JOURNAL OF AIRCRAFT
Vol. 36, No. 6, November–December 1999

# Development of a Fully Systemized Chimera Methodology for Steady/Unsteady Problems

Kum Won Cho* and Jang Hyuk Kwon†
*Korea Advanced Institute of Science and Technology, Taejon 305-701, Republic of Korea*
and
Seungsoo Lee‡
*Agency for Defence Development, Taejon 305-600, Republic of Korea*

**A new systemized procedure for chimera-domain decomposition is presented. This procedure consists of a new cut–paste algorithm for optimal mesh interface and a two-step search method for donor cell identification. It is fully automated and requires minimal user input. The cut–paste algorithm is based on the advancing front technique in which the fronts are iteratively determined from the initial fronts, which are a collection of fringe points obtained from conventional Chimera hole cutting. The final fronts are determined iteratively in such a way that the overlapping region is minimized. With this method, interpolation points are located away from solid walls where flow gradient is high. We also can reduce the error that may arise from interpolation in stiff gradient regions. Two- and three-dimensional examples are chosen to demonstrate the effectiveness of this new procedure.**

## Nomenclature

| | |
|---|---|
| $C_p$ | = pressure coefficient |
| $C_x$, $C_y$, $C_m$ | = axial, normal, and pitching moment coefficients |
| $\boldsymbol{D}$ | = diagonal matrix, $[\Delta\tau/\Delta t + 1]\boldsymbol{I}$ |
| $\hat{\boldsymbol{F}}$ | = generalized flux vector, $\boldsymbol{iE} + \boldsymbol{jF} + \boldsymbol{kG} - \boldsymbol{Q\xi}$ |
| $\tilde{\boldsymbol{F}}$ | = numerical flux vector |
| $\boldsymbol{I}$ | = identity matrix |
| $I_s$, $J_s$, $K_s$ | = spatial directions |
| $I_y$ | = inertia of the store |
| $i$, $j$, $k$ | = grid indices |
| $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ | = unit normal vectors in $x$, $y$, and $z$ directions |
| $i_c$, $j_c$, $k_c$ | = indices of the Cartesian grid |
| $k_r$ | = blanking index |
| $L$ | = characteristic length of the store |
| $\boldsymbol{L}_i$, $\boldsymbol{L}_j$, $\boldsymbol{L}_k$ | = implicit operators |
| $m$ | = mass |
| $\boldsymbol{n}$ | = outward normal vector |
| $\boldsymbol{Q}$ | = conservative variables |
| $\boldsymbol{R}$ | = residual |
| $t$ | = time |
| $t^*$ | = nondimensional time |
| $U$ | = normal velocity components to $\partial V$ in the moving coordinate, $\boldsymbol{n} \cdot (\boldsymbol{v} - \boldsymbol{\xi})$ |
| $u$, $v$, $w$ | = velocity components in $x$, $y$, and $z$ directions |
| $V$ | = control volume |
| $\boldsymbol{v}$ | = velocity vector |
| $x$, $y$, $z$ | = spatial coordinates |
| $\partial V$ | = surface of the control volume |
| $\theta$ | = pitch angle |
| $\boldsymbol{\xi}$ | = grid velocity |
| $\rho$, $p$, $e$ | = density, pressure, and internal energy |
| $\tau$ | = fictious time |
| $\phi$ | = circumferential angle of the three-dimensional store |

*Subscripts*

| | |
|---|---|
| $p$ | = position of coordinates |
| $t$ | = time |
| $\infty$ | = freestream |

## Introduction

IN the past decade, the chimera grid technique has gained popularity in the field of computational fluid dynamics (CFD). With this grid, one can generate a grid system over complex geometry by generating component grids and overlapping them. Usually, there are two steps in chimera-domain decomposition: removing nondiscretizable points (chimera hole cutting) and finding interpolation information for interpolation points (donor cell identification). Popular chimera-domain decomposition methodologies need user-specified hole-cutting surfaces. In PEGASUS,[1] one must specify the hole-cutting surfaces as well as the hole grids, whereas in DCF3D[2] one has to define the hole-cutting surfaces by combining simple analytical shapes such as cylinders and spheres. These procedures require a high level of expertise and a large number of work hours.

Several attempts have been made to automate hole cutting. Among these, Wey's method[3] is an efficient one that uses the advancing front technique to determine iteratively cutting surfaces. Two fronts are constructed from solid boundaries. The first front acts as a cutter to blank out the grid points that belong to other component grids; the other acts as a fence to protect grid points from being cut by other cutters. The advancing distances are computed from relative distances between fronts and neighboring grid points. The user has to supply a few constants to control advancing distance. Basically, Wey's method is a repetitive application of chimera hole cutting with advancing cutting surfaces.

In this paper, we present a new algorithm, a cut–paste algorithm, based on Wey's[3] idea of improving hole-cutting surfaces through iterations. Instead of applying expensive chimera hole cutting successively, we advance the fringe points (boundary points of hole points) discretely after applying the conventional chimera hole cutting. Unlike Wey's method, the fronts for the cut–paste algorithm are constructed with a set of fringe points. The movement of the fringe points, furthermore, is made by following the grid line, and the advancing distance is set to the length of that grid line. In other words, the advancing is done in a quantum manner. The number of iterations needed is smaller than that needed by Wey's algorithm, and no user inputs are required to control the advance of the fringe points. Like Wey's method, our new chimera decomposition method

Received 19 December 1998; revision received 13 May 1999; accepted for publication 17 May 1999. 

*Doctoral Candidate, Department of Aerospace Engineering, 373-1 Kusong-Dong, Yusong-Gu.
†Associate Professor, Department of Aerospace Engineering, 373-1 Kusong-Dong, Yusong-Gu.
‡Senior Researcher, Yusong P.O. Box 35-3.

consists of three steps: chimera hole cutting, improving hole cutting with the cut–paste algorithm, and donor cell identification.

Solid walls or nonpenetrable boundary conditions are used for the initial hole-cutting surface. The zones of interference scheme[3] and hole-map algorithm[4] are used for the initial chimera hole cutting. The hole-cutting surfaces are approximately modeled with a collection of small cubes that touch the cutting surfaces. It is relatively simple to discover whether a given point lies within a collection of cubes or not.

For donor cell identification, we use a two-step search method. First, a stencil walk[5] is used to find possible donor candidate cells for a given interpolation point. We then search the donor cells in the vicinity of this point using Newton–Raphson iteration over the isoparametric coordinate to check whether the cells actually enclose the interpolation point. This second step is referred to a gradient search in this paper. This two-step search guarantees success in identifying donor cells and reduces the computational cost of the gradient search.

