An Automated, Adaptive, Unstructured, Cartesian-Prism-Based Technique for Moving-Boundary Simulations

Z. J. Wang (zjw@cfdrc.com)
S. A. Bayyuk (sab@cfdrc.com)

CFD Research Corporation
Huntsville, AL 35805, USA

Abstract

A 2-D, solution-adaptive computational technique for generic moving-boundary problems, including ones with topologic transformations, is presented. The geometry of boundaries is represented by Composite Parametric Splines. Each boundary is encapsulated by a thin, arbitrarily-stretched, extruded Prism grid which is allowed to move and deform with the boundary. A single, stationary Cartesian grid is used to mesh the regions away from boundaries. Two different methods of combining the Cartesian and Prism grids while ensuring smoothness across grid interfaces are developed and compared. A second-order-accurate, Navier-Stokes flow-solver using a multi-stage inner iteration procedure for convergence acceleration is implemented. Two computations that demonstrate the capabilities of the technique are presented.

Introduction

A wide variety of structured- and unstructured-grid techniques have been developed [1-4] for reliable and accurate moving-boundary computations. However, most of these still impose significant restrictions either on the complexity of the geometries [2,3] or on the complexity of the motions and deformations that can be efficiently handled [2-4]. Allowable topologic transformations remain particularly limited. The aim of this work is to develop a fully-automated technique that minimizes these restrictions as far as possible. The technique is based on combining the advantages of unstructured, solution-adaptive Prism and Cartesian grids into a composite grid, and on using an efficient, general-purpose, Arbitrary-Lagrangian-Eulerian Navier-Stokes flow solver. Moving, deforming, extruded Prism grids [5-7] are exploited for their generality, robustness, and automatability in handling complex geometries and complex motions (including topologic transformations) near boundaries. A single, stationary, Cartesian grid [8-10] is exploited for its generality, robustness, and automatability for volume meshing away from boundaries. Since stretched Prism grids are used, the technique is applicable to both viscous and inviscid flows.

Grid Generation and Adaptation

The background Cartesian grid is generated by recursive, Quadtree-based subdivision of a single Root cell that is initially defined so that it encloses the entire solution domain and all boundaries in the problem [10-11].

Starting from a geometric definition of each boundary as a Composite Parametric Cubic Spline, the Prismatic grid generation algorithm first divides the spline segments to match a user-specified length-scale. Outward-pointing normals are extruded from these division points by a user-specified distance, and any crossings in the normals are eliminated by smoothing. The extruded points are then connected to form a "ring" of quadrilaterals surrounding the boundary. Each of these quadrilaterals is then treated as the Root cell of a Quadtree-based grid, which is then recursively divided to meet user-specified criteria for cell dimensions, stretchings, and curvature refinement. The Quadtrees are then assembled into a "forest" defining the Prism grid. Figure 1 shows an example Prism grid generated in this manner.

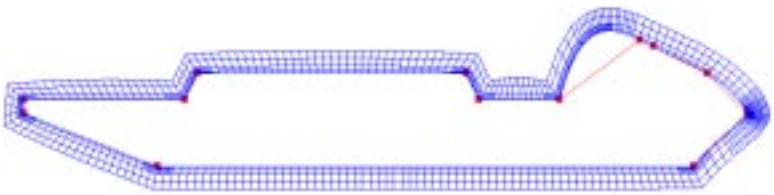


Figure 1. Prism Grid for an Amphibious Assault Vehicle Geometry, Showing the Boundary Alignment of the Cells, and the Non-Isotropic Refinement (with Stretching Toward the Boundary and in High Curvature Regions).

