AIAA JOURNAL
Vol. 48, No. 9, September 2010

Efficient Adaptive Cartesian Vorticity Transport Solver for Vortex-Dominated Flows

Robert E. Harris,* Essam F. Sheta,† and Sami D. Habchi‡
CFD Research Corporation, Huntsville, Alabama 35805

DOI: 10.2514/1.J050472

An efficient solver for the velocity–vorticity form of the Navier–Stokes equations on adaptive Cartesian grids is presented. The excessive numerical dissipation common to most grid-based Navier–Stokes solvers is avoided by solving the fluid dynamic equations in vorticity conservation form. Additionally, an adaptive Cartesian solver is employed to efficiently capture and preserve vorticity on-the-fly as the flow develops. For practical purposes, this solver would be used in the wake region and coupled with a full Navier–Stokes solver in the near-body region, thus allowing vorticity to be accurately generated and then convected in the wake region with minimal dissipation. The adaptive Cartesian framework allows for the rapid evaluation of the velocity field using a fast-summation technique based on the Cartesian Treecode method. The implementations of both the solution algorithm and velocity calculation are described in detail. Results are presented for vortex convection applications that show good agreement with the analytical solution, and the accuracy of the scheme is verified numerically using a series of increasingly fine grids. Additionally, fully three-dimensional flow in the presence of a vortex ring is investigated and results are shown to be in very close agreement with published data.

I. Introduction

COMPUTATIONAL fluid dynamics has seen significant progress in recent years in modeling the underlying flow physics and complex aerodynamic behavior associated with helicopter rotor wakes. Such flows are dominated by unsteady shedding and interaction of vortical structures, as well as blade-vortex interactions (BVI), which require a sufficiently resolved high-fidelity calculation in which vortical structures are captured and preserved for several blade revolutions. Accomplishing this using a traditional primitive-variable formulation of the Navier–Stokes equations can be a difficult task, as these formulations are often prone to excessive numerical dissipation of vortical structures [1].

Velocity-vorticity formulations of the Navier–Stokes equations are becoming an increasingly popular alternative to the primitive-variable approach, and have been the recent focus of major research efforts [2–5]. These so-called Eulerian vorticity transport (EVT) formulations offer several advantages over primitive-variable formulations and since they deal with vorticity as the fundamental conserved quantity, there is inherently less smearing and dissipation of vortical structures than in a comparable primitive-variable Navier–Stokes solution. Interestingly, when cast in a noninertial frame, the EVT formulation has the significant advantage that the noninertial effects only enter into the solution through the implementation of initial and boundary conditions [6] Additionally, there are fewer primary equations in the EVT solution process and the equations themselves are simpler and involve fewer operations than in Navier–Stokes solutions. The EVT formulation typically involves a vorticity transport equation and a Poisson equation relating the velocity field to the vorticity field. Alternatively, the velocity field could be obtained directly from the Cauchy–Riemann equations as demonstrated in Gatski et al. [7] Typically, a solution to the Poisson equation for the velocity is sought, either using an iterative approach or an inversion resulting in the Biot–Savart integral representation. One of the most challenging aspects of the EVT formulation, as discussed by many researchers [2,7–10], is the proper implementation of solid wall boundary conditions on vorticity. Physically, the no-slip boundary condition means that vorticity is generated at solid walls, so some means of modeling the vorticity generation is necessary for accurate flow simulation. Furthermore, in the presence of such numerical boundary conditions, both at solid walls and far-field boundaries, additional steps must often be taken to ensure that the Cauchy–Riemann equations are properly satisfied by the numerical solution. Namely, a solution is sought in which the predicted velocity field is divergence-free and the predicted vorticity field is closely approximated by the curl of the predicted velocity field. More details about this can be found in Davies and Carpenter [2], Gatski et al. [7,8], Wu et al. [9], and Bertagnolio and Daube [10].

Recently, several different researchers have developed vorticity transport capabilities for rotorcraft flowfield prediction and vortical wake modeling. In particular, Brown [3] presented a grid-based Eulerian formalism for the vorticity equation including a lifting-line approximation to model vorticity generation from the rotor blades and fuselage. Using this technique, Brown demonstrated valid solutions for the blade loading and wake structure for isolated and interacting rotors in both hover and forward flight. Later, Brown and Line [4] extended this approach to enable variable resolution using a hierarchy of nested grid levels, and included a reformulated velocity calculation based on the fast multipole method (FMM) [11]. More recently, Stone et al. [5] presented a vortex particle method coupled with an Eulerian finite difference Navier–Stokes solver using overset grid communication. This work uses the Navier–Stokes solver for the near-body region, and the particle solver for the rotor wake region. Additionally, Stone was able to accelerate the Biot–Savart evaluation of the velocity field by using graphics processing unit computing techniques.

This paper describes the development of an Eulerian vorticity transport solver with adaptive mesh refinement (EVT-AMR) capabilities suitable for Octree-based adaptive Cartesian grids. Unlike Brown's approach using nested grid levels, the approach presented here supports completely local and automatic grid refinement and coarsening on-the-fly to capture and preserve vortical structures with high fidelity in the helicopter rotor wake region. In addition, an efficient approach for the rapid evaluation of the velocity field based on the Cartesian Treecode method is presented. This new framework

Presented as Paper 2010-1072 at the 48th AIAA Aerospace Sciences Conference, Orlando, FL, 4–7 January 2010; received 1 February 2010; revision received 7 May 2010; accepted for publication 24 May 2010.

*Senior Engineer, Aerospace & Defense Division; reh@cfdrc.com. Member AIAA.
†Manager, Aerospace & Defense Division; efs@cfdrc.com. Senior Member AIAA.
‡Executive Vice President, Aerospace & Defense Division; sdh@cfdrc.com. Senior Member AIAA.

should offer increased accuracy, as well as a dramatic reduction in both memory and CPU requirements for the simulation of heavily vortex-dominated flows. For rotorcraft simulations, the solver presented here will be used in the rotor wake region and coupled with a full Navier–Stokes solver in the near-body region, thus allowing vortical structures to be accurately generated at the rotor blades and fuselage, and then convected into the wake region with minimal dissipation. The wall boundary condition difficulties for EVT mentioned above are completely avoided here since the geometry is contained in the near-body Navier–Stokes region. Great importance will be placed on accurate prediction and preservation of vortical structures for many blade revolutions, including accurate prediction of blade-vortex interactions (BVI).

The paper is organized as follows. In Sec. II, we present the basic formulation of the EVT-AMR method. After that, an efficient approach for evaluation of the velocity field is described in detail in Sec. III, and the adaptive mesh refinement procedure is outlined in Sec. IV. Numerical results including accuracy and efficiency studies for vortex propagation, as well as two- and three-way vortex-merging problems are presented in Sec. V. In addition, both inviscid and viscous simulations of 3-D flow in the presence of a vortex ring are carried out and results are presented that agree very well with published data. Finally, some conclusions and a discussion of ongoing work are summarized in Sec. VI.