We introduce our new chimera-domain decomposition method based on an iterative cut–paste algorithm. In the next section, we describe the zones of the interference scheme and the hole-map algorithm for the initial chimera hole cutting. A detailed description of the cut–paste algorithm will follow. The donor cell identification procedure and a brief discussion of the Euler solver will then be presented. Demonstration examples of this method are shown in the results section to illustrate the effectiveness of present method.

## Chimera-Domain Decomposition

As already stated, the chimera-domain decomposition method presented here consists of three steps. We will describe each step in the following subsection. The cut–paste algorithm, the second step, is designed to improve the chimera hole cutting with a minimal overlapped region with the interpolation points located away from the solid walls.

The given grid points are first converted to a cell-centered coordinate. This is done to maintain consistency with the current flow solver, which is a cell-centered finite volume code. Cell-centered points are used in the chimera hole cutting, in the cut–paste algorithm, and for donor cell identification. Each boundary cell of the component grid is flagged with boundary conditions. These will be used in the cut–paste algorithm and the donor cell identification so that the two-step search can be done across the multiblock grid system.

### Chimera Hole Cutting

Hole points are those points in a grid component whose dependent variable values will not be updated or defined. This is due to the presence of solid bodies embedded in the mesh. Mathematically, chimera hole cutting is equivalent to solving of the so-called point-in-polyhedron problem. Research has focused on finding an infallible and efficient method at low CPU time cost. Popular methods of doing this are Shimrat's algorithm (vector intersection test, see Ref. 6), the scalar product method,[7] etc. However, these methods require a lot of computation time and are known to fail for complex geometries. In this paper, chimera hole cutting is done in two steps. In the first step, the given hole-cutting surfaces are approximately modeled with a collection of cubes using the zones of interference scheme. This scheme was originally developed to remove hidden lines for isoparametric and perspective projections of three-dimensional objects.[8] It was later modified for the Chimera grid as a preprocessing algorithm for general interference problems.[3] In the second step, every computational cell-center point is tested for whether it lies inside the collection of cubes, using the hole-map algorithm, which has the advantage of being vectorizable.

As in most chimera-domain decomposition methods, the hole-cutting surface should be described first before applying the hole-cutting method implemented in this paper. We use a solid wall boundary as the hole-cutting surface. The only requirement for the present hole-cutting method to work is that every hole-cutting surface be closed. The hole-cutting surface consists of facets that are lines in two dimensions or quadrilaterals in three dimensions. The

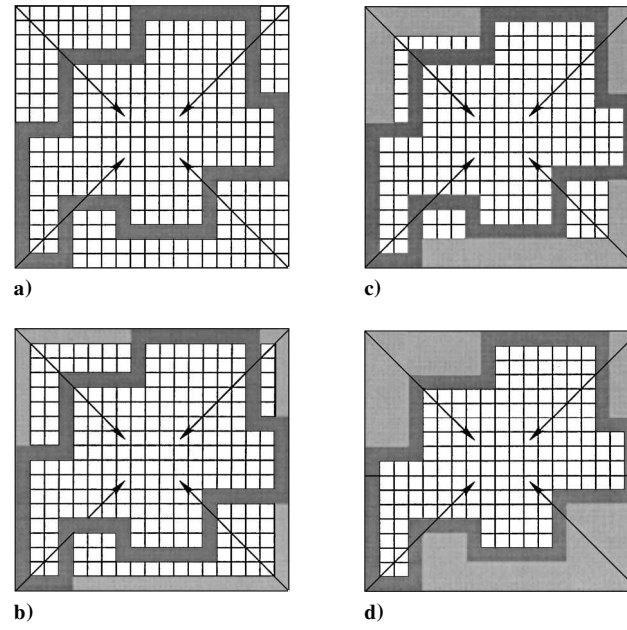


**Fig. 1 Constructing hole region using the Cartesian zone.**

next step is to compute the average length of the facets and the minimum and maximum coordinates of the hole-cutting surface in the $x$, $y$, and $z$ directions. With this, a box is constructed that encloses the hole-cutting surface. Then the box is divided into cubes or Cartesian grids, whose size is the average length of the facets, and zero is assigned to every cell of the Cartesian grid. Finally, we check if any facet of the hole-cutting surface touches a cell and assign a positive value to that cell, otherwise we leave it as it is. If the hole-cutting surface is a closed surface, the collection of the cells with positive value will also form a closed surface (see the dark cells in Fig. 1a). This is the end of the zones of interference scheme.

One can now determine whether a given cell lies inside the hole-cutting surface using the hole-map algorithm. The hole-map algorithm starts from the results of the zones of interference scheme.

1) If the value of the outer cell of the constructed Cartesian grid has a value of zero, it is replaced with a value of $-1$.

2) The cells march from the corners of the constructed Cartesian box into the inner area. For cells having the value of zero, if the value of the touching cell is $-1$, they are replaced with $-1$.

3) Repeat the preceding steps $M$ times, where $M$ is the iteration number that is calculated using

$$M_x = \text{Int}[N_x \times 0.5], \qquad M_y = \text{Int}[N_y \times 0.5]$$

$$M_z = \text{Int}[N_z \times 0.5], \qquad M = \min(M_x, M_y, M_z) \tag{1}$$

where $N_x$, $N_y$, and $N_z$ are the number of cells in each direction, respectively.

4) Construct 9 stencils (for the two-dimensional case, 27 stencils for three-dimensional case) using the center value of $-1$, which is the state value of the constructed cells. If any one of these has the value of zero, this means that the solid boundary is severely concave. The procedure should be repeated until the value of zero is removed. The final hole map is shown in Fig. 1d.

5) To determine whether a given cell-center point $(x_p, y_p, z_p)$ lies inside the cutting surface, compute the indices of the Cartesian grid using the following equations:

$$i_c = \text{Int}\left[(x_p - x_{\min})\frac{i_{\max} - 1}{x_{\max} - x_{\min}}\right] + 1$$

$$j_c = \text{Int}\left[(y_p - y_{\min})\frac{j_{\max} - 1}{y_{\max} - y_{\min}}\right] + 1$$

$$k_c = \text{Int}\left[(z_p - z_{\min})\frac{k_{\max} - 1}{z_{\max} - z_{\min}}\right] + 1 \tag{2}$$

If the state value of the Cartesian grid point is zero, then the cell-center point is a hole point.

