AIAA JOURNAL
Vol. 38, No. 11, November 2000

Intergrid-Boundary Definition Method for Overset Unstructured Grid Approach

Kazuhiro Nakahashi* and Fumiya Togashi†
Tohoku University, Aoba 01, Sendai 980-8579, Japan
and
Dmitri Sharov‡
Science Applications International Corporation, McLean, Virginia 22102

The use of the overset concept for the unstructured grid method is relatively unexplored. However, the overset approach can extend the applicability of the unstructured grid method for real engineering problems without much need for code development. The multiple moving-body problem is one of those applications. Improvement in local resolution for Euler/Navier–Stokes computations on unstructured grids is another use of the overset concept. An efficient and robust algorithm to localize the intergrid boundaries for the overset unstructured grid method is proposed. Simplicity and automation in the intergrid-boundary definition are realized using the wall distance as a basic parameter. The neighbor-to-neighbor jump search algorithm is efficiently utilized in the method. The robustness and efficiency of the search is improved by the use of subsidiary grids that are generated as a byproduct of the Delaunay triangulation method. The basic procedure of the present method is described for a multielement airfoil problem. The effects of the overset method on the solution accuracy and the convergence are tested by ONERA M6-wing. The capability of the method is demonstrated by application to an airplane-rocket booster separation problem.

Introduction

RECENTLY, unstructured grids are becoming popular for complex geometry computational fluid dynamics (CFD) computations because of their advantages over conventional structured grids. The unstructured grids are flexible in that they treat complex geometry and have an adaptive refinement/unrefinement capability; thus, the total number of grid points can be reduced. The computational efficiency of unstructured grid Euler and Navier–Stokes solvers has been significantly improved by recent developments in multigrid methods, implicit time integration methods, and other convergence acceleration methods.[1–3] High-Reynolds-number viscous flows can be also accurately solved by employing an anisotropic stretched tetrahedral mesh or a hybrid grid near solid surfaces.[4–6]

Even with this recent progress, some difficulties still remain in the application of the unstructured grid method to some real engineering problems. One such problem is the unsteady flow computation around multiple bodies in relative motion. The unstructured grid is capable of treating moving bodies as well.[7] However, for computation of unsteady flows with moving bodies, a part of the computational grid or, in some cases, the whole grid, has to be regenerated at every time step. This procedure may become quite complex and computationally expensive.[8] A simple and efficient approach should be developed to treat such a problem.

Another issue in unstructured grid computations is the generation of adequate grids for Navier–Stokes problems. Anisotropic and hybrid grids are used for resolving wall boundary layers and wakes. However, the control of the grid anisotropy is not straightforward for the unstructured grids. Moreover, for some three-dimensional geometries, the viscous grid generation itself becomes difficult. These difficulties in the viscous unstructured grid hinder the application of the unstructured grid methods to the high-Reynolds-number viscous flows. For some problems, use of a subgrid patch in a specific region might resolve these problems.

Received 4 June 1999; presented as Paper 99-3304 at the AIAA 14th Computational Fluid Dynamics Conference, Norfolk, VA, 28 June–1 July 1999; revision received 27 March 2000; accepted for publication 27 March 2000.

*Professor, Department of Aeronautics and Space Engineering; naka@ad.mech.tohoku.ac.jp. Associate Fellow AIAA.
†Graduate Student, Department of Aeronautics and Space Engineering.
‡Researcher, Center for Applied Computational Sciences Advanced Technology Group. Member AIAA.

In this paper, we discuss the application of the overset concept to unstructured grid methods. The overset grid method was introduced by Steger et al.[9] and Benek et al.[10] to provide a powerful means of handling complex geometries by structured grid methods. Currently, the method is used to solve both steady and unsteady Navier–Stokes equations for complex geometry.[11–13] In the overset grid method, valid Chimera holes need to be cut in each grid in regions that overlap with solid bodies or any other non-flow regions which belong to the other grids of the overset grid system. Next, interpolation stencils have to be established for points that lie along the hole boundary. Although several codes have been developed for constructing the intergrid communications among overset structured grids, solving the codes requires significant technical expertise and considerable man-hours.[13,14] One of the reasons for the difficulty of the conventional overset method is that if the geometry of the computational models becomes complex, the number of grids that overlap increases because of the structured grid.

By using unstructured meshes, the number of submeshes required for covering the flow field can be significantly reduced as compared with that needed in the overset structured grid. Using unstructured meshes can also extend the applicability of the unstructured grid method to multiple moving-body problems without much need for code development. However, the overset unstructured grid method is relatively unexplored, and various aspects such as automatic hole cutting, efficient search, interpolations, etc., remain to be developed.

The objective of this paper is to propose an efficient and reliable algorithm to automatically localize the intergrid boundaries for the overset unstructured grid method. The intergrid boundaries are localized using the distance to the wall. The search for donor cells is efficiently performed by the neighbor-to-neighbor search on a modified convex domain utilizing a byproduct of the Delaunay triangulation method. The method is tested for ONERA M6-wing and a separation simulation of a supersonic airplane and a rocket booster.

Intergrid-Boundary Definition

There are two major steps to establish intergrid communications in the overset method:

1) Hole cutting, which involves dividing all points of each subgrid into two groups, active and nonactive. The intergrid-boundary points are identified as the active points next to nonactive points.

2) Identification of interpolation stencils, which involves a search of donor cells for all intergrid-boundary points.

The second step, identification of interpolation stencils, is straightforward for unstructured grids. Once a donor cell is identified, values on the point in this cell are interpolated from values on the vertex of the cell using the area coordinates for a triangle and the volume coordinates for a tetrahedral cell. This treatment of the interface between meshes is nonconservative. It is practical enough for the accuracy of computed large-scale values such as pressure loading on surfaces.[15] However, a conservative treatment of the interface should be developed when the oversetting is used near critical regions such as shear layers for the Navier–Stokes computations.

In the present approach, the donor cell for the interpolation at each intergrid-boundary point is identified during the process of hole cutting. Therefore, the first step is discussed here.

Automatic Hole Cutting

The identification of the intergrid boundary must be performed completely automatically if unstructured grids are used for the overset approach. Manual creations or corrections of the hole cutting for the overset unstructured grids are almost impossible because of the unstructured numbering of the node points. The wall distance is used as a parameter to construct the intergrid boundary. In the mesh overlapping region, a node point having a shorter distance to the solid boundary that belongs to the same grid of the node is selected.