II. Finite Volume Formulation on Adaptive Cartesian Grid

Consider the Navier–Stokes equations written in velocity–vorticity form

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \mathbf{V}\cdot\nabla\boldsymbol{\omega} - \boldsymbol{\omega}\cdot\nabla\mathbf{V} = \nu\nabla^2\boldsymbol{\omega} \tag{1a}$$

on domain $\Omega \times [0, T]$ and $\Omega \subset R^3$ with the initial condition

$$\boldsymbol{\omega}(x, y, z, 0) = \boldsymbol{\omega}_0(x, y, z) \tag{1b}$$

and appropriate boundary conditions on $\partial\Omega$. In Eq. (1), x, y, and z are the Cartesian spatial coordinates and $(x, y, z) \in \Omega$, $t \in [0, T]$ denotes time, $\boldsymbol{\omega}$ is the vorticity vector, $\mathbf{V}$ is the velocity vector, and ν is the kinematic viscosity. Domain Ω is discretized into N nonoverlapping, potentially nonconformal, Cartesian cells, which are organized into a hierarchical cell-based Quadtree (2-D) or Octree (3-D) data structure [12]. A stencil showing such an arrangement of cells is shown in Fig. 1.

The unknowns, or degrees of freedom, are the volume-averaged components of the vorticity vector at the N cells. Define the cell-averaged vorticity for cell C_i as

$$\boldsymbol{\omega}_i = \frac{1}{\cancel{V_i}}\int_{C_i} \boldsymbol{\omega}\, \mathrm{d}\cancel{V}, \qquad i = 1, \ldots, N \tag{2}$$

where $\cancel{V_i}$ is the volume of C_i. Then integrating Eq. (1a) over C_i, we obtain

$$\frac{\partial \boldsymbol{\omega}_i}{\partial t}\cancel{V_i} + \sum_j (\mathbf{F}_I - \mathbf{F}_V) - (\boldsymbol{\omega}_i\cdot\nabla\mathbf{V}_i)\cancel{V_i} = 0 \tag{3}$$

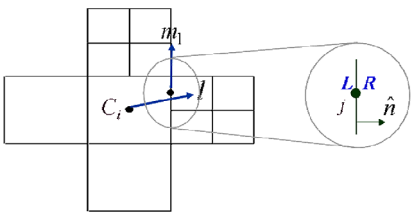

Fig. 1 Stencil showing arrangement of cells in a Quadtree grid. The current cell is denoted as C_i, and the faces bounding C_i are denoted as j. The vectors m_1 and l are used in the viscous flux formulation (m_2 points out of the page).

where $\mathbf{F}_I$ and $\mathbf{F}_V$ are the inviscid and viscous flux integrals for face j, respectively, and $\nabla\mathbf{V}_i$ is the cell-averaged velocity gradient for cell C_i. In the derivation of Eq. (3), the transport term is integrated by parts, and the flow is assumed to be incompressible. The second term in Eq. (3) is a sum over all faces bounding C_i. To avoid any ambiguity, the third term in Eq. (3) represents vortex stretching and has three components: i.e.,

$$\{(\boldsymbol{\omega}_i\cdot\nabla u_i)\cancel{V_i}, (\boldsymbol{\omega}_i\cdot\nabla v_i)\cancel{V_i}, (\boldsymbol{\omega}_i\cdot\nabla w_i)\cancel{V_i}\}^T$$

where u_i, v_i, and w_i are the Cartesian components of velocity. The inviscid flux integral in Eq. (3) necessitates the solution of a Riemann problem to resolve the distinct states immediately to the left and right of face j. Here, we employ either the upwind flux, or the Rusanov flux [13] shown in Eqs. (4a) and (4b), respectively:

$$\mathbf{F}_I \cong \begin{cases} A_j\boldsymbol{\omega}_L(\mathbf{V}_L\cdot\hat{n}), & (\mathbf{V}_L + \mathbf{V}_R)\cdot\hat{n} > 0 \\ A_j\boldsymbol{\omega}_R(\mathbf{V}_R\cdot\hat{n}), & (\mathbf{V}_L + \mathbf{V}_R)\cdot\hat{n} < 0 \end{cases} \tag{4a}$$

$$\mathbf{F}_I \cong \frac{A_j}{2}\left\{\boldsymbol{\omega}_L\left[\mathbf{V}_L\cdot\hat{n} + \frac{1}{2}|(\mathbf{V}_L + \mathbf{V}_R)\cdot\hat{n}|\right] + \boldsymbol{\omega}_R\left[\mathbf{V}_R\cdot\hat{n} - \frac{1}{2}|(\mathbf{V}_L + \mathbf{V}_R)\cdot\hat{n}|\right]\right\} \tag{4b}$$

where $\hat{n}$ is the outward unit normal vector for C_i, A_j is the area of face j, and the subscripts L and R denote states immediately to the left and right of face j, respectively. The left and right states are obtained using the approach optimized for Octree grids presented in Popinet [12]. This approach ensures that the spatial discretization is formally second-order-accurate. The viscous flux at a face f can be expressed as a function of the flow variables and their gradients:

$$\mathbf{F}_V \cong \nu A_j(\nabla\tilde{\omega}\cdot\hat{n}) \tag{5}$$

where the gradients $\nabla\tilde{\omega}$ are obtained as follows. Let $\hat{m}_1$ and $\hat{m}_2$ be orthogonal unit vectors tangent to the face and let $\hat{l}$ be the unit vector connecting the left cell and the right cell of a face, as shown in Fig. 1. The derivative of a variable in the $\hat{m}_1$ or $\hat{m}_2$ direction is obtained from the cellwise inviscid reconstruction: i.e.,

$$\frac{\partial\omega}{\partial m_1} = \frac{1}{2}(\nabla\omega_L\cdot\hat{m}_1 + \nabla\omega_R\cdot\hat{m}_1) \tag{6}$$

and

$$\frac{\partial\omega}{\partial m_2} = \frac{1}{2}(\nabla\omega_L\cdot\hat{m}_2 + \nabla\omega_R\cdot\hat{m}_2) \tag{7}$$

where $\nabla\omega_L$ and $\nabla\omega_R$ are the gradient in the left and right cells of the face from the inviscid reconstruction, and ω, here and in the following, is any component of the vorticity vector. Then $\partial\omega/\partial l$ is calculated from definition: i.e.,

$$\frac{\partial\omega}{\partial l} = \frac{\omega_R - \omega_L}{|\mathbf{r}_R - \mathbf{r}_L|} \tag{8}$$

Finally, $\nabla\tilde{\omega}$ is solved from the three equations:

$$\nabla\tilde{\omega}\cdot\hat{m}_1 = \frac{\partial\omega}{\partial m_1}, \qquad \nabla\tilde{\omega}\cdot\hat{m}_2 = \frac{\partial\omega}{\partial m_2}, \qquad \nabla\tilde{\omega}\cdot\hat{l} = \frac{\partial\omega}{\partial l} \tag{9}$$

While the above formulation generally involves a matrix inversion, it becomes greatly simplified for the Octree-based framework considered here, and a matrix inversion is not necessary. This viscous flux reconstruction has the advantage of compact support (i.e., using only data at the two cells sharing the face) and avoids an expensive separate reconstruction for viscous fluxes.