Once the Prism and Cartesian grids have been independently generated, the next step is to create the composite grid. Two options for this have been explored in this work: (i) Cell-Cutting; and (ii) Hole-Cutting. In the first approach, all Cartesian cells or segments of cells that overlap Prism cells are eliminated. This involves "cutting" the Cartesian cells that are intersected to form arbitrary (possibly non-convex) polyhedra. The flow-solver treats all cells as arbitrary polyhedra in this option. In the second option, all Cartesian cells that overlap the outermost layer or two of Prism cells are marked as "interpolation cells" and all remaining *overlapped* Cartesian cells are then eliminated, creating a "hole". The flow-solver recognizes only quadrilateral cells here and uses the overlap layers to interpolate the solution from one grid type to the other. Figures 2(a) and 2(b) show examples of the two alternatives.

Since the two grid types are generated independently, their resolutions at grid interfaces must be subsequently matched to preserve the solution accuracy there.

This is achieved by refining the cells of whichever is the coarser grid in the interface region, as demonstrated in Figures 2(a) and 2(b), in both of which the Cartesian grid is initially at the uniform refinement level indicated by the largest cells shown.

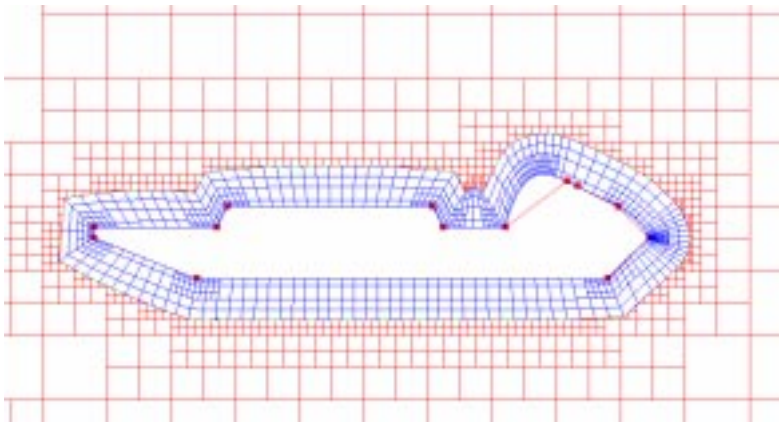


Figure 2(a). Formation of Conservative Hybrid Grids: The Conservative Treatment is Based on Cutting the Cartesian Cells and Eliminating All Overlapped Cartesian Cells and Cell Fragments From the Hybrid Grid

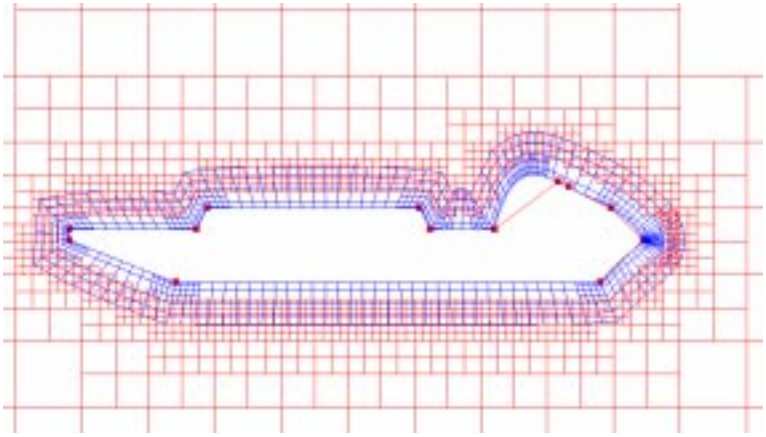
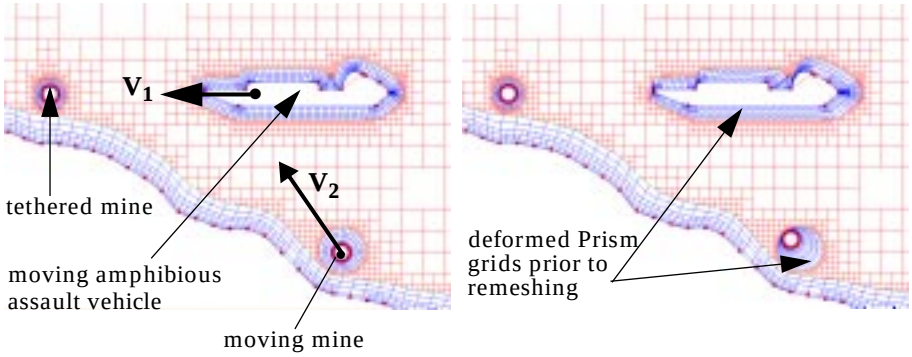