The procedure of the intergrid-boundary definition is shown schematically in Fig. 1. Suppose that the dotted lines show a grid (grid A) generated around body A, and the solid lines show a grid (grid B) for body B. Before the hole cutting, the minimum distance of each node point to its body surface is computed. The minimum wall distance of each node to the body surface can be efficiently computed by advancing the computational front starting from the boundary. Then, the hole-cutting procedure is divided into two steps. The first step is to designate all node points as active or nonactive. The second step is to classify all cells into three groups: active, nonactive, and intergrid-boundary.

Let's consider node point i in grid A in Fig. 1a. Suppose we know the donor cell in grid B for this node. In Fig. 1a, the donor cell is indicated by abc. The distance to wall B from position i in cell abc is then evaluated by a linear interpolation from its vertex values. This distance to wall B of the donor cell is compared with the distance to body A of the node i. Since the distance of this node i to wall A is shorter than that of the donor cell to wall B, we select this node as an active node. In contrast, node j in Fig. 1a will be selected as a nonactive node. This assignment procedure is repeated for all node points in both grids. In Fig. 1b, nodes shown by circles are active nodes in grid A, and those shown by squares are active nodes in grid B. The remaining node points are nonactive.

By designating all node points in the grids as active or nonactive, the next step is to classify all cells into three groups: active, nonactive and intergrid boundary. An active cell is a cell whose vertex nodes are all active, whereas a nonactive cell is the one whose vertexes are all nonactive nodes. The remaining cells are the intergrid-boundary cells that construct the overlapping layers among subgrids for intergrid communications. Figure 1c shows the grids after the removal of the nonactive cells.

The mentioned procedure is very simple, yet it automatically defines both the intergrid boundary and the overlapping layer between grids. The overlapping layer usually has a width of one or two cells. An example of hole cutting for a multielement airfoil is shown in Fig. 2. As shown in Fig. 2a, two unstructured meshes cover the main (grid A) and subairfoils (grid B), respectively. Figure 2b is the grids after holes have been cut, showing the created overlapping layers between the two grids. The present method was tested for overset three-dimensional grids consisting of a total of four million tetrahedral cells; overlapping layers were successfully created.

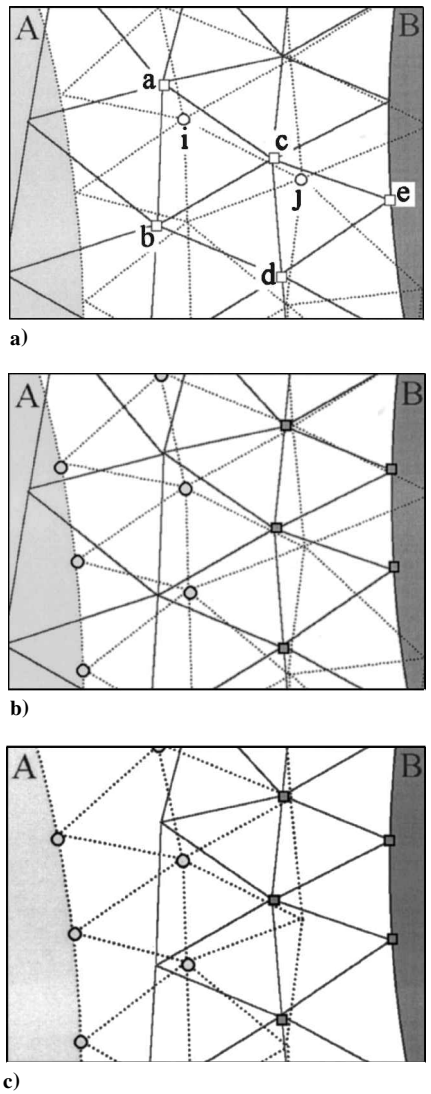


Fig. 1 Determination of intergrid-boundary: a) grid A (. . . .) and grid B (——) are overlapped; b) after the node identification; circles are the active nodes in the grid A, and squares are the ones in grid B; c) after removal of nonactive cells.

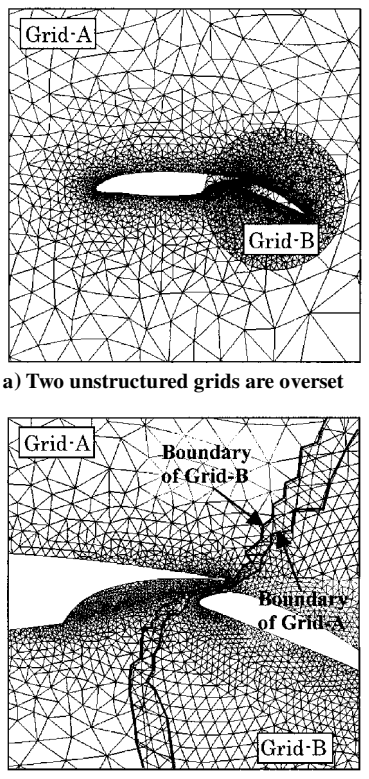


Fig. 2 Determination of intergrid boundary.

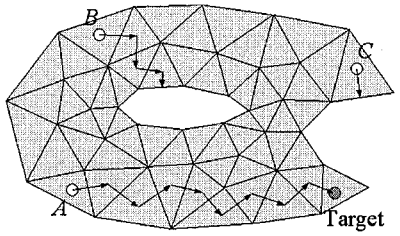


Fig. 3 Neighbor-to-neighbor search. The searches from B and C will fail.

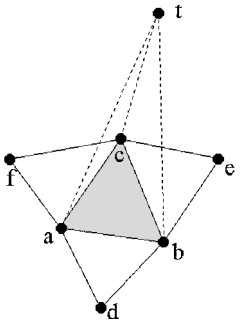


Fig. 4 Selection of the next cell among three neighbors.

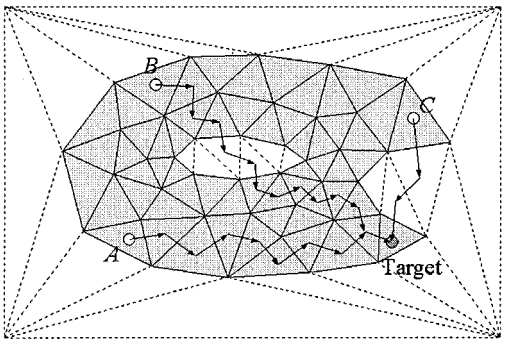


Fig. 5 Neighbor-to-neighbor search in a convex domain.

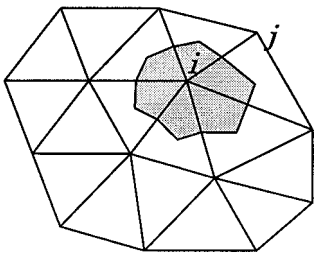


Fig. 6 Control volume of a nonoverlapping dual cell.