Rewriting Eq. (3) in terms of the residual operator $R(\boldsymbol{\omega}_i)$, we get

$$R(\boldsymbol{\omega}_i) = -\frac{1}{\cancel{V_i}}\sum_j (\mathbf{F}_I - \mathbf{F}_V) + (\boldsymbol{\omega}_i\cdot\nabla\mathbf{V}_i) \tag{10}$$

For time integration, we employ the second-order-accurate strong stability-preserving [14] Runge–Kutta scheme as follows:

$$\boldsymbol{\omega}^{(1)} = \boldsymbol{\omega}^n + \Delta t R(\boldsymbol{\omega}^n) \qquad \boldsymbol{\omega}^{n+1} = \tfrac{1}{2}(\boldsymbol{\omega}^{(1)} + \boldsymbol{\omega}^n + \Delta t R(\boldsymbol{\omega}^{(1)})) \quad (11)$$

where the time step Δt is computed by a Courant–Friedrichs–Lewy (CFL) condition based on the maximum eigenvalue of the inviscid flux Jacobian.

The differential form of the Biot–Savart relationship

$$\nabla^2 \mathbf{V} = -\nabla \times \boldsymbol{\omega} \quad (12)$$

relates the velocity at any point in the flow to the vorticity field. The solution to Eq. (12) can be written in integral form as

$$\mathbf{V}(\mathbf{x}) \approx \int_{\not V} K_\delta(\mathbf{x}, \mathbf{y}) \times \boldsymbol{\omega}(\mathbf{y})\, d\mathbf{y} \quad (13)$$

where

$$\mathbf{K}_\delta(\mathbf{x}, \mathbf{y}) = -\frac{1}{2\pi} \frac{(\mathbf{x} - \mathbf{y})}{|\mathbf{x} - \mathbf{y}|^2 + \delta^2} \qquad \text{(2D)} \quad (14a)$$

or

$$\mathbf{K}_\delta(\mathbf{x}, \mathbf{y}) = -\frac{1}{4\pi} \frac{(\mathbf{x} - \mathbf{y})}{(|\mathbf{x} - \mathbf{y}|^2 + \delta^2)^{3/2}} \qquad \text{(3D)} \quad (14b)$$

is the Rosenhead–Moore kernel, a regularized form of the Biot–Savart kernel [15,16], and the domain $\not V$ contains all of the vorticity in the flow. The exact kernel is recovered by setting the smoothing parameter δ to zero. In the present framework, the velocity is only evaluated at cell centers. Thus, the Biot–Savart kernel is guaranteed to be nonsingular and the smoothing parameter δ is set to zero for all cases considered here.

The pressure field is obtained by taking the divergence of the incompressible momentum equations and using the continuity equation. This gives a Poisson equation for pressure as

$$\nabla^2 p = -\rho \left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 + \left(\frac{\partial w}{\partial z}\right)^2 + 2\left(\frac{\partial v}{\partial x}\frac{\partial u}{\partial y} + \frac{\partial w}{\partial x}\frac{\partial u}{\partial z} + \frac{\partial w}{\partial y}\frac{\partial v}{\partial z}\right)\right] \quad (15)$$

where u, v, and w are the velocity components in x, y and z directions, respectively, and ρ is the density, which is taken to be constant.

III. Formulation for Efficient Evaluation of Velocity Field

The most computationally intensive aspect of the solution procedure given in the previous section is the evaluation of the velocity field in Eq. (13). This is essentially the classical n-body problem, which is ubiquitous in the physical and computational sciences. The simplest, and most naïve, method for evaluating Eq. (13) is the exhaustive summation of every discrete Biot–Savart interaction present in the entire domain, which results in a computational cost that is $\mathcal{O}(n^2)$. This cost is unacceptable for problems of practical interest, so we seek an alternative method for evaluating Eq. (13) at reduced cost. Over the past several decades, several new efficient procedures have been presented that significantly reduce the computational cost associated with the n-body problem. Some relevant approaches are the Treecode method [17,18], and the FMM [11]. These methods are capable of reducing the computational cost of the n-body problem from $\mathcal{O}(n^2)$ to $\mathcal{O}(n^* \log(n))$ or $\mathcal{O}(n)$. Here, we employ the Cartesian Treecode method of Lindsay and Krasny [18] because of the potential for efficient parallelization and scalability over the FMM method. In addition, the multigrid Poisson solver from Popinet [12] is employed to solve the differential form of the Biot–Savart law given in Eq. (12). The accuracy and efficiency of both methods for evaluating the velocity field are tested and rigorously compared in subsequent sections.

In the Treecode method, a hierarchical data structure is necessary to aid in the computation of the velocity field. A tree structure would typically need to be created specifically for this purpose, but it turns out that the tree structure [19] already being used to manage the computational grid can be readily adapted to serve this purpose as well, resulting in significant savings. The foundation of the Treecode method involves replacing the particle-particle interactions customary to the n-body problem, with a sum of particle-cluster interactions as in

$$\mathbf{V}(\mathbf{x}_i) \approx \sum_c \sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}_i, \mathbf{y}_j) \times (\boldsymbol{\omega}_j \not V_j) \quad (16)$$

where $c = \{\mathbf{y}_j, j = 1, \ldots, N_c\}$ denotes a cluster of particles and $\boldsymbol{\omega}_j$ and $\not V_j$ are the vorticity and volume associated with $\mathbf{y}_j$. To prevent any ambiguity, a particle in this context just refers to a point at which the velocity is to be calculated: namely, the centroid of each computational cell. A schematic of the above arrangement is given in Fig. 2.

The particle-cluster interaction given in Eq. (16) thus gives the velocity at point $\mathbf{x}_i$ induced by the cluster c. The interactions are evaluated using either Taylor approximation or direct summation as follows. First, referring to Eq. (16), we expand $\mathbf{K}_\delta(\mathbf{x}_i, \mathbf{y}_j)$ in a Taylor series with respect to $\mathbf{y}$ about the cluster center $\mathbf{y}_c$, giving

$$\begin{aligned} \sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}_i, \mathbf{y}_j) \times (\boldsymbol{\omega}_j \not V_j) &= \sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}_i, \mathbf{y}_c + (\mathbf{y}_j - \mathbf{y}_c)) \times (\boldsymbol{\omega}_j \not V_j) \\ &= \sum_{j=1}^{N_c} \sum_k \frac{1}{k!} D_{\mathbf{y}}^k \mathbf{K}_\delta(\mathbf{x}_i, \mathbf{y}_c)(\mathbf{y}_j - \mathbf{y}_c)^k \times (\boldsymbol{\omega}_j \not V_j) \\ &= \sum_k \mathbf{a}_k(\mathbf{x}_i, \mathbf{y}_c) \times \mathbf{m}_k(c) \end{aligned} \quad (17)$$

where $k = (k_1, k_2, k_3)$, $D_{\mathbf{y}}^k = \partial / \partial y_1^{k_1} \partial y_2^{k_2} \partial y_3^{k_3}$, $|k| = k_1 + k_2 + k_3$, $k! = k_1! k_2! k_3!$, and $\mathbf{x}^k = x_1^{k_1} x_2^{k_2} x_3^{k_3}$ for $k_i \geq 0$, subscripts 1–3 refer to the Cartesian directions,