Figure 2(b). Formation of Interpolation-Based Hybrid Grids: The Interpolation-Based Treatment Retains a Sufficiently Thick Overlap Layer Between the Cartesian and Prism Cells for Use in Solution Interpolation

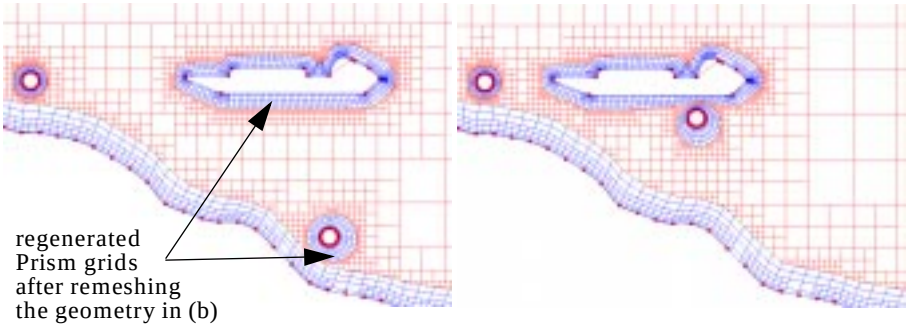
During boundary motion, the innermost layer of edges of the Prism grids remains attached to the boundaries regardless of the motion or deformation of the boundaries, while the outermost layer of edges remains fixed. The deformation of the leaf cells in each Root cell of each Prism Quadtree-forest is computed by transfinite interpolation

on the geometry of the deformed Root quadrilateral. Figures 3(a) and (b) demonstrate the deformation and motion of Prism grids during rigid body motion.



(a) The Original Prism Grids Before Commencement of Motion

(b) The Deformed Prism Grids After Thirteen Motion Steps



(c) The Regenerated Prism Grids After the Thirteenth Motion Step

(d) The Composite Grids After the Tenth Remeshing Cycle

Figure 3. The Motion, Deformation, and Remeshing of Prism Grids

If the deformation of any Prism grid during motion violates one of several pre-specified "quality" criteria (which include minimum and maximum elongation and shear-distortion ratios), that Prism grid is regenerated from scratch and the solution on the new Prism grid is interpolated from that on the old composite grid. Figures 3(b) and (c) show the Prism grids immediately before and immediately after remeshing. Figure 3(d) shows the composite grid after several remeshing cycles. The background Cartesian grid remains fixed regardless of the motion of boundaries.

Solution-based adaptation uses appropriately-scaled sensors of the curl and divergence of the velocity field, following [12], namely:

$$\tau_{ci} = |\nabla \times \vec{v}| d_i^{3/2} \quad \text{and} \quad \tau_{di} = |\nabla \cdot \vec{v}| d_i^{3/2} \ .$$

Additional scaling is applied to account for stretching of cells in an arbitrary direction, and special procedures are required to maintain grid smoothness across the interfaces between the Cartesian and Prism grids during adaptation. During remeshing, the spatial resolution on the new grid must everywhere be forced to at least equal that on the old grid to prevent loss of the original solution adaptation.