Neighbor-to-Neighbor Search

The use of the wall distance for the automatic definition of the intergrid boundary is simple and very reliable. However, all node points must find their donor cells in the overset meshes. The number of searches easily surpasses one million for three-dimensional problems. Therefore, an efficient and reliable search algorithm must be developed.

The neighbor-to-neighbor jump search algorithm[16,17] is efficiently utilized in the present method. The procedure is schematically shown in Fig. 3 for a triangular grid. Starting from an initial guess, cell A in Fig. 3, for example, the method is to repeat a jump to the neighboring cell that is located on the target side of the current cell. In Fig. 4, suppose that we want to find a cell that includes a target point t and that the triangle abc is the current position for commencement of the search. The selection of the next cell from among three neighbors is determined by comparing the directional areas of triangles, bat, cbt, and act. The directional area is the area of the triangle computed by the right-hand rule. For a triangle abc, it is computed by

$$A_{\text{abc}} = \frac{1}{2}\begin{vmatrix} 1 & x_a & y_a \\ 1 & x_b & y_b \\ 1 & x_c & y_c \end{vmatrix} \tag{1}$$

Therefore, the directional area of bat for the case of Fig. 4 becomes negative. With these values, a neighbor whose base edge has the largest positive directional area is selected. For the case shown in Fig. 4, the triangle cbt has the largest positive value, so the search moves to the triangle cbe. If all areas become negative, it means that the current cell encloses the target point, and the search is terminated. Sometimes the search moves back and forth between two neighboring cells. This is the case when the target is located on the edge and the back-and-forth movement is caused by truncation error. To avoid the infinity loop of this situation, the search is terminated if the moved cell is the one that has been previously selected.

This search is very efficient because the search path is one dimensional even in a three-dimensional field. However, the search easily fails, depending on the starting point. As shown in Fig. 3, a search starting from point A succeeds in reaching the target. Searches from B and C, however, get stuck at the body boundary or the outer boundary. For these cases, the search has to be restarted by changing the cell from which the search is commenced.

To avoid such uncertainty of the search, the search domain is modified to be a convex hexahedron for any computational geometry. This modification is done so as to add subsidiary grids into the bodies and outside of the computational region as shown in Fig. 5. If we use the Delaunay triangulation for the grid generation, the subsidiary grids can be obtained automatically as a byproduct of the grid generation procedure. By utilizing the subsidiary grids, the neighbor-to-neighbor search becomes reliable and more efficient.

For multiple moving-body problems, the intergrid-boundary definition must be as efficient as possible. Since the efficiency of the neighbor-to-neighbor search depends on the initial guess, the computational load for the second search after the initial hole cutting becomes significantly smaller. Moreover, once the initial hole cutting has been done, the search of the donor cells after a small relative movement of the subgrid can be limited to nodes around the current intergrid boundaries. For the search, we need information on the neighboring cell numbers. For a tetrahedral grid, this is the pointers from a cell to its face numbers and from its face to its cell numbers on both sides. This information requires about 40 words per node point. The original three-dimensional flow solver for tetrahedral cells uses about 180 words per node, so the neighboring search requires a 20 to 25% increase in memory.

Flow Solver

In the present method, the computational domain consists of several unstructured subgrids, which may overlay each other. The unstructured subgrids generated around each component in the flowfield are put together to discretize the whole computational domain. The Euler equations are solved in each subgrid with the proper boundary conditions.

Solution Algorithm

The Euler equations for compressible inviscid flows are written in an integral form as follows:

$$\frac{\partial}{\partial t}\int_{\Omega} \boldsymbol{Q}\,\mathrm{d}V + \int_{\partial\Omega} \boldsymbol{F}(\boldsymbol{Q})\cdot\boldsymbol{n}\,\mathrm{d}S = 0 \tag{2}$$

where $\boldsymbol{Q} = [\rho, \rho u, \rho v, \rho w, e]^T$ is the vector of conservative variables; ρ the density; u, v, and w the velocity components in the x, y, and z directions; and e the total energy. The vector $\boldsymbol{F(Q)}$ represents the inviscid flux vector, and $\boldsymbol{n}$ is the outward normal of $\partial\Omega$, which is the boundary of the control volume Ω. This system of equations is closed by the perfect gas equation of state.

The equations are solved by a finite volume cell-vertex scheme. The control volume is a nonoverlapping dual cell, which is shown for two dimensions in Fig. 6. For the control volume, Eq. (2) can be written in an algebraic form as follows:

$$\frac{\partial \boldsymbol{Q}_i}{\partial t} = -\frac{1}{V_i}\sum_{j(i)} \Delta S_{ij}\boldsymbol{h}\left(\boldsymbol{Q}_{ij}^{+}, \boldsymbol{Q}_{ij}^{-}, \boldsymbol{n}_{ij}\right) \tag{3}$$

where ΔS_{ij} is the segment area of the control volume boundary associated with the edge connecting points i and j. This segment area ΔS_{ij}, as well as its unit normal $\boldsymbol{n}_{ij}$, can be computed by summing up the contribution from each tetrahedron sharing the edge. The term $\boldsymbol{h}$ is an inviscid numerical flux vector normal to the control volume boundary, and $\boldsymbol{Q}_{ij}^{\pm}$ are values on both sides of the control volume boundary. The subscript of summation *j(i)* means all node points connected to node i.

The numerical flux $\boldsymbol{h}$ is computed using an approximate Riemann solver of Harten–Lax–van Leer–Einfeldt–Wada.[18] The second-order spatial accuracy is realized by a linear reconstruction of the primitive gas dynamic variables $\boldsymbol{q} = [\rho, u, v, w, p]^T$ inside the control volume using the following equation:

$$\boldsymbol{q}(\boldsymbol{r}) = \boldsymbol{q}_i + \psi_i \nabla \boldsymbol{q}_i \cdot (\boldsymbol{r} - \boldsymbol{r}_i) \qquad (0 \le \psi \le 1) \tag{4}$$

where $\boldsymbol{r}$ is a vector pointing to point (x, y, z) and i is the node number. The gradients associated with the control volume centroids are volume-averaged gradients computed by the surrounding grid cells. Venkatakrishnan's limiter[19] is used for the function ψ_i in Eq. (4) because of its superior convergence properties.