$$\mathbf{a}_k(\mathbf{x}_i, \mathbf{y}_c) = \frac{1}{k!} D_{\mathbf{y}}^k \mathbf{K}_\delta(\mathbf{x}_i, \mathbf{y}_c) \quad (18)$$

is the kth Taylor coefficient of $\mathbf{K}_\delta(\mathbf{x}_i, \mathbf{y})$ at $\mathbf{y} = \mathbf{y}_c$, and

$$\mathbf{m}_k(c) = \sum_{j=1}^{N_c} (\mathbf{y}_j - \mathbf{y}_c)^k (\boldsymbol{\omega}_j \not V_j) \quad (19)$$

is the kth moment of cluster c about its center $\mathbf{y}_c$. The approximation is obtained by truncating the infinite series in Eq. (17):

$$\sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}_i, \mathbf{y}_j) \times (\boldsymbol{\omega}_j \not V_j) \approx \sum_{|k|<p} \mathbf{a}_k(\mathbf{x}_i, \mathbf{y}_c) \times \mathbf{m}_k(c) \quad (20)$$

where the order p is chosen to ensure that the accuracy of the velocity calculation is consistent with the formal order of accuracy of the solver. We refer to the right side of Eq. (20) as a pth-order particle-cluster approximation, where the error is bounded by the following multipole acceptance criterion (MAC):

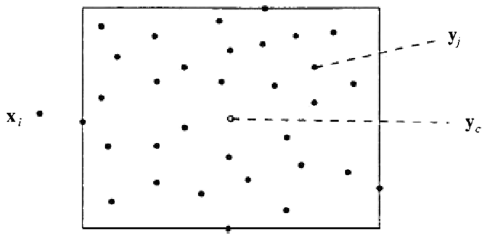

Fig. 2 Schematic taken from Lindsay and Krasny [18] depicting a particle $\mathbf{x}_i$ and a disjoint cluster $c = \{\mathbf{y}_j, j = 1, \ldots, N_c\}$; $\mathbf{y}_c$ is taken as the geometric center of the box.

$$(r/R) \leq \theta \tag{21}$$

where r the cluster radius, R is the regularized distance between the particle and the cell center, and θ is a user-specified empirical parameter for controlling the error. Note that the Taylor coefficients are independent of the particles $\mathbf{y}_j$ in cluster c, and the cluster moments are independent of the particle $\mathbf{x}_i$.

These features give rise to an efficient solution algorithm as discussed below. For each particle $\mathbf{x}_i$, the cell tree is traversed starting at the root and the MAC is computed from Eq. (21). If the MAC is not satisfied, the recursive tree traversal continues and the MAC is recomputed. Once the MAC is satisfied, the Taylor coefficients for $|k| < p$ are computed using the recurrence relation given in Draghicescu and Draghicescu [20] for 2-D or Lindsay and Krasny [18] for 3-D. The cluster moments are then computed for $|k| < p$, unless they are already available from a previous interaction. When the moments of a particular cluster c are first computed, they are stored and used again in subsequent interactions between c and other particles $\mathbf{x}_i$. This procedure concludes with the summation of the truncated series on the right-hand side of Eq. (20). To save time and memory, if during the above procedure a cluster contains $\leq N_0$ particles, the particle-cluster velocity is computed by direct summation. N_0 is a user-specified parameter to control both the cost and accuracy of the Treecode. For additional information on the Treecode, including a rigorous error analysis and discussion of the scalability in terms of polynomial order p, refer to Lindsay and Krasny [18].

IV. Adaptive Mesh Refinement

The adaptive mesh refinement strategy employed here is described in detail in Popinet [12], but we review the procedure here for completeness. In the first step, all the leaf cells that satisfy a given criterion are refined. This includes neighboring cells when necessary, in order to satisfy the neighboring cell depth requirements of the Octree grid. In the second step, the parent cells of all the leaf cells are considered, and all of these cells that do not satisfy the refinement criterion are coarsened. The solution quantities for newly coarsened cells are computed as volume-weighted averages from their former children, and for newly created cells the solution quantities are computed from a linear interpolation using the parent cell value and its gradients.

The refinement criterion considered here is based on gradients in vorticity. At each adaptation step, the adaptive cell cost is computed as the norm of the local vorticity gradient multiplied by the cell size. Then if the adaptive cell cost exceeds a user-defined maximum allowable cell cost, the cell is refined. Additionally, the grid will not be refined past a user-defined maximum refinement level or maximum number of cells. The computational cost of this algorithm is small compared to the cost of the velocity calculation. It can be applied at every time step with a negligible overall penalty of less than 5% of the total cost, while in practice every 10 or 100 steps is usually sufficient depending on the application.

V. Results

In this section, the efficient EVT-AMR approach is evaluated for both 2-D and 3-D inviscid and viscous flow problems. Several test cases are selected to elucidate both the accuracy and efficiency EVT-AMR formulation. All cases considered here employ the Rusanov flux [13] given by Eq. (4b) as the inviscid flux formulation, and all errors presented are time-step-independent because the time step Δt was made small enough so that the errors are dominated by the spatial discretization.

A. Accuracy Study with Vortex Propagation Problem

To verify that the EVT-AMR method is formally second-order-accurate, we employ the problem of vortex convection by a uniform flow. This is a test of the inviscid EVT-AMR approach using the Treecode method with $p = 4$, and $\theta = 0.3$ for evaluation of the velocity field. This case was executed on a machine running LINUX CentOS 4.4 with a dual core 3.0 GHz AMD64 Athlon processor and 4 GB memory. The required CPU time for this case is 8.3 μs per cell per time step (two Runge–Kutta stages). The vortex is allowed to convect until $t = 2$, and then the vorticity distribution is compared to the analytical solution. Table 1 shows the L_1 and L_∞ errors and orders for this case. As the grid is refined, it is evident that the results are exceeding second-order accuracy, which is the expected accuracy of the scheme. To compare the vorticity profiles after a longer time has passed, this problem is revisited using a periodic domain and multigrid Poisson solution for evaluation of the velocity field. Figure 3 shows the final vorticity distribution at $y = 0$ after one complete pass compared to the analytical solution on two different grids. The grids contain 64×64 cells and 128×128 cells, with about 8 and 16 cells across the vortex core, respectively. The vortex core in this study is defined as the distance from the center of the vortex to the point where the vorticity drops below 10% of its envelope. It is apparent that 16 cells across the vortex core preserve the vortex for one pass through a periodic domain.