Flow Solver

The system of equations solved is a Finite-Volume discretization of the Navier-Stokes equations. For each cell in the domain, the discrete equation is given by

$$\frac{d\overline{Q}V}{dt} + \sum_f (F - \overline{Q}(\vec{\upsilon}_g \cdot \vec{n}))_f dS_f = \sum_f F_{\upsilon,f} dS_f \quad , \tag{1}$$

where υ_g is the velocity of face f, F_f and $F_{\upsilon,f}$ are the inviscid and viscous fluxes on face f, respectively, and the other terms have their usual meanings. A linear Least-Squares reconstruction scheme is used to obtain second-order spatial accuracy, while a Crank-Nicholson time-integration scheme, given by

$$\frac{Q^{n+1}V^{n+1} - Q^n V^n}{\Delta t} + \frac{1}{2}\sum_f (F - Q(\vec{\upsilon}_g \cdot \vec{n}))_f^{n+1} dS_f^{n+1} - \frac{1}{2}\sum_f F_{\upsilon,f}^{n+1} dS_f^{n+1}$$

$$= -\frac{1}{2}\sum_f (F - Q(\vec{\upsilon}_g \cdot \vec{n}))_f^{n} dS_f^{n} + \frac{1}{2}\sum_f F_{\upsilon,f}^{n} dS_f^{n} \tag{2}$$

is used to obtain second-order temporal accuracy. In order to enforce the Geometric Conservation Laws [13], it is necessary to compute the face velocity over each time-step from the mid-point value, $\upsilon_g^{n+1/2}$, using an expression of the form

$$\upsilon_g^{n+1/2} = \frac{2\Delta V_f}{\vec{d} \cdot (\overline{dS}^{n+1} + \overline{dS}^{n})\Delta t} \quad , \tag{3}$$

where ΔV_f is the volume swept by face f of the cell, $\vec{d}$ is the displacement vector of the face centroid, Δt is the time-step, and $\overline{dS}^n$ and $\overline{dS}^{n+1}$ are the face edge vectors at the end of time-steps n and $n+1$, respectively.

In order to improve the computational efficiency, an explicit three-stage scheme with local time-stepping is used, namely:

$$Q^{(0)} = Q^n \quad ,$$

$$Q^{(k)} = Q^{(0)} - \alpha_k \frac{\Delta\tau}{V^{n+1}} Res(Q^{(k-1)}) \qquad (\text{k} = 1,2,3) \, ,$$

$$Q^{new} = Q^{(3)} \quad , \tag{4}$$

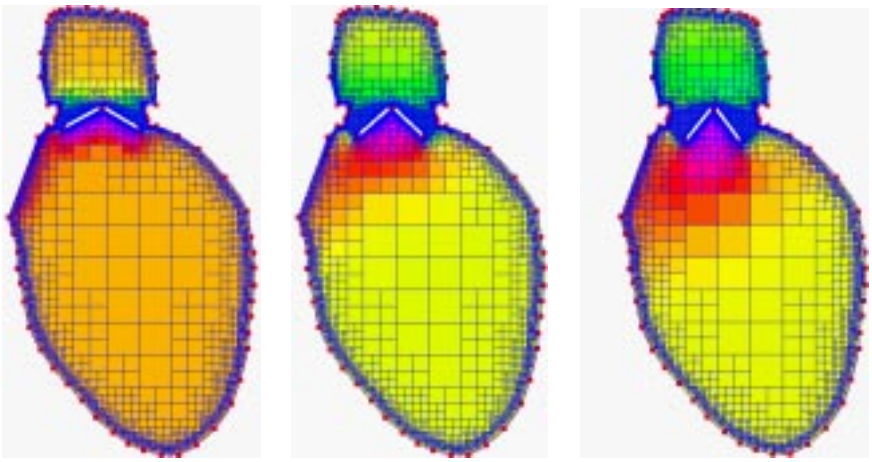
where the α_k values are chosen to appropriately adjust the damping and where the condition $Res(Q^{n+1}) = 0$ is equivalent to Equation (2).

Demonstrative Computations

Validation tests demonstrating agreement with analytical results for ideal compressible flow over a cylinder and for preservation of the free-stream with several configurations of moving and deforming grids were successfully performed. The following two cases are selected to show the general capabilities of the technique.