**Cut–Paste Algorithm**

With conventional chimera hole-cutting methods, the mesh interface region becomes excessively large, and the interpolation region is often located near the solid wall of the interpolation regions. In addition, the hole-cutting surface is kept the same all of the time for the unsteady problem. Thus, interpolations are performed over regions where the solution gradient is relatively high and where the size of the interpolation cell is not usually of the same size as the donor cell. This may degrade the accuracy of the solution. If we move the interpolation region away from the wall, and if we keep it there during the unsteady computation, we can expect the solution to be more accurate.

The cut–paste algorithm is a two-step search method for optimal mesh interface. It is impossible to know beforehand when the front should stop if we use a one-step iterative advancing-front-type method. The cut phase continues to advance until no cell overlaps. The paste phase then adds grid points inside the fringe points to get an optimal mesh interface. This allows one to not only minimize the overlapped region but to control the number of grids overlapped, if necessary. With this two-step search method, one can eliminate the user input to control the front.

To explain the cut–paste algorithm more clearly, we have set up an example problem similar to the one used in Wey's paper.[3] Figure 2 shows four O-type grids ($41 \times 41$) overlapped on each other. Figure 3 shows the results after the hole cutting. Notice that the cell-center points are plotted in Fig. 3.

*Cut Phase*

The cut phase pushes the interpolation region away from the solid bodies. The cut phase ends when a nonoverlapped cell-centered grid is obtained. The procedure is as follows.

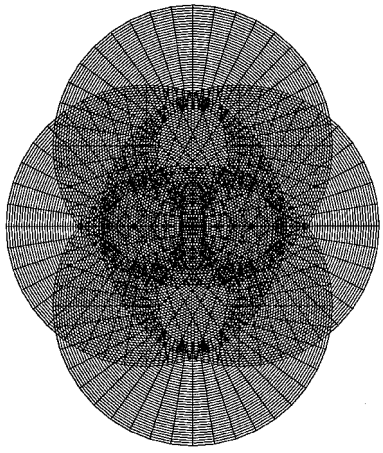

**Fig. 2 Initial chimera grids with four circles.**

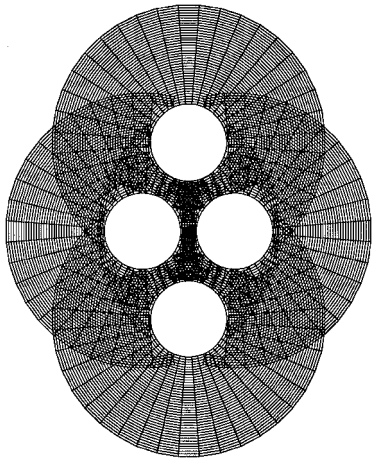

**Fig. 3 Status after the conventional chimera hole cutting.**

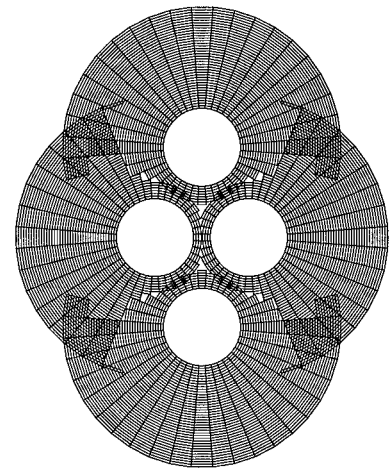

**Fig. 4 Status of grid points after three iterations of the cut phase.**

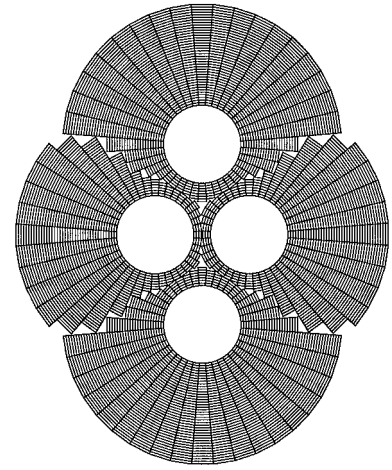

**Fig. 5 Status of grid points after seven iterations of the cut phase.**

1) Construct a front with fringe points. At the start, every point of the front is active.

2) Advance the front in the $I_s$, $J_s$, and $K_s$ directions away from the cutting surface.

3) Find the donor cells for the point using the two-step search method, which will be described in the next subsection.

4) If all of the donor cells of the point are normal points, change the status of the point to hole point. Otherwise, deactivate it.

5) Repeat the loop until there is no active point.

Figure 4 shows the intermediate stage for the cut phase, and Fig. 5 shows the normal cells after the cut phase. Notice that nonoverlapped structured grids with cell-centered coordinates are obtained.

*Paste Phase*

The paste phase constructs the optimal mesh interface region and is the reverse of the cut-phase procedure. This procedure is as follows.

1) Construct a front with fringe points. At the start, every point of the front is active.

2) Advance the front in the $I_s$, $J_s$, and $K_s$ directions toward the cutting surface.

3) Find the donor cells for the point using the two-step search method.

4) Check the status of the donor cells of the point. If any one of the donor cells is a hole point, change the status of the point to that of a normal point. Otherwise, deactivate it.

5) Repeat the loop until every point is deactivated.

Figure 6 shows the normal cell after the paste phase. On comparing it with Wey's results shown in Figs. 3–5 of his paper,[3] one can see that the overlapped region is smaller and that the number of iterations is less. The final step of the paste phase is to construct two layers of interpolation. This can be done by switching hole points neighboring the fringe points to the interpolation points. The final

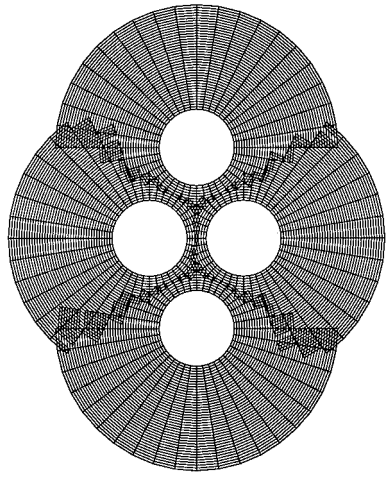

**Fig. 6 Status of grid points after paste phase (two iterations).**

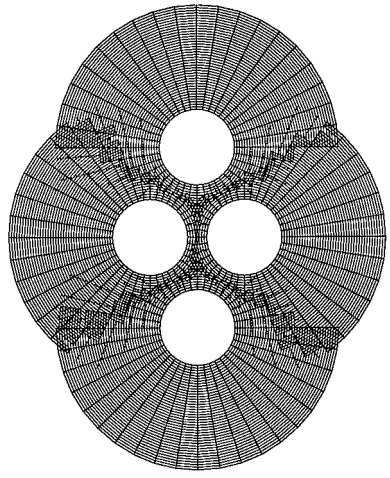

**Fig. 7 Status of grid points after cut–paste algorithm with two-layer interpolation points.**

grid is shown here in Fig. 7. The cell-centered interpolation points are marked with circles for clarity. Two layers of interpolation points are necessary to maintain the solver as second-order accurate.