The performance of the multigrid Poisson solver and Treecode methods for evaluating the velocity field is now compared by looking at the CPU time for evaluating a single velocity field for an isotropic vortex on various grids. Figure 4 shows the results of this study on a log–log plot with power fits of the data included. For Treecode cases when $p = 0$, the performance of the Treecode is comparable to multigrid for high values of the multipole acceptance criteria (MAC). The direct sum method scales as $\mathcal{O}(n^2)$ as expected, while the multigrid Poisson solver scales as $\mathcal{O}(n^{1.04})$, and the best of the Treecode results scales like $\mathcal{O}(n^{1.11})$, with $n^* \log(n)$ scaling like $\mathcal{O}(n^{1.05})$, in this region. This shows that the Treecode is approaching the theoretical $n^* \log(n)$ performance. Although the multigrid Poisson solver scales slightly better for this case, it should be noted that when solving the velocity by the Poisson equation given by Eq. (12), the application of numerical boundary conditions can result in a violation of the continuity equation, which can spread throughout the domain if proper steps are not taken. This can be remedied by using a projection method to ensure a divergence-free velocity field. Such methods are discussed in Davies and Carpenter [2] in which the velocity field is represented using a Helmholtz decomposition and projected onto a divergence-free field. Some details of similar methods can be found in Gatski et al. [7,8], Wu et al. [9], and Bertagnolio and Daube [10]. The integral formulation given by Eq. (13) does not necessitate direct application of boundary conditions, and thus the flowfield remains divergence-free. All subsequent results presented use the Treecode method for velocity field evaluation with $p = 4$, and $\theta = 0.3$. These parameters were found to give sufficient accuracy for our application.

B. Two- and Three-Way Vortex Merging Problems

As an additional test case for the inviscid EVT-AMR approach, the merging of two vortices is considered. This case involves two Gaussian vortices in close proximity in which the flowfield evolves solely due to the mutual interaction of the vortices. In this case, we employ up to 11 levels of local adaptive mesh refinement based on gradients in vorticity, and use CFL = 0.5 to compute Δt. The vorticity contours and computational grids for this case are shown in Figs. 5 and 6, respectively. The AMR procedure produces highly resolved flow features at a very low computational cost when compared to a global refinement strategy, as the grid is automatically refined locally when more resolution is necessary and coarsened when less resolution is required.

The problem of three-way vortex merging is also considered, using similar conditions to the two-way vortex-merging case. Here,

Table 1 L_1 and L_∞ errors and orders at $t = 1$ for vortex convection problem

Grid	L_1 error	L_1 order	L_∞ error	L_∞ order
16×16	2.05E − 01	——	9.40E + 00	——
32×32	9.13E − 02	1.16	7.45E + 00	0.34
64×64	2.46E − 02	1.89	2.42E + 00	1.62
128×128	4.78E − 03	2.36	4.50E − 01	2.43

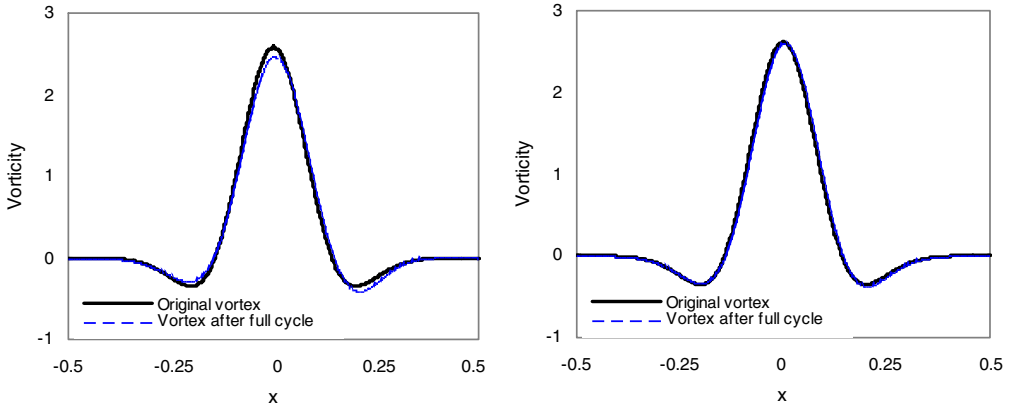

Fig. 3 Vorticity distribution at $y = 0$ after one full cycle: a) 64 × 64 cells (eight cells across vortex core) and b) 128 × 128 cells (16 cells across vortex core).

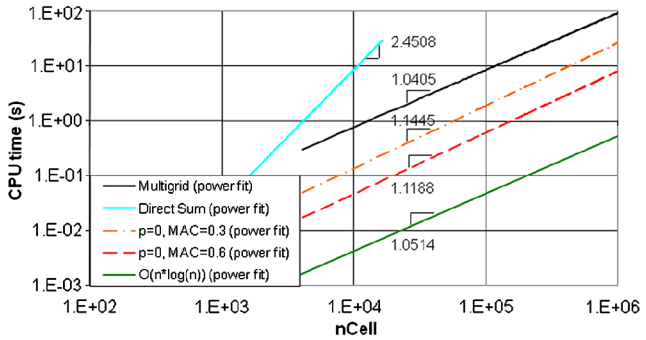

Fig. 4 Comparison of CPU times for evaluation of a single velocity field using the Treecode, multigrid Poisson solver, and direct summation.

three smaller vortices surround a larger central vortex of opposite sign. This arrangement causes the smaller vortices to orbit the central vortex and subsequent interactions between the vortices occur. The vorticity contours and computational grids for this case are shown in Figs. 7 and 8, respectively. Again, the adaptive mesh refinement procedure produces highly resolved flow features at a very low computational cost when compared to a global refinement strategy.

C. Evolution of a Vortex Ring at an Intermediate Reynolds Number

Finally, as a rigorous test of both the inviscid and viscous formulations of the EVT-AMR approach for fully 3-D flow problems, we consider the flowfield in the presence of a vortex ring. The initial geometry of the ring is shown in Fig. 9.

This case involves a single vortex ring at $Re_\Gamma = \Gamma/\nu = 500$. A ring of radius R and core radius a is initially placed at the $y = 0$ plane, and the core of the ring is represented as

$$\omega_\phi = \frac{K}{\pi}\frac{\Gamma}{a^2}\exp\left\{-K\left(\frac{R^2}{a^2}+\frac{r^2}{a^2}-\frac{2Rr}{a^2}\sin\theta\right)\right\} \tag{22}$$

where $r = \sqrt{x^2+y^2+z^2}$, $\tan\theta = (\sqrt{x^2+z^2})/y$, and $K = (2.24182)^2/4$. The ring has unit circulation $\Gamma = 1$, and the ring radius taken to be $R = 0.1$. The core radius is chosen to be $a/R = 0.35$ to be consistent with that used by Stanaway et al. [21,22]. The extent of the domain is $-1/2 \le x \le 3/2$, and $-1/2 \le y, z \le 1/2$. The ring resides in an initially quiescent flow and ultimately induces a velocity upon itself due to the presence of the vorticity field and begins moving in the x direction. This case has been studied extensively by Stanaway et al. [21,22], Wee and Ghoniem [23], and others, and there is a large amount of published data for use in assessing the validity of our numerical results.