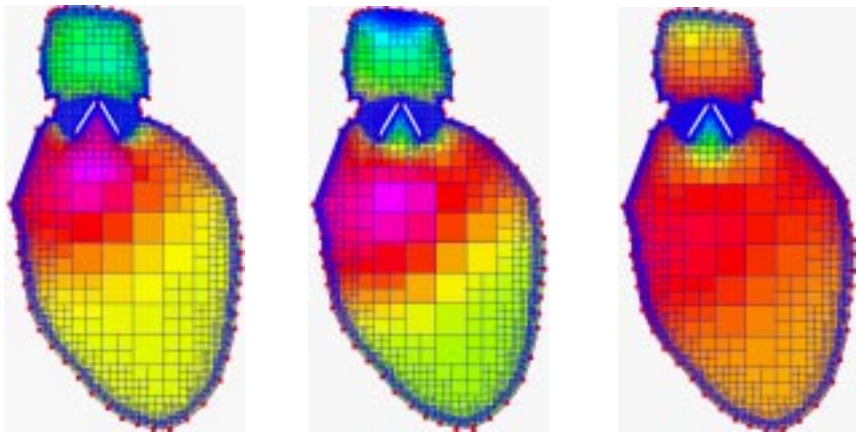
Flow Through Artificial Valves of a Human Heart

Figures 4 (a) - (i) show the valve location, the composite grid, and the pressure distribution in the left ventricle and the left atrium at various times during the complete opening and closing cycle of an artificial heart valve. The valves and valve seats were treated as rigid bodies with prescribed motion. As the figures indicate, the computation was performed using the “hole-cutting” option.

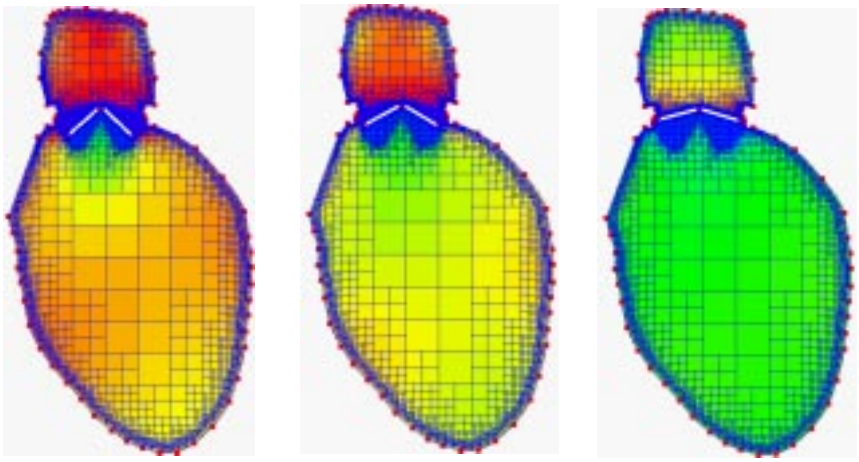


(a) After 0.02 seconds (b) After 0.04 seconds (c) After 0.06 seconds

Figure 4. Grid, Geometries, and Pressure Distributions at Various Times During a Complete Cycle of an Artificial Valve Fitted to a Human Heart



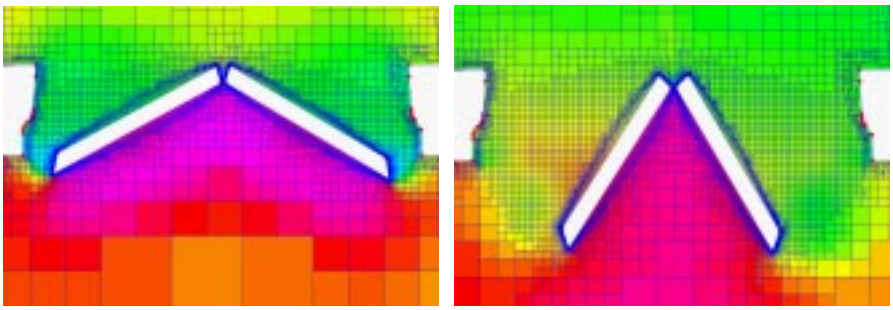
(d) After 0.08 seconds (e) After 0.10 seconds (f) After 0.12 seconds