The LU-SGS implicit method[2] is applied to integrate Eq. (3) in time. With $\Delta \boldsymbol{Q} = \boldsymbol{Q}^{n+1} - \boldsymbol{Q}^n$ and a linearlization of the numerical flux term as $\boldsymbol{h}_{ij}^{n+1} = \boldsymbol{h}_{ij}^n + \boldsymbol{A}_i^+ \Delta \boldsymbol{Q}_i + \boldsymbol{A}_j^- \Delta \boldsymbol{Q}_j$, a procedure similar to the conventional derivation of the LU-SGS on structured grids[20] leads to the following equations:

$$\left(\frac{V_i}{\Delta t} \boldsymbol{I} + \sum_{j(i)} \Delta S_{ij} \boldsymbol{A}_i^+ \right) \Delta \boldsymbol{Q}_i + \sum_{j(i)} \Delta S_{ij} \boldsymbol{A}_j^- \Delta \boldsymbol{Q}_j = \boldsymbol{R}_i \tag{5}$$

where

$$\boldsymbol{R}_i = -\sum_{j(i)} \Delta S_{ij} \boldsymbol{h}_{ij}^n \tag{6}$$

The LU-SGS method on an unstructured grid can be derived by splitting the node points *j(i)* into two groups, $j \in L(i)$ and $j \in U(i)$, for the second summation in the left-hand side of Eq. (5).

The final form of the LU-SGS method for the unstructured grid becomes the following. Forward sweep:

$$\Delta \boldsymbol{Q}_i^* = \boldsymbol{D}^{-1} \left[\boldsymbol{R}_i - 0.5 \sum_{j \in L(i)} \Delta S_{ij} \left(\Delta \boldsymbol{h}_j^* - \rho_A \Delta \boldsymbol{Q}_j^* \right) \right] \tag{7a}$$

Backward sweep:

$$\Delta \boldsymbol{Q}_i = \Delta \boldsymbol{Q}_i^* - 0.5 \boldsymbol{D}^{-1} \sum_{j \in U(i)} \Delta S_{ij} (\Delta \boldsymbol{h}_j - \rho_A \Delta \boldsymbol{Q}_j) \tag{7b}$$

where $\Delta \boldsymbol{h} = \boldsymbol{h}(\boldsymbol{Q} + \Delta \boldsymbol{Q}) - \boldsymbol{h}(\boldsymbol{Q})$ and $\boldsymbol{D}$ is a diagonal matrix derived by the Jameson–Turkel approximation of Jacobian[21] as $\boldsymbol{A}^{\pm} = 0.5(\boldsymbol{A} \pm \rho_A \boldsymbol{I})$, where ρ_A is a spectral radius of Jacobian $\boldsymbol{A}$:

$$\boldsymbol{D} = \left(\frac{V_j}{\Delta t} + 0.5 \sum_{j(i)} \Delta S_{ij} \rho_A \right) \boldsymbol{I} \tag{8}$$

The lower/upper splitting of Eq. (7) for the unstructured grid is realized by using a grid reordering technique[2] to improve the convergence and the vectorization.

Overset Implementation

The flow solver must be modified to account for the use of multiple meshes. In addition to the boundaries of the computational domain, subgrids have intergrid boundaries with the neighboring donor subgrids and may also contain holes. The nonactive cells must be excluded or blanked from the flowfield solution.

All node points can be identified as to whether they belong to the active or nonactive cells:

$$IBLANK = \begin{cases} 1, \text{ if a point in not blanked} \\ 0, \text{ if a point is blanked} \end{cases} \tag{9}$$

This value is 1 or 0, depending on the area inside or outside the computational subregion. In the flow solver, the right-hand side vector $\boldsymbol{R}_i$ in Eq. (7) is multiplied by the value ***IBLANK(i)***. The $\Delta \boldsymbol{Q}$ in the outside region (hole region) is set to zero.

Results and Discussion

ONERA M6-Wing

The effects of the present overset grid method on solution accuracy and convergence were tested using ONERA M6-wing.[22] At first, a single grid covering the entire flowfield between the wing surface and the outer boundary was generated (grid A in Fig. 7). The outer boundary was a hemisphere whose radius was twenty times the root chord length of the wing. The number of node points of this hemispherical grid was 244,285, and the number of tetrahedral cells was 1,268,582.

For a test of the overset grid, a box-shaped subregion covering the near field of the wing was defined as shown in grid B of Fig. 7, and a relatively fine tetrahedral grid was generated in it. The number of node points of the box subgrid was 268,094, and the number of cells was 1,418,277. The surface grid on the wing was the same as that used for the hemispherical grid. An overset grid was constructed using the hemispherical and box grids. The intergrid boundaries between these grids were identified using the distances from the wing surface for the box grid and from the outer-boundary for the hemispherical grid. With this approach, the cells in the box grid are preserved when the near-field cells in the hemispherical grid are automatically blanked for computations. The number of cells after the hole cutting of the hemispherical grid was 163,287.

The computations were performed using both the single hemispherical grid and the overset grid at the freestream Mach number of 0.84 and the angle of attack of 3.06. The computed pressure contours are shown in Fig. 8. The figure also shows the intergrid boundary on the symmetrical plane where smooth transitions of the contour lines through the boundary can be observed. Figure 9 shows cut views of the grids and computed pressure contours at

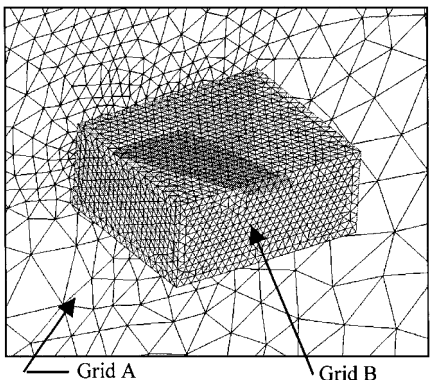


Fig. 7 Overset unstructured grids for ONERA M6-wing. A fine grid (grid B) covering the wing region is embedded on the entire grid (grid A).