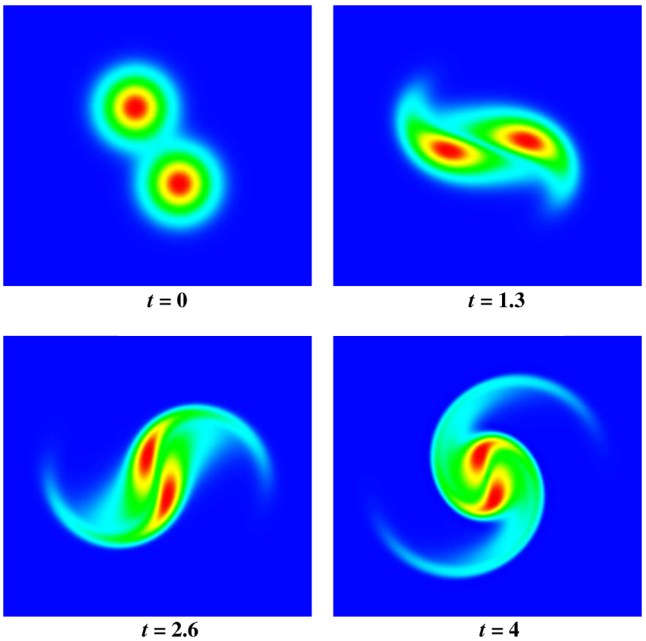

Fig. 5 Vorticity contours at various time instances for problem of two merging Gaussian vortices.

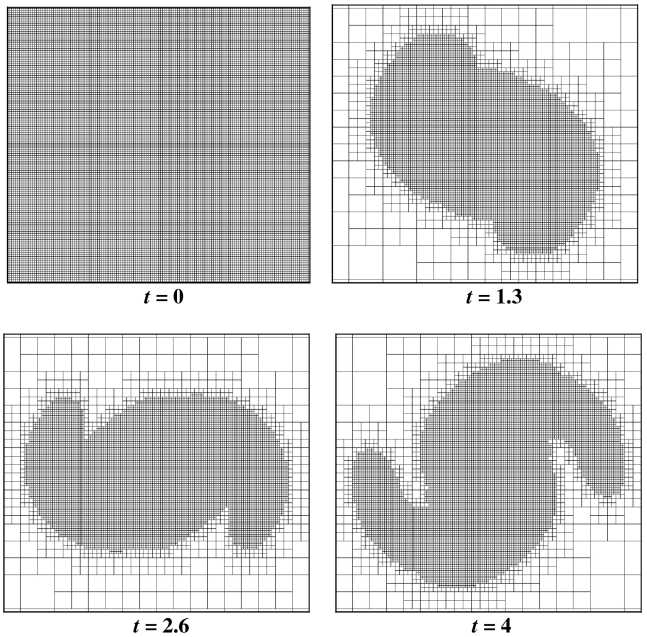

Fig. 6 Computational grid at various time instances for problem of two merging Gaussian vortices.

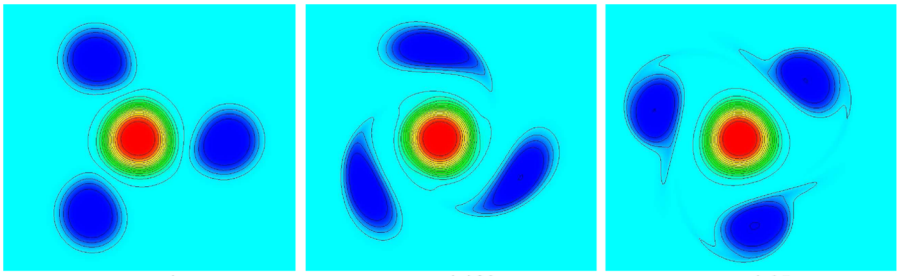

Fig. 7 Vorticity contours at various times for three-way vortex merging.

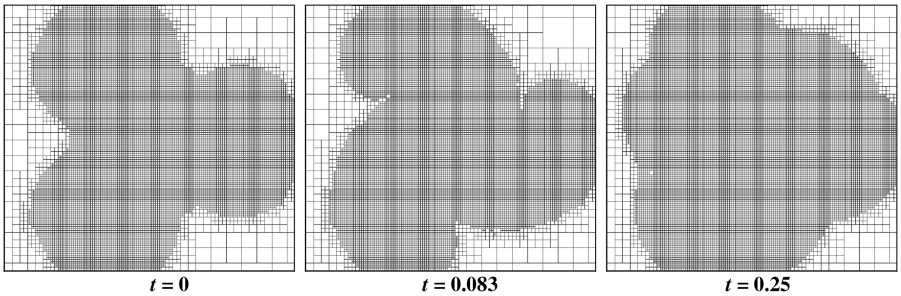

Fig. 8 Computational grid at various times for three-way vortex merging.

A relevant metric often used for assessing the accuracy of a scheme for vortex ring problems, is the induced velocity at the ring centroid. The generally accepted definition for the position of the ring centroid with respect to time was given in Stanaway et al. [21,22] as

$$\mathbf{X} = \frac{1}{2}\int_V \frac{\mathbf{r}\times\boldsymbol{\omega}\cdot\mathbf{I}}{|I|^2}\mathbf{r}\,\mathrm{d}V \tag{23}$$

where

$$\mathbf{I}/\rho = \frac{1}{2}\int_V \mathbf{r}\times\boldsymbol{\omega}\,\mathrm{d}V \tag{24}$$

is the impulse per unit density, and $\mathbf{r}$ is a position vector with respect to a fixed reference point. Since the computational domain is assumed to contain all of the vorticity in the flow, the impulse given by Eq. (23) does not change with time during the simulation. The ring speed $\mathbf{U}_c$ is then given by the temporal rate of change of Eq. (23), which results in

$$\mathbf{U}_c = \frac{1}{2}\int_V \frac{\mathbf{r}\times R(\boldsymbol{\omega})\cdot\mathbf{I}}{|I|^2}\mathbf{r}\,\mathrm{d}V \tag{25}$$

where $R(\boldsymbol{\omega})$ is the finite volume residual given by Eq. (10).

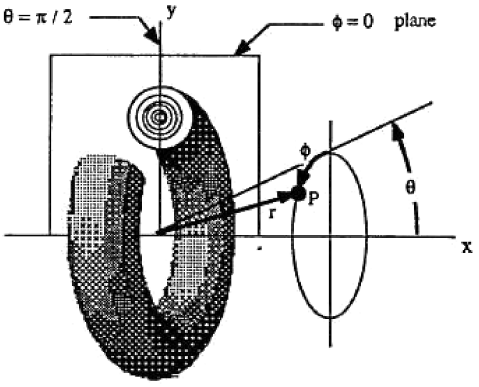

Fig. 9 Schematic of a vortex ring in spherical coordinates from Stanaway et al. [21,22] The domain is infinite, where $0 \le r \le \infty$, $0 \le \theta \le \pi$, and $0 \le \varphi \le 2\pi$. The cross section indicates lines of constant vorticity.

We first consider the inviscid EVT formulation and then extend to the viscous EVT formulation in order to study the significance of viscosity on the temporal evolution of ring. The results are reported in the following dimensionless variables, which were also used in Stanaway et al. [21,22] and Wee and Ghoniem [23]. The dimensionless speed of the vortex ring centroid is given by

$$\mathbf{U} = \mathbf{U}_c\frac{(I_0/\rho)^{1/2}}{\nu^{3/2}} \tag{26}$$

where I_0 is the initial linear impulse of the ring, and $\mathbf{U}_c$ is the speed in computational units given by Eq. (25). We use CFL = 0.5, and a dimensionless time, which is scaled as

$$\bar{t} = t\frac{\nu^2}{I_0/\rho} \tag{27}$$

and shifted to match the initial time reported in Stanaway et al. [21,22].