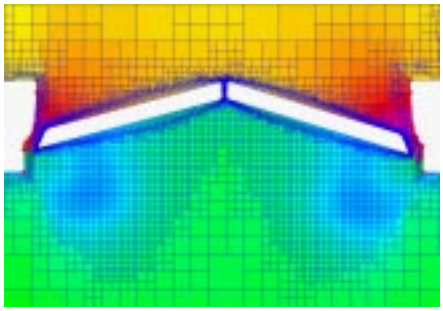
(g) After 0.16 seconds (h) After 0.18 seconds (i)After 0.20 seconds

Figure 4. Grid, Geometries, and Pressure Distributions at Various Times During a Complete Cycle of an Artificial Valve Fitted to a Human Heart *(cont.)*

Figures 5 (a) - (c) show the geometry, grid, and solution near the beginning, mid-point, and the end of the valve cycle. The figures show how the Prism grids move with the valves to which they are attached, and how the Cartesian grid is automatically refined and coarsened as the Prism grids travel across it, and how the grid resolutions remain matched in the overlap zones throughout the motion.



(a) After 0.02 seconds (b) After 0.08 seconds

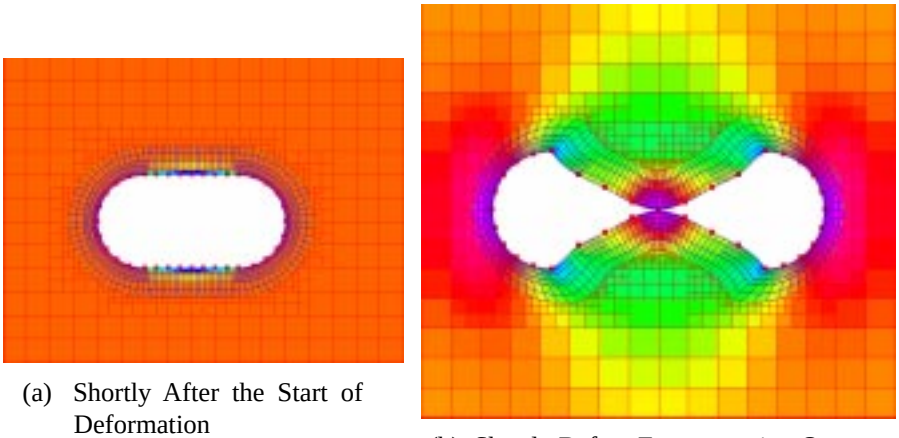


(c) After 0.20 seconds

Figure 5. Zoom Plots of the Valve Region at Three Times During a Complete Cycle of an Artificial Valve Fitted to a Human Heart

Compressible Flow About a Fragmenting Boundary

In this computation, a boundary that is initially smooth and stationary, starts to expand and deform in a quiescent, compressible ideal gas, as shown in figures 6(a) through 6(d). When the thickness of the contraction region reaches a sufficiently small value, the boundary is automatically split into two boundaries which continue to move away from each other. The topologic transformation is handled by automatically redefining the number of closed boundaries in the domain, by redefining the geometry of each new boundary from the original boundary, and by a global remeshing and solution interpolation operation. The same procedure is applied in the case of boundary coalescence. Although the user is still required to specify general geometric criteria for initiation of a topologic transformation, this computation demonstrates the fully-automatic handling of a topologic transformation.