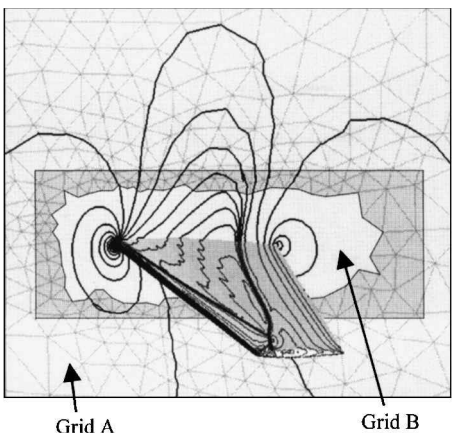
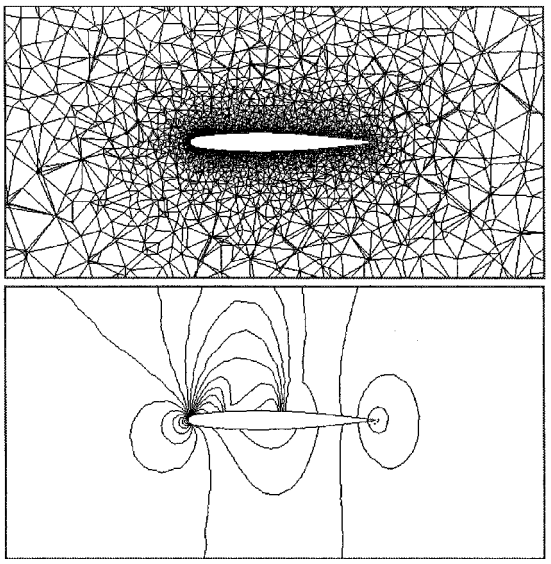
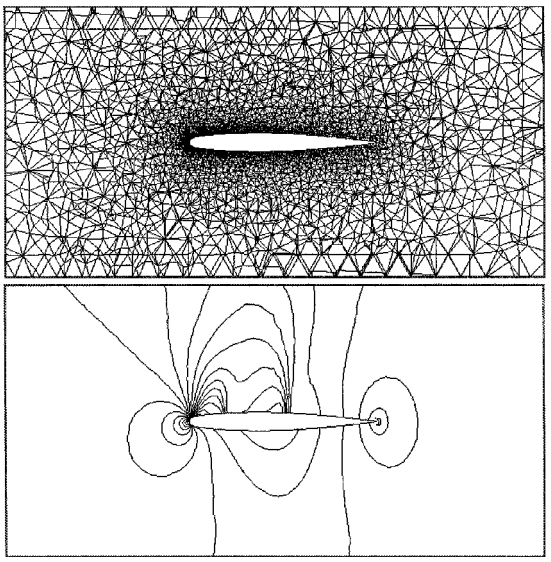


Fig. 8 Computed pressure contours on the wing and symmetrical plane for ONERA M6-wing at $M_\infty = 0.84$ and $\alpha = 3.06$ deg.



a) Single grid



b) Overset grid

Fig. 9 Cross-sectional views of grid and pressure countours at semispan locations of 65%, $M_\infty = 0.84$ and $\alpha = 3.06$ deg.

a 65% semispan location for both single and overset grids. Both grids have the same wing-surface grid, but the overset grid has a much higher density in the near field, as shown in the figure. The computed pressure contours reflect the difference of grid densities between the two grids. The shock waves on the upper surface are sharply captured by the overset grid.

Figure 10 shows the convergence histories of both single and overset grids solutions. The Courant-Friedrichs–Lewy (CFL) number of 10,000 was used for both cases from the beginning of the computations. Both solutions have about the same convergence rate indicating that the efficiency of the flow solver is not degraded by the current overset strategy and the treatment of intergrid boundaries.

Figure 11 shows comparisons of the Cp distributions on the wing at several semispan locations between the experiment,[22] single grid, and overset grid computations. The results show that both single and overset solutions are very close to each other, in good agreement with the experimental data. A small discrepancy between the single and overset grids can be observed in the results at the 44 and 65% semispan locations. The position of the second shock wave in the overset result shifted downward. However, the locations of the first shock wave in the two computations agree well with each other. Therefore, the discrepancy of both single and overset grid results at these semispan locations seems to be due to the difference in

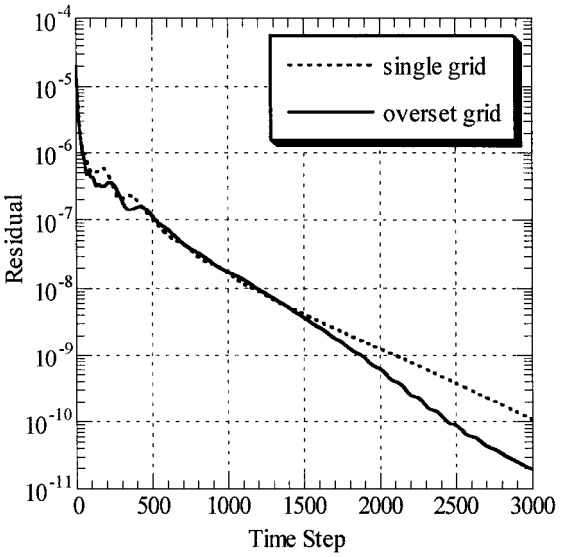


Fig. 10 Convergence histories for ONERA M6-wing at $M_\infty = 0.84$ and $\alpha = 3.06$ deg.

the grid resolutions. Although both single and overset grids use the same surface grid, the overset grid uses a much finer spatial grid for the near field of the wing than that of the single grid.

Supersonic Airplane and Booster Separation

To evaluate the capability of the overset unstructured grid method for a multiple moving-body problem, the method was applied to the numerical simulation of an experimental supersonic airplane's separation from a rocket booster, as shown in Fig. 12. The National Aerospace Laboratory (NAL) of Japan is currently working on a project to develop experimental supersonic airplanes[23] as a basic study for the next generation of supersonic transport. The first model of the experimental airplane is unpowered, and a solid rocket booster will be used to launch it to a high altitude at a speed of about Mach 2.5.

For the configuration shown in Fig. 12, two unstructured grids, each of which covers the airplane and the rocket booster, respectively, were generated as shown in Fig. 13. The outer cylindrical grid was generated for the airplane and the inner cylindrical grid for the rocket booster. The number of node points and cells of these grids are shown in Table 1. For a simulation of the airplane-booster separation, the inner grid moves with the rocket booster in the stationary outer cylindrical grid.

Figure 14 shows the intergrid boundaries of the booster grid for several relative locations between the airplane and the booster. The relative positions and angles of attack were prescribed. In the figure, the dark rough surfaces are the cut surfaces (intergrid boundaries) of the booster grid due to the existence of the airplane. As shown in the figure, the intergrid boundary becomes complex because of the large fins attatched to the end of the rocket booster. Figure 15 shows the grids on a cut plane in the tail-fin region perpendicular to the axis of the airplane fuselage. The gray region is a cross-sectional view of the booster subgrid, and the solid lines show a cross-sectional view of the airplane mesh.