In Fig. 10, the dimensionless speed of the vortex ring centroid is plotted using both the inviscid and viscous EVT-AMR approaches. For comparison, the curve reported in Stanaway et al. [21,22] is also shown. It is evident that the inviscid formulation is unable to predict the speed of the ring centroid as this problem becomes heavily

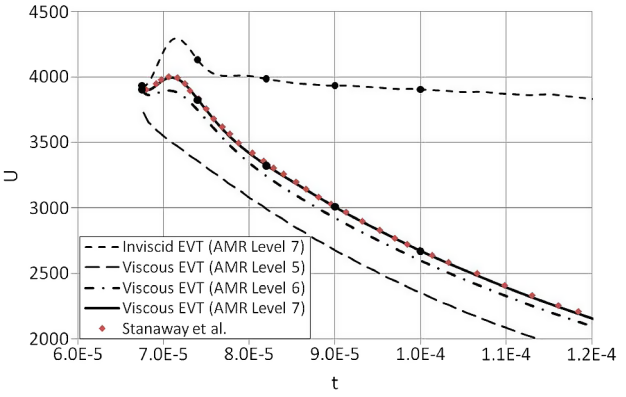

Fig. 10 Speed of the vortex ring centroid versus time. Diamonds correspond to computational study by Stanaway et al. [21,22], and dots on the curves correspond to the instances shown in Fig. 11.

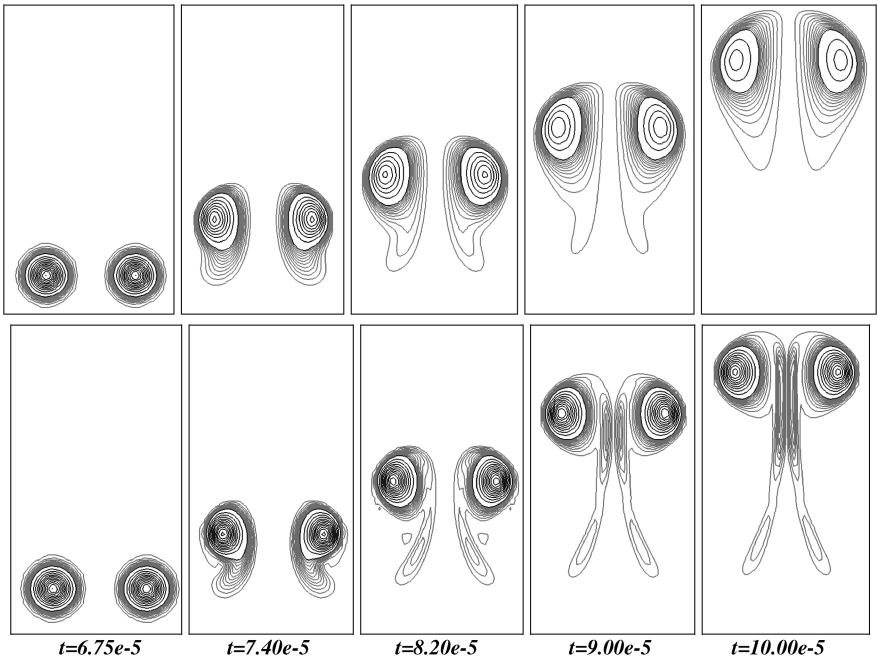

Fig. 11 Vorticity contours at several time instances for viscous EVT (top) and inviscid EVT (bottom). The contours levels for gray lines vary from 3.15 to 31.5, while the contour levels for black lines vary from 31.5 to 315. For lines of the same type, the vorticity varies linearly.

dominated by viscous effects, which effectively slow the ring down as time progresses. This viscous formulation, on the other hand, is able to sufficiently predict the speed of the ring centroid and the results show very close agreement with that predicted in Stanaway et al. [21,22]. In this study, we present three different mesh refinement AMR levels 5, 6, and 7, which correspond to 3, 6, and 12 cells across the vortex core at the finest grid level. The core size is computed as the distance until the vorticity magnitude drops below 10% of its envelope. It is interesting to note that although the coarsest results (AMR level 5) do not match the data vary well, the general trend is still represented even with only three cells across the vortex core.

We also show vorticity contours on the $z = 0$ plane in Fig. 11, and isosurfaces in Fig. 12 for this case. We have chosen similar instances as those reported in Stanaway et al. [21,22] for a better comparison. The dots on the curves in Fig. 10 correspond to the time instances shown in Figs. 11 and 12. The change in vorticity between darker (black) contours is a factor of 10 larger than between lighter (gray) contours, but the vorticity varies linearly for lines of the same type. Results for both the inviscid and viscous EVT-AMR approaches exhibit a wake region of shed vorticity under the vortex ring as it rises, with the numerical wake in the inviscid results appearing to be larger than the physical wake in the viscous results, and this disparity only gets more pronounced as time progresses. Again, the contours for the viscous EVT-AMR approach agree very favorably with the results presented in Stanaway et al. [21,22] for all time instances shown.

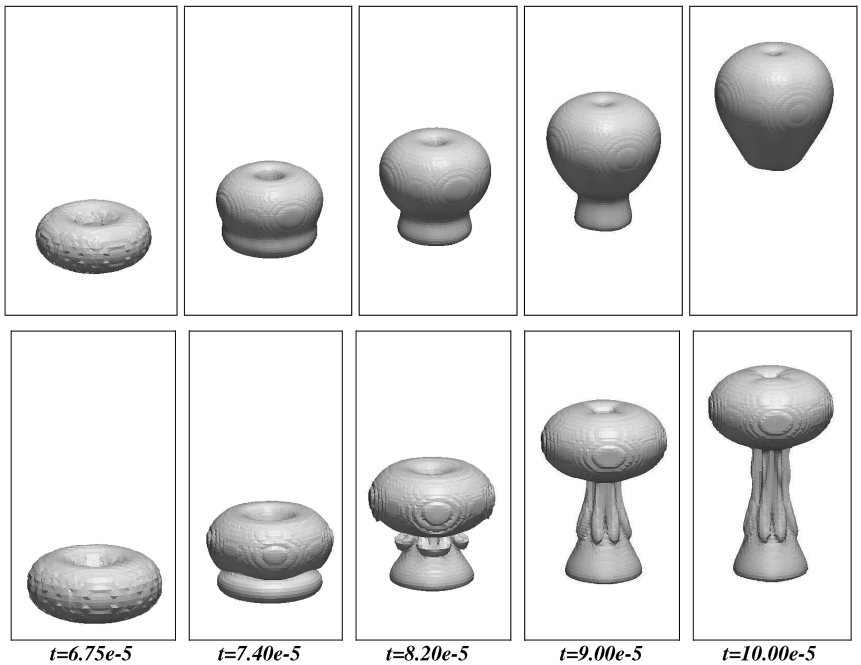

Fig. 12 Vorticity isosurfaces at several time instances for viscous EVT (top) and inviscid EVT (bottom). The contour levels presented are the same as in Fig. 11.