A dynamic search can be used for unsteady problems to reduce the computational cost. The fringe points of the previous time step are used to construct the front. Therefore, no chimera hole cutting is needed in unsteady computations. Two or three iterations for the cut phase and the paste phase are usually needed to construct the final mesh interface. Compared to the steady state, computation time can normally be reduced by 50–60%.

### Donor Cell Identification

The transmission of the solutions occurs through the interpolation points and corresponding donor cells. Solution accuracy depends on the quality of the interpolation points. As stated in the preceding subsection, donor cell identification is repeatedly used in the cut–paste algorithm to determine the status of the front. Thus, use of a fast and reliable donor cell identification method is necessary for the success of this chimera method. We use a two-step search method: a stencil walk and a gradient search. The stencil walk finds the nearest point to a given interpolation point by comparing the distance between the interpolation point and the donor candidate cells. When the grid is severely skewed, the stencil walk fails to find correct donor cells. However, the gradient search checks whether the interpolation point actually lies inside the cube that is made with the eight donor candidate cells. Because Newton–Raphson iteration is used to determine the isoparametric coordinates, the computational cost is relatively high. Thus, the two-step search maintains a reasonable computational cost while guaranteeing the success of donor cell identification. For multiple overlapped regions, the donor cell is determined among possible donor cells according to a user-specified preference list. For unsteady problems, the search cost can be reduced by using the donor cells at the previous time step as the initial search data.

## Solver

The compressible Euler equations can be written in integral form over a control volume $V(t)$ moving with velocity $\boldsymbol{\xi}$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V(t)} \boldsymbol{Q}\,\mathrm{d}V + \int_{\partial V(t)} \hat{\boldsymbol{F}}\,\mathrm{d}S = 0 \tag{3}$$

with

$$\boldsymbol{Q} = (\rho, \rho u, \rho v, \rho w, e)^T \tag{4}$$

$$\hat{\boldsymbol{F}} = (i\boldsymbol{E} + j\boldsymbol{F} + k\boldsymbol{G} - \boldsymbol{Q}\boldsymbol{\xi})\cdot\boldsymbol{n} = \begin{bmatrix} \rho U \\ \rho u U + p n_x \\ \rho v U + p n_y \\ \rho w U + p n_z \\ (e+p)U + p\xi_t \end{bmatrix} \tag{5}$$

also

$$U = (\boldsymbol{v} - \boldsymbol{\xi})\cdot\boldsymbol{n} = -\xi_t + n_x u + n_y v + n_z w \tag{6}$$

$$\xi_t = \boldsymbol{\xi}\cdot\boldsymbol{n} = \xi_x n_x + \xi_y n_y + \xi_z n_z \tag{7}$$

The system of Euler equations is discretized using a finite volume method in conjunction with Roe's approximated Riemann solver.[9] MUSCL extrapolation of primitive variables is used to obtain second-order spatial accuracy, whereas Van Albada's or a min-mod limiter is used to maintain total variation diminishing property near shocks. Also, the entropy fix, suggested by Harten, is applied to remove the glitch near sonic points. Dual time stepping[10] with alternating direction implicit is used to advance the solution in time. This allows not only the use of large time increment but also the maintenance of temporal accuracy. The dual time stepping eliminates factorization and linearization errors by iterating the solutions along artificial time. The resulting discretized form of Eqs. (3) can be written as

$$\boldsymbol{L}_i\boldsymbol{D}^{-1}\boldsymbol{L}_j\boldsymbol{D}^{-1}\boldsymbol{L}_k\Delta\boldsymbol{Q} = -k_r\Delta\tau\boldsymbol{R} \tag{8}$$

where

$$\boldsymbol{R} = (\boldsymbol{Q}^n - \boldsymbol{Q})/\Delta t + (1/V)\{(\tilde{\boldsymbol{F}}\Delta S)_{i+1/2} - (\tilde{\boldsymbol{F}}\Delta S)_{i-1/2} + (\tilde{\boldsymbol{F}}\Delta S)_{j+1/2} - (\tilde{\boldsymbol{F}}\Delta S)_{j-1/2} + (\tilde{\boldsymbol{F}}\Delta S)_{k+1/2} - (\tilde{\boldsymbol{F}}\Delta S)_{k-1/2}\}^n$$

$$\boldsymbol{D} = \left(\frac{\Delta\tau}{\Delta t} + 1\right)\boldsymbol{I} \tag{9}$$

$$\boldsymbol{L}_i = \left(\boldsymbol{D} + k_r\frac{\Delta\tau}{V}\left\{\left(\frac{\partial\tilde{\boldsymbol{F}}_{i+1/2}}{\partial\boldsymbol{Q}_{i+1}} + \frac{\partial\tilde{\boldsymbol{F}}_{i+1/2}}{\partial\boldsymbol{Q}_i}\right)\Delta S_{i+1/2} - \left(\frac{\partial\tilde{\boldsymbol{F}}_{i-1/2}}{\partial\boldsymbol{Q}_i} + \frac{\partial\tilde{\boldsymbol{F}}_{i-1/2}}{\partial\boldsymbol{Q}_{i-1}}\right)\Delta S_{i-1/2}\right\}\right) \tag{10}$$

Other factors take similar forms. The number of dual time iterations is found to be around 20, sufficient for accurate simulations through numerical experiments. A detailed description of the solver may be found in Ref. 11.

The time step size in time accurate computation using the chimera grid is limited not only by the numerical instability and the accuracy of the CFD method but also by the time step size limitation. The latter limitation arises when the hole points in the previous time step become normal points, which results in the invalid computation of the time term, $\partial\boldsymbol{Q}/\partial t$. This limitation can be avoided by reinterpolating these points at the previous time step.

## Results

In this section, the numerical results for two- and three-dimensional problems are presented. The two-layer interpolation region is used to guarantee second-order accuracy. For clarity, the outer boundaries of the geometry are shown in grid and pressure contour figures. The gap between the outer boundary of minor grid and normal grid points shows two-layer interpolation regions in the grid plot figures.