Table 1 shows the average number of cell jumps required for one search path measured for the configuration of zero relative angle of attack. The search from the rocket booster to the airplane is to find a donor cell from among 3.3 million cells. Even with this large number of cells, a search path required an average of only 102 cell jumps. The initial guess for each search used the previous search result. Therefore, the search path length may be further reduced if the node numbering is well ordered. The central procesing unit (CPU) cost of the search for all node points in both directions (approximately 0.8 million points) was about 10 min on a Dec Alpha personal computer (21164/500 MHz). After the initial intergrid definition, the CPU cost for the next search became a small fraction of the initial search owing to a good guess of the starting point. On a NEC-SX-4 with a single processor, the CPU time for the initial search was 910, and for the

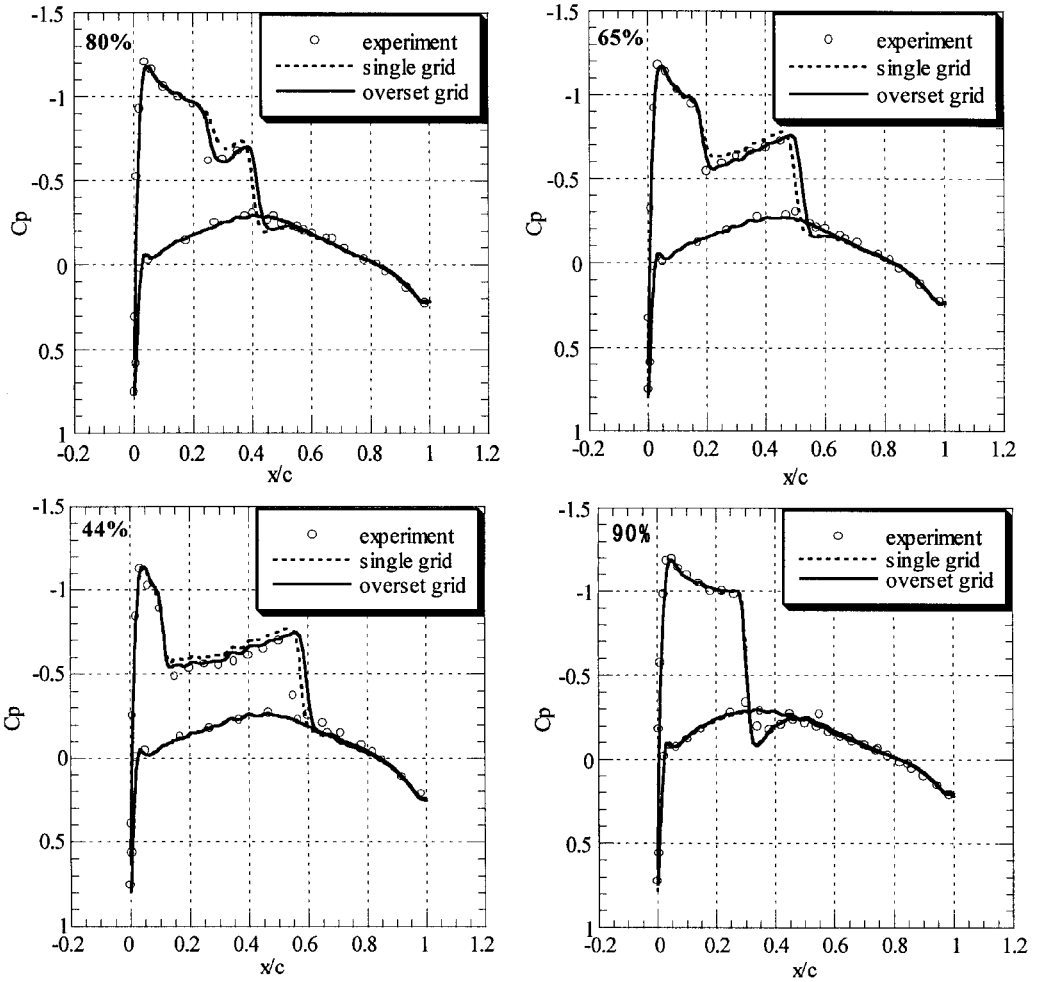


Fig. 11 Comparison of *Cp* distributions for the ONERA M6-wing at semispan locations of 44%, 65%, 80%, and 90%, $M_\infty = 0.84$ and $\alpha = 3.06$ deg.

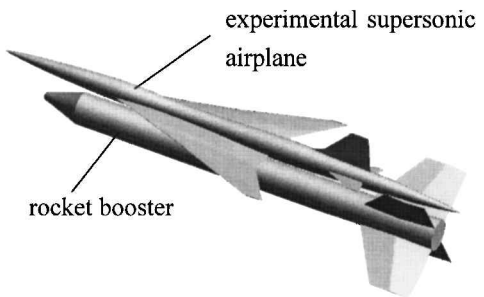


Fig. 12 National Aerospace Laboratory's experimental supersonic airplane with rocket booster for launch.

following one was 18 s. The computational time of the flow solver per step for this case was about 40 s.

Computations were performed at a Mach number of 2.5 under an assumption of quasi-steady flow. The angle of attack of the airplane was fixed at zero, and the angles of attack of the rocket booster relative to the airplane were specified at 0, −3, and −5 deg. Figure 16 shows the computed pressure contours around the airplane and booster. Shock waves generated at the noses of the airplane and the booster create a complex reflection pattern in the narrow region between the bodies. At the beginning of the booster separation, the shock wave from the booster nose hits the forward part of the lower surface of the airplane wing. This initially causes an increase in the pitching moment of the airplane. This pitching moment then decreases to a negative value as the impinging point of the booster-nose shock on the airplane wing moves downward.

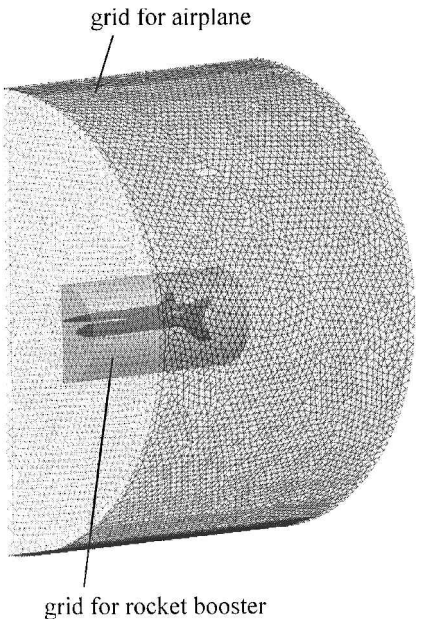


Fig. 13 Overset grids for the supersonic airplane (outer cylindrical region) and rocket booster (inner cylindrical region).