VI. Conclusions

An efficient Eulerian vorticity transport solver with Octree-based adaptive mesh refinement capabilities (EVT-AMR) has been successfully implemented and demonstrated for solutions of the vorticity transport equation for both inviscid and viscous flows in 2-D and 3-D. Two different approaches for efficient evaluation of the velocity field, a multigrid Poisson solver and a Cartesian Treecode approach, have been developed and evaluated for several test cases and were shown to be very accurate and efficient. A study of vortex convection by a uniform flow was carried out and the results successfully verified the formal second-order accuracy of the EVT-AMR approach. In addition, both inviscid and viscous simulations of 3-D flow in the presence of a vortex ring were carried out and results were presented that show very close agreement with published data. The EVT-AMR capability is currently being coupled to a Navier–Stokes solver using overset grids to allow accurate vorticity generation in the near-body region and efficient convection of vortical structures with minimal dissipation in the wake region for rotorcraft simulations.

Acknowledgments

This study is being supported by the U.S. Army Aeroflightdynamics Directorate (AFDD) and contracted through the U.S. Army Aviation Applied Technology Directorate. The authors would like to thank Venke Sankaran of AFDD for many fruitful discussions relating to this work. The authors would also like to thank Robert Krasny of the University of Michigan for several helpful discussions pertaining to the Treecode method for evaluation of the velocity field. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Army.

References

[1] Caradonna, F. X., "Developments and Challenges in Rotorcraft Aerodynamics," AIAA Paper 2000-0109, Jan. 2000.

[2] Davies, C., and Carpenter, P., "A Novel Velocity-Vorticity Formulation of the Navier–Stokes Equations with Applications to Boundary Layer Disturbance Evaluation," *Journal of Computational Physics*, Vol. 172, 2001, pp. 119–165. doi:10.1006/jcph.2001.6817

[3] Brown, R. E., "Rotor Wake Modeling for Flight Dynamic Simulation of Helicopters," *AIAA Journal*, Vol. 38, No. 1, 2000, pp. 57–63. doi:10.2514/2.922

[4] Brown, R. E., and Line, A. J., "Efficient High-Resolution Wake Modeling Using the Vorticity Transport Equation," *AIAA Journal*, Vol. 43, No. 7, 2005, pp. 1434–1443. doi:10.2514/1.13679

[5] Stone, C., Duque, E., Hennes, C., and Gharakhani, A., "Rotor Wake Modeling with a Coupled Eulerian and Vortex Particle Method," AIAA Paper 2010-312, 2010.

[6] Speziale, C. G., "On the Advantage of the Vorticity-Velocity Formulation of the Equations of Fluid Dynamics," *Journal of Computational Physics*, Vol. 73, 1987, p. 476. doi:10.1016/0021-9991(87)90149-5

[7] Gatski, T. B., Grosch, C. E., and Rose, M. E., "A Numerical-Solution of the Navier–Stokes Equations for 3-Dimensional, Unsteady, Incompressible Flows by Compact Schemes," *Journal of Computational Physics*, Vol. 82, No. 2, 1989, pp. 298–329. doi:10.1016/0021-9991(89)90051-X

[8] Gatski, T. B., Grosch, C. E., and Rose, M. E., "A Numerical Study of the 2-Dimensional Navier–Stokes Equations in Vorticity-Velocity Variables," *Journal of Computational Physics*, Vol. 48, No. 1, 1982, pp. 1–22. doi:10.1016/0021-9991(82)90032-8

[9] Wu, X. H., Wu, J. Z., and Wu, J. M., "Effective Vorticity-Velocity Formulations for 3-D Incompressible Viscous Flows," *Journal of Computational Physics*, Vol. 122, No. 1, 1995, pp. 68–82. doi:10.1006/jcph.1995.1197

[10] Bertagnolio, F., and Daube, O., "Solution of the DIV-Curl Problem in Generalized Curvilinear Coordinates," *Journal of Computational Physics*, Vol. 138, No. 1, 1997, pp. 121–152. doi:10.1006/jcph.1997.5800

[11] Greengard, L., and Rokhlin, V., "A Fast Algorithm for Particle Simulations," *Journal of Computational Physics*, Vol. 73, No. 2, 1987, pp. 325–348. doi:10.1016/0021-9991(87)90140-9

[12] Popinet, S., "Gerris: A Tree-Based Adaptive Solver for the Incompressible Euler Equations in Complex Geometries," *Journal of Computational Physics*, Vol 190, 2003, pp. 572–600. doi:10.1016/S0021-9991(03)00298-5

[13] Rusanov, V. V., "Calculation of Interaction of Nonsteady Shock Waves with Obstacles," *Computational Mathematics and Mathematical Physics*, Vol. 1, 1961, pp. 267–279.

[14] Gottlieb, S., Shu, C-W., and Tadmor, E., "Strong Stability-Preserving High-Order Time Discretization Methods," *SIAM Review*, Vol. 43, No. 1, 2001, pp. 89–112. doi:10.1137/S003614450036757X

[15] Rosenhead, L., "The Spread of Vorticity in the Wake Behind a Cylinder," *Proceedings of the Royal Society of London A*, Vol. 127, No. 806, 1930, pp. 590–612. doi:10.1098/rspa.1930.0078

[16] Moore, D. W., "Finite Amplitude Waves on Aircraft Trailing Vortices," *Aeronautical Quarterly*, Vol. 23, No. 1972, p. 307.

[17] Barnes, J., and Hut, P., "A Hierarchical $\mathcal{O}(N \log N)$ Force-Calculation Algorithm," *Nature*, Vol. 324, 1986, pp. 446–449. doi:10.1038/324446a0

[18] Lindsay, K., and Krasny, R., "A Particle Method and Adaptive Treecode for Vortex Sheet Motion in Three-Dimensional Flow," *Journal of Computational Physics* Vol. 172, 2001, pp. 879–907. doi:10.1006/jcph.2001.6862

[19] Khokhlov, A. M., "Fully Threaded Tree Algorithms for Adaptive Refinement Fluid Dynamic Simulations," *Journal of Computational Physics* Vol. 143, No. 2, 1998, pp. 519–543. doi:10.1006/jcph.1998.9998

[20] Draghicescu, C., and Draghicescu, M., "A Fast Algorithm for Vortex Blob Interactions," *Journal of Computational Physics*, Vol 116, 1995, pp. 69–78. doi:10.1006/jcph.1995.1006

[21] Stanaway, S. K., Cantwell, B. J., and Spalart, P. R., "Navier–Stokes Simulations of Axisymmetric Vortex Rings," AIAA Paper 88-0318, 1988.

[22] Stanaway, S. K., Cantwell, B. J., and Spalart, P. R., "A Numerical Study of Viscous Vortex Rings Using a Spectral Method," NASA TM 101041, 1988.

[23] Wee, D., and Ghoniem, A. F., "Modified Interpolation Kernels for Treating Diffusion and Remeshing in Vortex Methods," *Journal of Computational Physics*, Vol. 213, 2006, pp. 239–263. doi:10.1016/j.jcp.2005.08.009

W. Anderson
Associate Editor