### Two NACA0012 Airfoils

The first example is a two-dimensional problem. Figure 8 shows two identical C grids ($217 \times 35$) and the H grid ($97 \times 77$) overlapped on each other. The resulting chimera grid using our method is shown in Fig. 9. With a DEC-3000-300 computer, 1434 interpolation points are found in 3.2 s for the full construction of the chimera grids. The numbers of iterations for the cut and paste phases are 44 and 6, respectively. Figure 9 clearly shows the advantages of using our method. The overlapped regions are formed as far as possible from the airfoils where the flow gradient is relatively high. This can reduce errors that may occur by interpolating the solution near stiff flow gradients.

Figure 10 shows the pressure contours with the grid shown in Fig. 9. The pressure contours with the multiblock grid shown in Fig. 11 is shown in Fig. 12. The freestream Mach number is 0.55,

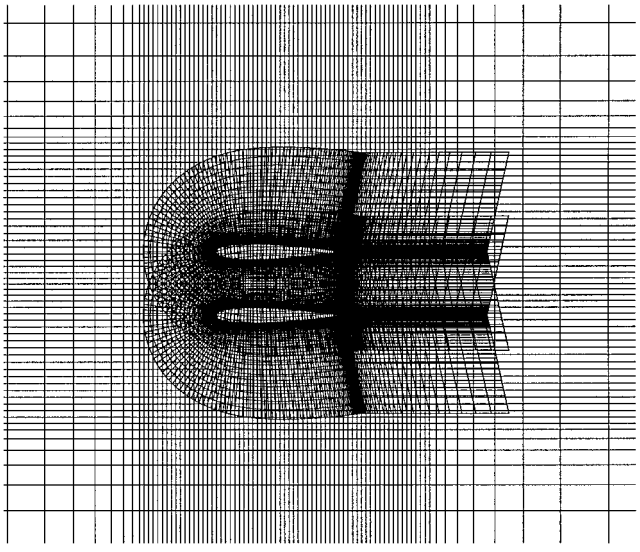

**Fig. 8 Initial chimera grids: two NACA0012 airfoils.**

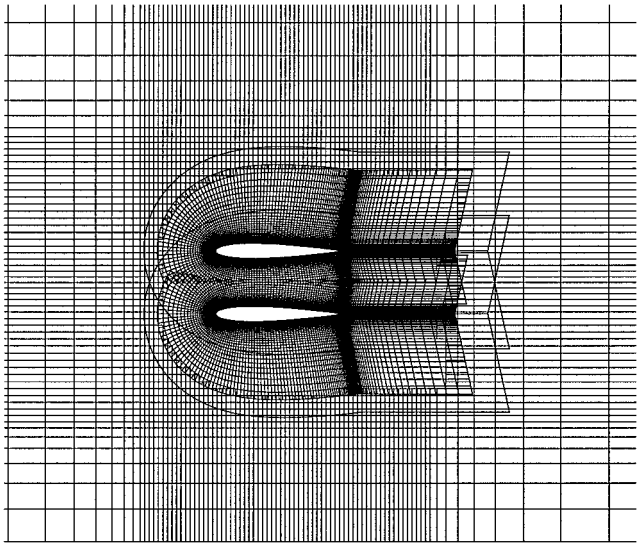

**Fig. 9 Chimera grids: two NACA0012 airfoils.**

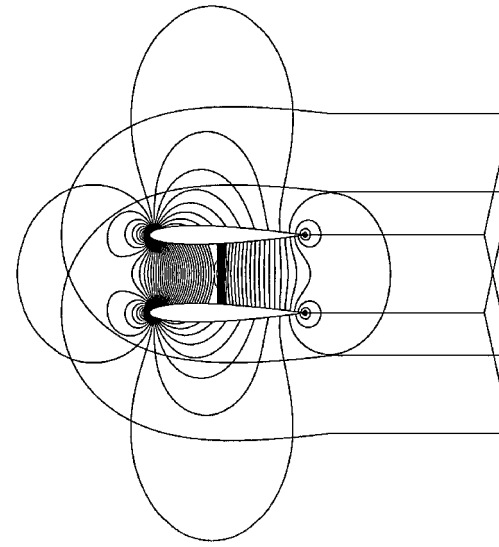

**Fig. 10 Pressure contours (minimum value = 0.4, maximum value = 0.85, and $\Delta$ = 0.01): two NACA0012 airfoils, $M_\infty$ = 0.55 and $\alpha$ = 0 deg.**

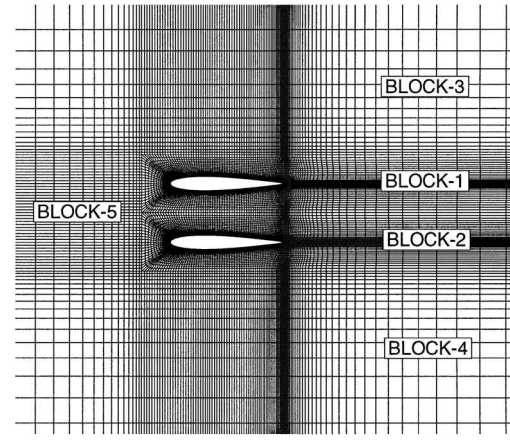


**Fig. 11 Multiblock grids: two NACA0012 airfoils.**

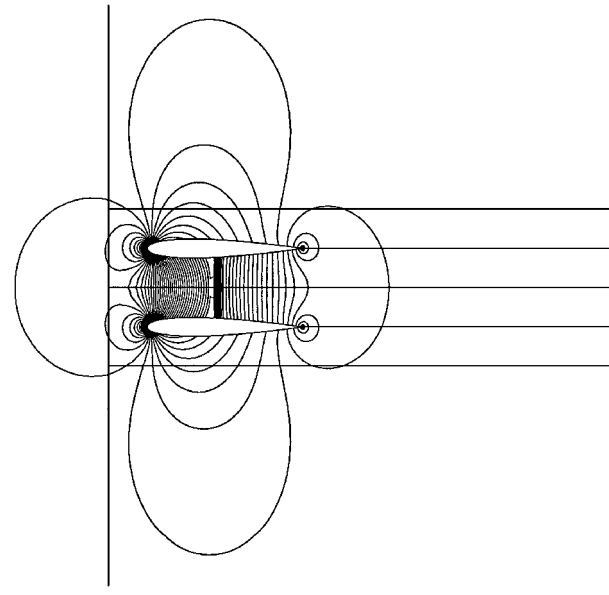

**Fig. 12 Pressure contours (minimum value = 0.4, maximum value = 0.85, and $\Delta$ = 0.01): multiblock grids, two NACA0012 airfoils, $M_\infty$ = 0.55 and $\alpha$ = 0 deg.**