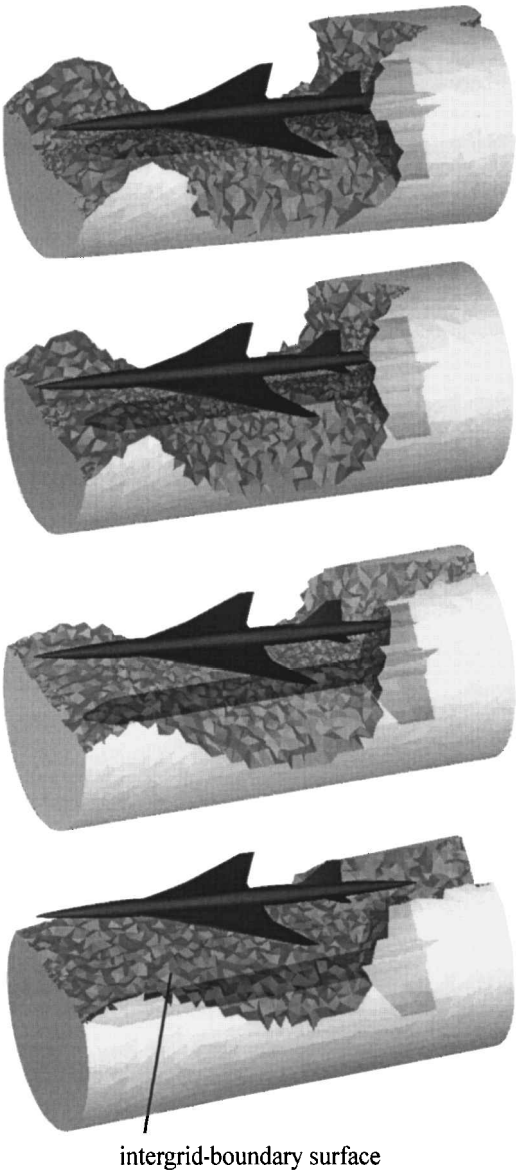


Fig. 14 Intergrid boundaries of the booster subgrid for relative angles of attack of 0, − 3, − 5, and − 7 deg.

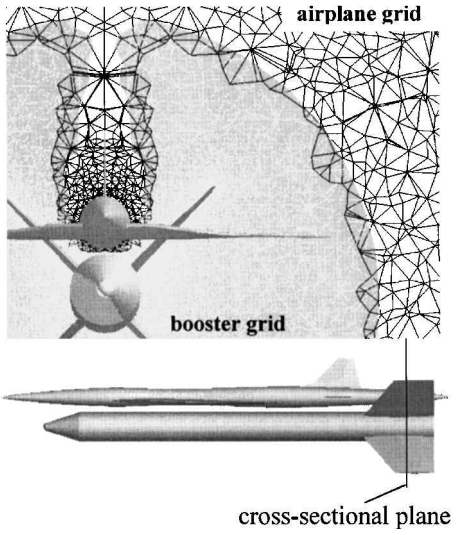


Fig. 15 Cross-sectional view of the overset grids in the tail-fin region for zero relative angle of attack.

Table 1 Grid data and average number of cell jumps per search

Parameter	Value
Airplane grid	599,203 node points, 3,268,529 cells
Booster grid	184,409 node points, 968,679 cells
Donor cell search from airplane to booster	68 cells/node
Donor cell search from booster to airplane	102 cells/node

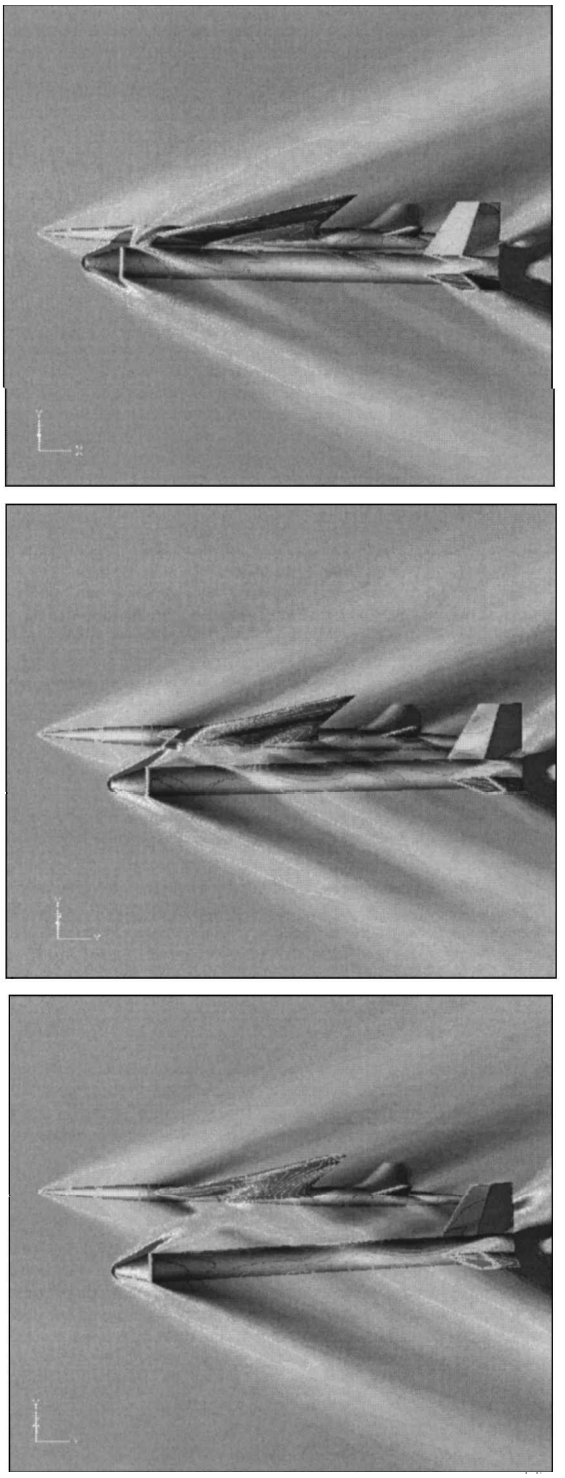


Fig. 16 Pressure contours on the surface and symmetrical plane of supersonic airplane–rocket booster separation simulation at freestream Mach number of 2.5 and relative angles of attack of booster at 0,− 3, and − 5 deg.

