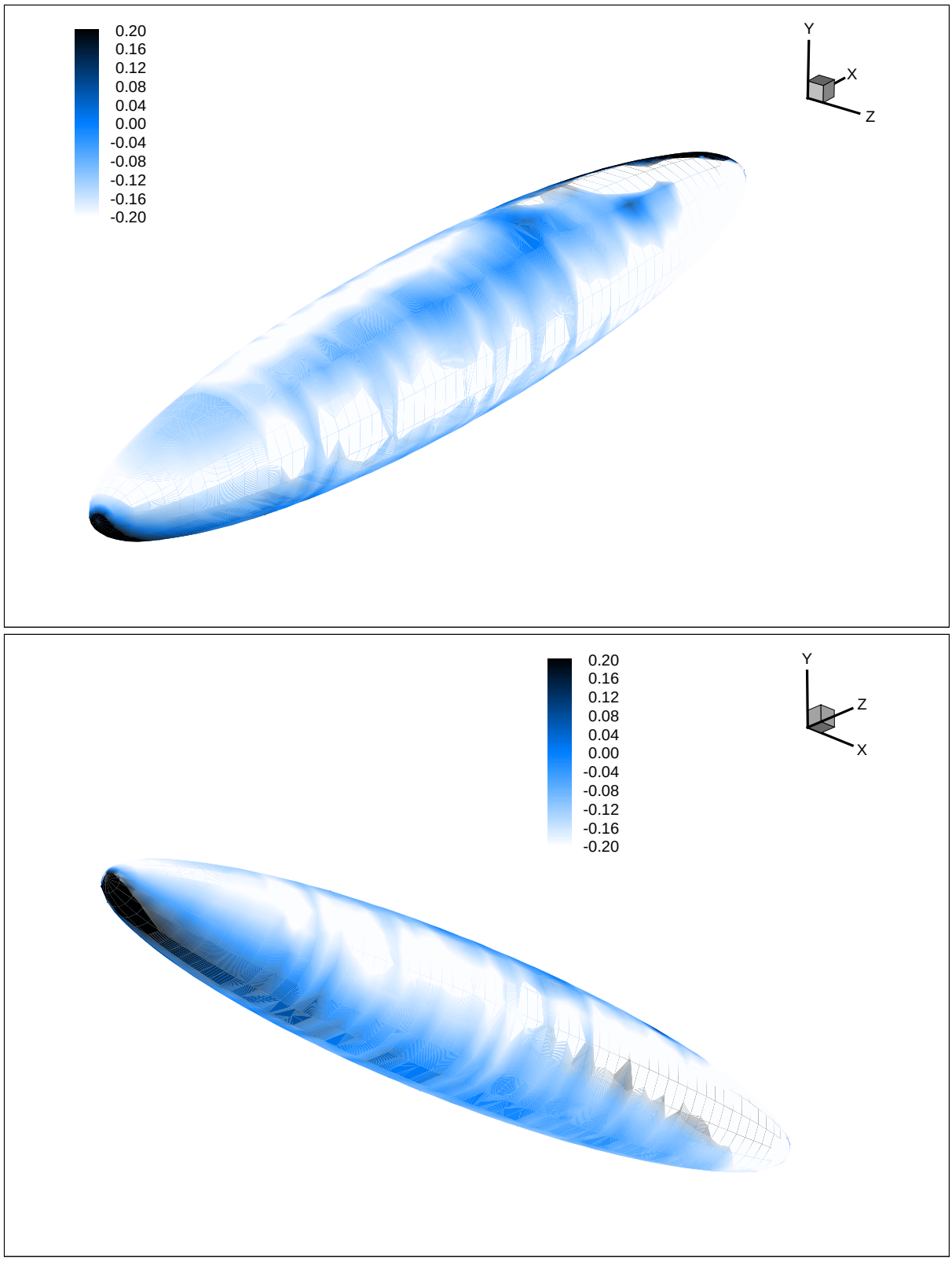

Figure 7: Pressure distribution on the prolate spheroid surface at the latest time instant shown in figures 5 and 6.