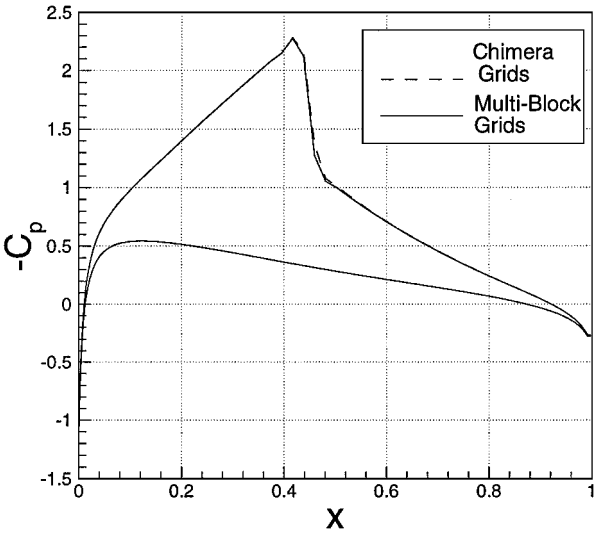


**Fig. 13 $C_p$ on the upper and lower airfoils: two NACA0012 airfoils, $M_\infty = 0.55$ and $\alpha = 0$ deg.**

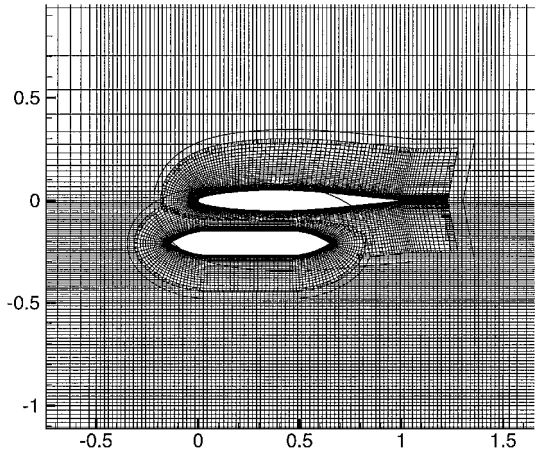

**Fig. 14 Chimera grids: steady-state case, airfoil plus store drop test case.**

with an angle of attack of 0 deg. The number of grid points on the airfoils are kept equal for the chimera grid and the multiblock grid for a fair comparison. As may be seen from Fig. 13, the shock obtained with the chimera grid is as sharp as that obtained with the multiblock grid, and the contour lines smoothly match across the overlapped region.

### Two-Dimensional Store Separation from a Wing Section

The second example is an unsteady problem. A two-dimensional store is separating from an airfoil with three degrees of freedom, which is a two-dimensional version of the Eglin/store problem (see Ref. 12). The freestream Mach number chosen is 0.6. Figures 14 and 15 show the initial grid and the normalized pressure contours of the initial condition. The computational domain is decomposed into three different grids: the C-type grid about the airfoil ($189 \times 31$), the O-type grid ($141 \times 25$) about the store, and the H-type grid ($148 \times 145$) for the entire computational domain. The total number of interpolation points for the initial grid is 1223, and 3.1 s are required for the full construction of the chimera grids in the steady case with the same computer.

In the unsteady case, the total number of interpolation points is 1220, and the computing time is 0.8 s, respectively, on average. Also, less than three iterations are required in the cut and paste procedures during the unsteady run. The grids and the pressure contours of the subsequent time steps are shown in Figs. 16–19. Figures 16–19 show that the contour lines of the store grid, the airfoil grid, and the major grid match well even in the shock region.

The mass and moment of inertia of the store are chosen to be large compared with the aerodynamic forces and moments of the store ($m/\rho_\infty L^3 = 10$ and $I_y/\rho_\infty L^5 = 10$). Otherwise, the simulation would terminate prematurely due to the collision of the store with the airfoil. The time used here is nondimensionalized with the

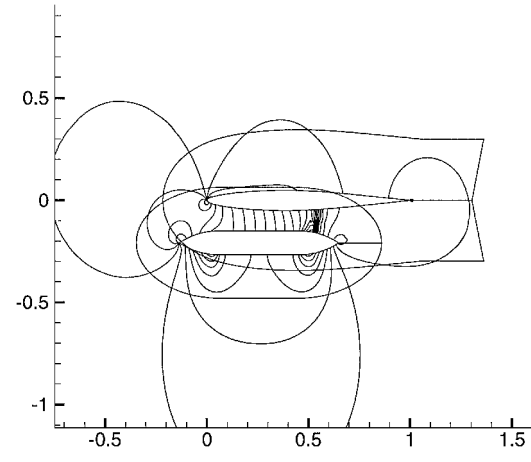

**Fig. 15 Pressure contours (minimum value = 0.25, maximum value = 0.85, and $\Delta = 0.04$): steady state case, airfoil plus store drop test case, $M_\infty = 0.6$, $m/\rho_\infty L^3 = 10.0$, and $I_y/\rho_\infty L^5 = 10.0$.**

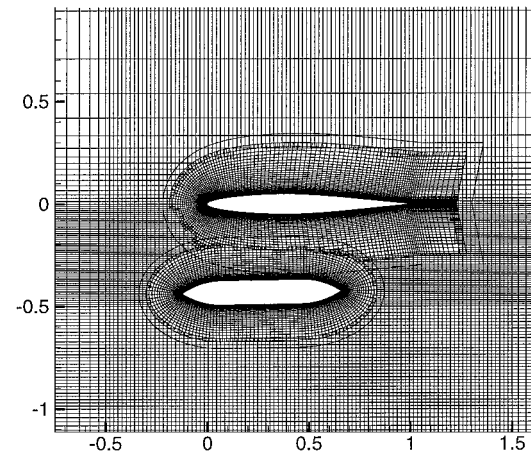

**Fig. 16 Chimera grids: $t^* = 5.45$, airfoil plus store drop test case.**

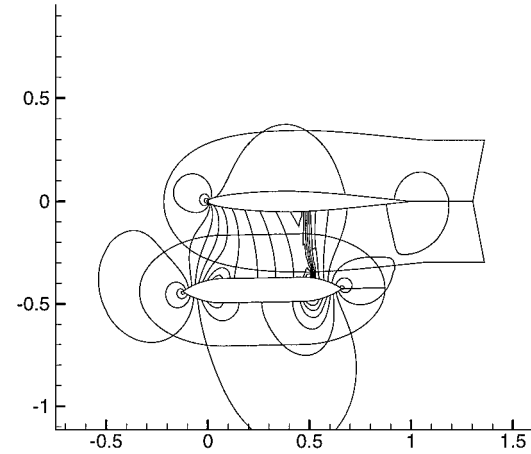

**Fig. 17 Pressure contours (minimum value = 0.25, maximum value = 0.85, and $\Delta = 0.04$): $t^* = 5.45$, airfoil plus store drop test case, $M_\infty = 0.6$, $m/\rho_\infty L^3 = 10.0$, and $I_y/\rho_\infty L^5 = 10.0$.**