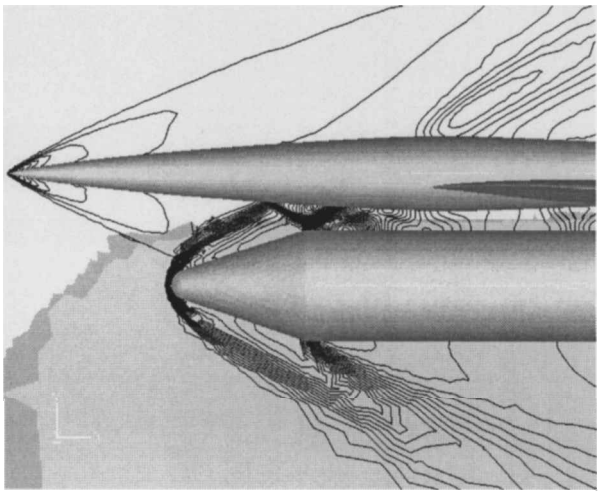


Fig. 17 Pressure contours on the symmetrical plane of a supersonic airplane–rocket booster separation simulation at freestream Mach number of 2.5 and relative angle of attack of booster at 0 deg.

Figure 17 shows an enlarged view of the booster-nose region. A strong shock wave generated at the booster nose crosses the intergrid boundary and hits the undersurface of the airplane fuselage. The reflected shock wave again crosses the grid boundary and hits the booster surface.

Conclusion

An efficient and robust algorithm to localize the intergrid boundaries for the overset unstructured grid method has been proposed. Simplicity and automation of hole cutting are realized using the wall distance as a basic parameter. The use of subsidiary grids, which are generated as a byproduct of the Delaunay triangulation method, makes the neighbor-to-neighbor jump search reliable and efficient. For an airplane-rocket separation simulation, one search path in 3.3 million cells consisted of about 100 cell jumps, on the average. With a better guess of the starting cell, the efficiency of the search will be further improved. The computational test for ONERA M6-wing shows that the accuracy and the convergence efficiency of the flow solver are not degraded with the current overset strategy and the treatment of intergrid boundaries. The capability of the method was demonstrated for a separation simulation of a supersonic experimental airplane and a rocket booster. The use of the overset concept with the unstructured grid methods holds great promise for extending the applicability of the unstructured grid method for real engineering problems without much need for code development.

Acknowledgments

The authors wish to thank T. Iwamiya of the National Aerospace Laboratory for providing us with the airplane and rocket booster geometry data used for the wind tunnel experiment. The authors also wish to thank Y. Ito, graduate student of Tohoku University, for his help in generating the grids.

References

[1] Mavriplis, D. J., "On Convergence Acceleration Techniques for Unstructured Meshes," AIAA Paper 98-2966, June 1998.

[2] Sharov, D., and Nakahashi, K., "Reordering of Hybrid Unstructured Grids for Lower-Upper Symmetric Gauss–Seidel Computations," *AIAA Journal*, Vol. 36, No. 3, 1998, pp. 484–486.

[3] Luo, H., Baum, J., and Löhner, R., "A Fast, Matrix-free Implicit Method for Compressible Flows on Unstructured Grids," *Journal of Computational Physics*, Vol. 146, 1998, pp. 664–690.

[4] Sharov, D., and Nakahashi, K., "Hybrid Prismatic/Tetrahedral Grid Generation for Viscous Flow Applications," *AIAA Journal*, Vol. 36, No. 2, 1998, pp. 157–162.

[5] Löhner, R., "Generation of Unstructured Grids Suitable for RANS Calculations," AIAA Paper 99-0662, Jan. 1999.

[6] Khawaja, A., and Kallinderis, A., "Adaptive Hybrid Grid Generation for Turbomachinery and Aerospace Applications," AIAA Paper 99-0916, Jan. 1999.

[7] Löhner, R., and Baum, J. D., "Three-Dimensional Store Separation Using a Finite Element Solver and Adaptive Remeshing," AIAA Paper 91-0602, Jan 1991.

[8] Morgan, K., Perarie, J., and Perio, J., "Unstructured Grid Methods for Compressible Flows," AGARD Rept. 787, 1992.

[9] Steger, J. L., Dougherty, F. C., and Benek, J. A., "A Chimera Grid Scheme," *Advances in Grid Generation*, edited by K. N. Ghia and U. Ghia, FED-Vol. 5, American Society of Mechanical Engineers, New York, 1983.

[10] Benek, J. A., Buning, P. G., Steger, J. L., "A 3-D Chimera Grid Embedding Technique," AIAA Paper 85-1523, June 1985.

[11] Merkin, R. L., "On Adaptive Refinement and Overset Structured Grids," AIAA Paper 97-1858, 1997.

[12] Cao, H. V., and Su, T. Y., "Navier–Stokes Analyses of a 747 High Lift Configuration," AIAA Paper 98-2623, June 1998.

[13] Rogers, S. E., Cao, H. V., and Su, T. Y., "Grid Generation for Complex High-Lift Configurations," AIAA Paper 98-3011, June 1998.

[14] Wang, Z. J., Parthasarathy, V., and Hariharan, N., "A Fully Automated Chimera Methodology for Multiple Moving Body Problems," AIAA Paper 98-0217, Jan. 1998.

[15] Hariharan, N., Wang, Z. J., and Buning, P., "Application of Conservative Chimera Methodology in Finite Difference Settings," AIAA Paper 97-0627, Jan. 1997.

[16] Mavriplis, D. J., "Three-Dimensional Unstructured Multigrid for the Euler Equations," AIAA Paper 91-1459, June 1991, pp. 239–247.

[17] Löhner, R., "Robust, Vectorized Search Algorithms for Interpolation on Unstructured Grids," *Journal of Computational Physics*, Vol. 118, 1995, pp. 380–387.

[18] Obayashi, S., and Guruswamy, G. P., "Convergence Acceleration of an Aeroelastic Navier–Stokes Solver," AIAA Paper 94-2268, June 1994.

[19] Venkatakrishnan, V., "On the Accuracy of Limiters and Convergence to Steady-State Solutions," AIAA Paper 93-0880, Jan. 1993.

[20] Jameson, A., and Yoon, S., "Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations," *AIAA Journal*, Vol. 25, No. 7, 1987, pp. 929–935.

[21] Jameson, A., and Turkel, E., "Implicit Schemes and LU Decompositions," *Mathematics of Computation*, Vol. 37, No. 156, 1981, pp. 385–397.

[22] Schmitt, V., and Charpin, F., "Pressure Distributions on the ONERA M6-Wing at Transonic Mach Numbers," *Experimental Data Base for Computer Program Assessment*, AR-138, AGARD, 1979.

[23] Iwamiya, T., "NAL SST Project and Aerodynamic Design of Experimental Aircraft," *Proceedings Computational Fluid Dynamics*, '98, Vol. 2, ECCOMAS 98, Wiley, New York, 1998, pp. 580–585.

P. Givi
Associate Editor