(a) Shortly After the Start of Deformation

(b) Shortly Before Fragmentation Occurs

(c) Shortly After Fragmentation Occurs

(d) Shortly Before Cessation of Motion

Figure 6. The Pressure Field and Grids Around a Deforming, Fragmenting Body

Concluding Remarks

The work presented here is based on the strategy of combining the advantages of moving, deforming Prism grids in the vicinity of boundaries, and the advantages of a stationary Cartesian grid away from boundaries. The results and test-cases attempted demonstrate the capabilities of the two-dimensional version of the technique, and show strong promise for full automation and handling of arbitrary geometries, motions, and topologic transformations. Based on general experience with extending pure-Cartesian and pure-Prism grid generation techniques from 2-D to 3-D, extension of the technique presented here to 3-D appears promising. Comparison of results from the “cell-cutting” and “hole-cutting” approaches shows that while the former is slightly more computationally-intensive, it is also slightly more accurate.

Acknowledgments

The work presented here was funded by the National Science Foundation under Award DMI-9660943, monitored by Dr. G. P. Johnson. The authors are also grateful for the input and guidance received from Prof. K. Powell (The University of Michigan), for the help received from Drs. J. Siegel and V. Makhijani (CFDRC) in defining the geometry and flow conditions for the heart-valve case, and for the guidance and support received throughout this project from Drs. A. Przekwas, and A. Singhal (CFDRC).

References

[1] Baum, J.D., Luo, H., Lohner, R., and Yang, C., "A Coupled Fluid/Structure Modeling of Shock Interaction with a Truck," AIAA Paper 96-0795, 1996.

[2] Benek, J., Buning, P., and Steger, J., "A 3-D Chimera Grid Embedding Technique," AIAA Paper 85-1523-CP, 1985.

[3] Batina, J., "Unsteady Euler Algorithm with Unstructured Dynamic Mesh for Complex Aircraft Aeroelastic Analysis, AIAA Paper 89-1189, 1989.

[4] Trepanier, J.Y., Reggio, M., Paraschivoiu, M., and Cameraro, R., "Unsteady Euler Solutions for Arbitrarily Moving Bodies and Boundaries," AIAA Paper 92-0051, 1992.

[5] Kallinderis, Y. and Ward, S., "Prismatic Grid Generation for Three-Dimensional Complex Geometries," AIAA Journal, Volume 31, No. 10, pp. 1850-1856, October 1993.

[6] Karman, S.L., "SPLITFLOW: A 3D Unstructured Cartesian-Prismatic Grid CFD Code for Complete Geometries," AIAA Paper 95-0343, 1995.

[7] Wang, Z.J., "A Fast Nested Multi-Grid Viscous-Flow Solver for Adaptive Cartesian/Quad Grids," AIAA Paper 96-2091, 1996.

[8] Charlton, E., "An Octree Solution to Conservation Laws Over Arbitrary Regions with Applications to Aircraft Aerodynamics," Ph.D. Thesis, The University of Michigan, 1997.

[9] Melton, J.E., Enomoto, F.Y., and Berger, M.J., "3D Automatic Cartesian Grid Generation for Euler Flows," AIAA Paper 93-3386-CP.

[10] Bayyuk, S.A., Powell, K.G., and van Leer, B., "A Simulation Technique for 2-D Unsteady Inviscid Flows Around Arbitrarily Moving and Deforming Bodies of Arbitrarily Geometry," AIAA Paper 93-3391-CP, 1993.

[11] De Zeeuw, D. and Powell, K., "An Adaptively Refined Cartesian Mesh Solver for the Euler Equations," AIAA Paper 91-1542, 1991.

[12] Palleire, H., Powell, K., and De Zeeuw, D., "A Wave-Model-Based Refinement Criterion for Adaptive-Grid Computations of Compressible Flows," AIAA Paper 92-0322, 1992.

[13] Lesoinne, M. and Farhat, C., "Geometric Conservation Laws for Aeroelastic Computations Using Unstructured Dynamic Meshes," AIAA Paper 95-1709-CP, 1995.