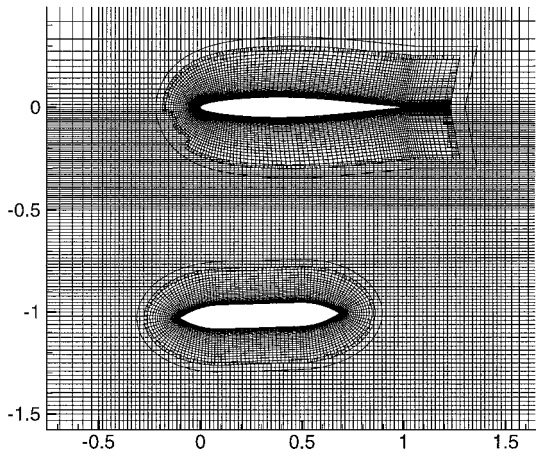


**Fig. 18 Chimera grids: $t^* = 12.45$, airfoil plus store drop test case.**

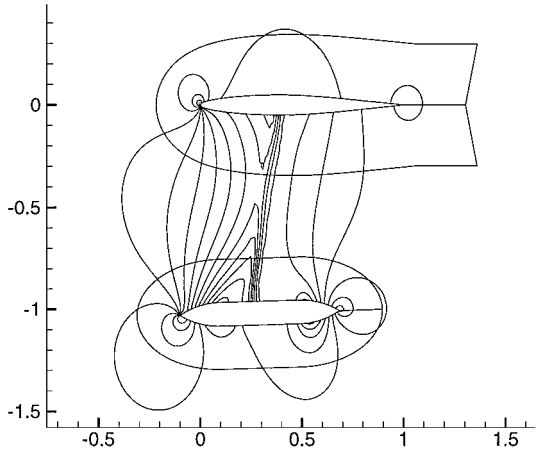


**Fig. 19 Pressure contours (minimum value = 0.25, maximum value = 0.85, and $\Delta = 0.04$): $t^* = 12.45$, airfoil plus store drop test case, $M_\infty = 0.6$, $m/\rho_\infty L^3 = 10.0$, and $I_y/\rho_\infty L^5 = 10.0$.**

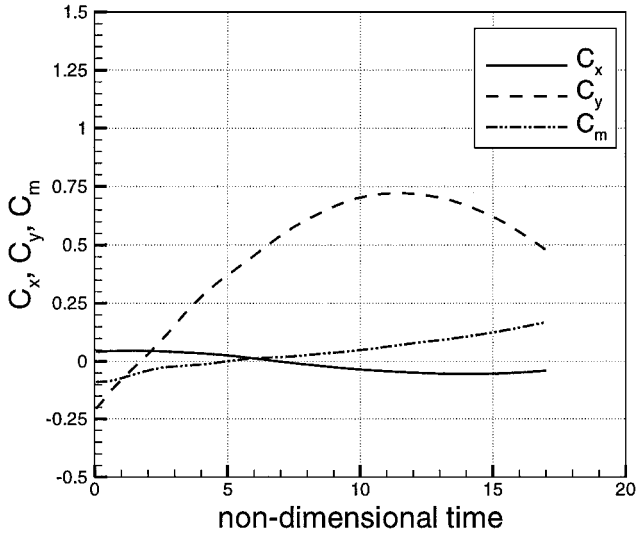


**Fig. 20 Force and moment coefficients: airfoil plus store drop test case, $M_\infty = 0.6$, $m/\rho_\infty L^3 = 10.0$, and $I_y/\rho_\infty L^5 = 10.0$.**

chord length of airfoil $c$ and the speed of sound $a$, that is, $t^* = ta/c$. The local angle of attack of the store increases as the store separates from the airfoil. Because the store is aerodynamically unstable (this is true for any two-dimensional body), it would pitch up, experience increase in lift, and eventually collide with the airfoil. In Figs. 20 and 21, the normal force, axial force, and moment coefficients are shown, as well as the trajectory of the store. Even with the heavy inertia of the store, the store shows a pitch-up tendency.

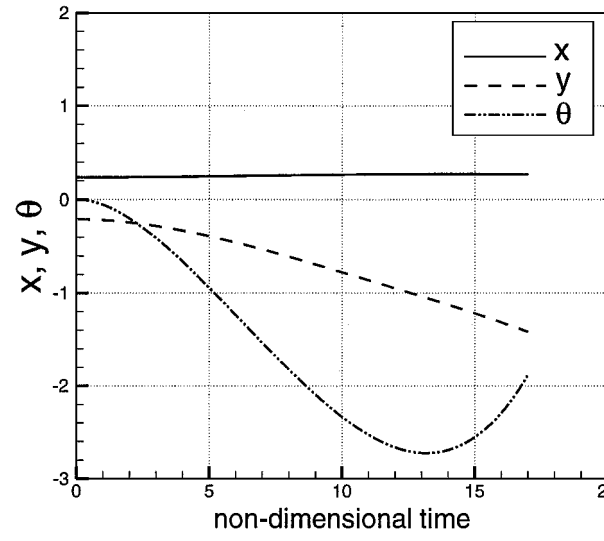


**Fig. 21 Store trajectory: airfoil plus store drop test case, $M_\infty = 0.6$, $m/\rho_\infty L^3 = 10.0$, and $I_y/\rho_\infty L^5 = 10.0$.**

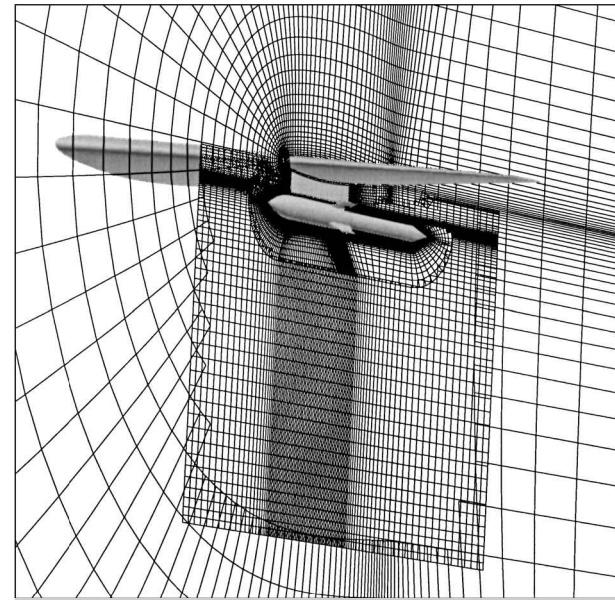

**Fig. 22 Chimera grids: Eglin test case.**

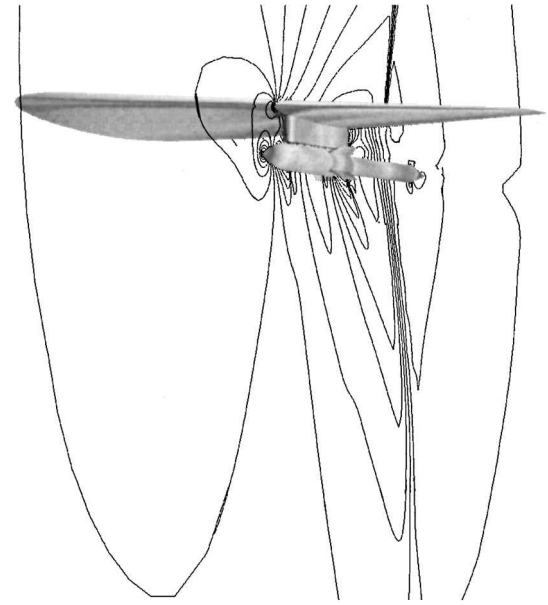

**Fig. 23 Pressure contours: Eglin test case, $M_\infty = 0.95$, and $\alpha = 0.0$ deg.**

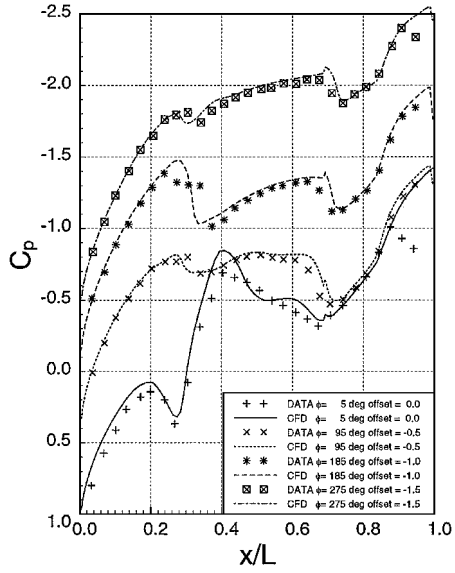


**Fig. 24** $C_p$ **on the store: Eglin test case,** $M_\infty$ **= 0.95, and** $\alpha$ **= 0.0 deg.**

#### Generic Wing/Pylon/Store System

The final example is a steady three-dimensional problem. A C–H-type grid generated over the wing and an O–O-type grid over the pylon are overlapped and used as a major grid system, together with an H–H grid to ease the interpolation. A grid system of four block grids is used to discretize the domain near the missile and is the only grid system allowed to move freely. The total number of grids is approximately $1.2 \times 10^6$ points. The freestream Mach number is 0.95, and the altitude is 26,000 ft. Figure 22 shows the constructed chimera grid at the center of the pylon. The sting is modeled as a wind-tunnel model. Notice that the overlapped region is minimized and placed midway between the solid walls. Figure 23 shows the pressure contours in this case. The complexity of the shock interaction downstream of the store can be clearly seen from Fig. 23. The shock in the overlapped region is as sharp as in the nonoverlapped region. In Fig. 24, the surface pressure coefficient distributions at four circumferential positions are compared with the experiments in Ref. 12. The agreement with the experiments is excellent.

## Conclusions

A newly developed procedure for chimera-domain decomposition consisting of the cut–paste algorithm and the two-step search method has been presented. In the new procedure, the chimera grids can be generated automatically with minimal user input. Two- and three-dimensional examples demonstrate that this procedure is effective for both steady and unsteady problems.

## Acknowledgment

This work was supported by the Korea Science and Engineering Foundation under Contract 96-0200-03-01-3.

## References

[1]Suhs, N. E., and Tramel, R. W., "PEGASUS 4.0 User's Manual," U.S. Air Force Arnold Engineering Development Center, TN, AEDC-TR-91-8, June 1991.

[2]Meakin, R., "A New Method for Establishing Intergrid Communication Among Systems of Overset Grids," AIAA Paper 91-1586, June 1991.

[3]Wey, T. C., "Development of a Mesh Interface Generator for Overlapped Structured Grids," AIAA Paper 94-1924, 1994.

[4]Chiu, I., and Meakin, R., "On Automating Domain Connectivity for Overset Grids," AIAA Paper 95-0854, Jan. 1995.

[5]Dougherty, F. C., "Development of a Chimera Grid Scheme with Applications to Unsteady Problems," Ph.D. Dissertation, Dept. of Aeronautics and Astronautics, Stanford Univ., Stanford, CA, June 1985.

[6]Milgram, M. S., "Does a Point Lie Inside a Polygon?," *Journal of Computational Physics*, Vol. 84, No. 1, 1989, pp. 134–144.

[7]Steger, J. L., and Benek, J. A., "On the Use of Composite Grid Schemes in Computational Aerodynamics," *Computer Methods in Applied Mechanics and Engineering*, Vol. 64, Oct. 1987, pp. 301–320.

[8]Lo, S. H., "Perspective Projection of Non-Convex Polyhedra," *International Journal for Numerical Methods in Engineering*, Vol. 26, No. 7, 1988, pp. 1485–1506.

[9]Roe, R. L. "Approximated Riemann Solvers, Parameter Vectors and Difference Scheme," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372.

[10]Merkle, C. L., and Athavale, M., "Time-Accurate Unsteady Incompressible Flow Algorithms Based on Artificial Compressibility," AIAA Paper 87-1137, 1987.

[11]Lee, S., "Numerical Computation of Flows About an Aircraft with Flow-Through Inlet," *Journal of the Korean Society for Aeronautical and Space Sciences*, Vol. 25, No. 5, 1997, pp. 16–23 (in Korean).

[12]Lijewski, L. E., and Suhs, N. E., "Time-Accurate Computational Fluid Dynamics Approach to Transonic Store Separation Trajectory Prediction," *Journal of Aircraft*, Vol. 31, No. 4, 1994, pp. 886–891.