AIAA JOURNAL
Vol. 33, No. 4, April 1995

Dynamic Unstructured Method for Flows Past Multiple Objects in Relative Motion

Kamakhya P. Singh,* James C. Newman,† and Oktay Baysal‡
Old Dominion University, Norfolk, Virginia 23529

A new methodology is developed to simulate unsteady flows about objects in relative motion and compute the trajectory of their motion as determined by this flowfield. The method couples the fluid dynamics and rigid-body dynamics equations to capture the time-dependent interference between stationary and moving boundaries. The unsteady, compressible, inviscid (Euler) equations are solved on dynamic unstructured grids by an explicit, finite-volume, upwind method. For efficiency, the grid adaptation is perfomed within a window around the moving object. The Eulerian equations of the rigid-body dynamics are solved by a Runge-Kutta method in a noninertial frame of reference. This dynamic, unstructured flow solver is validated by computing the flow past a sinusoidally pitching airfoil and comparing these results with the experimental data. The trajectory code is tested by computing the six-degree-of-freedom trajectory of a store separating from a wing using the experimentally determined force and moment fields, then comparing with the experimental trajectory. Finally, the overall methodology is used for two two-dimensional examples: the flow past a pitching and plunging airfoil, and the free fall of a store after separation from a wing section.

Introduction

BECAUSE of the lower cost and time incurred to generate and dynamically test existing as well as conceptual configurations, computational fluid dynamics (CFD) has emerged as a crucial technology for the development of advanced aerospace vehicles. Furthermore, experimental test facilities for unsteady flowfields involving dynamic bodies, particularly at high speeds, are prohibitively scarce. Thus, increased importance is being placed on unsteady CFD methods as they probably are the only approaches available for predicting the pertinent transient phenomena. These simulations also require the analysis of nontrivial and geometrically complex configurations. Some typical examples, where the prediction of dynamic loads, moments, and trajectories are essential, include flight maneuvers, store (or missile) separation sequences, escape-pod ejections, detachment of multistage-rocket components, and separation of booster tanks from the Space Shuttle. To answer this demand, new numerical algorithms and grid generation methods need to be developed and existing ones to be made more efficient and robust.

Unstructured grid methods have the potential to handle these complex geometries somewhat easier than their structured grid counterparts. This can be attributed to the fact that triangles and tetrahedra are the simplest geometrical shapes possessing area and volume, respectively. Hence, they are capable of discretizing irregularly shaped domains more efficiently and with less effort. For the problems which involve moving bodies, some method of tracking the body must be implemented.

Examples of current methods being researched to handle this class of problems are the dynamic overlapped grids (e.g., Refs. 1–3), grid adaptation for structured and unstructured grids (e.g., Refs. 4–6), and mesh regeneration techniques for unstructured grids (e.g., Ref. 7). Developed in the present work is an adaptive window method to efficiently redistribute the unstructured grid cells about moving bodies, and a dynamic algorithm which is used to aerodynamically determine the motion of released bodies.

Presented in this paper is a discussion of the adaptive window procedure via demonstrative examples followed by the details of the dynamic solution algorithm. In an attempt to provide a comparison with published experimental data,[8] the flowfield about a sinusoidally oscillating NACA 0012 airfoil is computed. Then, the adaptive window procedure is applied to a more complex prescribed motion, where the airfoil is given a three-degrees-of-freedom (DOF) trajectory.

The primary focus of the paper is the development of a dynamic unstructured grid method capable of aerodynamically determining the motion and the related unsteady flowfield of released bodies. This method is demonstrated with an external airfoil-store (AS) configuration, where the 3-DOF trajectory of the store is determined based on the computed aerodynamic loads.

Governing Equations and Solution Algorithm

Equations of Fluid Flow

In this section, the governing equations for unsteady fluid flows and the solution algorithm used in the present computations for moving boundary problems are briefly discussed. A detailed discussion of the steady flow solution algorithm on stationary grids, which serves as the baseline to the present dynamic algorithm, may be found in Ref. 9.

The three-dimensional time-dependent Euler equations for dynamic grids can be expressed in the integral form for a bounded domain Ω with a boundary $\partial\Omega$ as

$$\frac{\partial}{\partial t}\iiint_{\Omega} \boldsymbol{Q}\,\mathrm{d}V + \iint_{\partial\Omega} \boldsymbol{F}(\boldsymbol{Q})\cdot\hat{\eta}\,\mathrm{d}S = 0 \tag{1}$$

where

$$\boldsymbol{Q} = [\rho, \rho u, \rho v, \rho w, \rho e_0]^T \tag{2}$$

and

$$\boldsymbol{F}(\boldsymbol{Q})\cdot\hat{\eta} = (V\cdot\hat{\eta})\begin{Bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ \rho e_0 + p\end{Bmatrix} + p\begin{Bmatrix}0\\ \eta_x\\ \eta_y\\ \eta_z\\ a_t\end{Bmatrix} \tag{3}$$

Presented as Paper 94-0058 at the AIAA 32nd Aerospace Sciences Meeting, Reno, NV, Jan. 10–13, 1994; received March 1, 1994; revision received Sept. 23, 1994; accepted for publication Oct. 10, 1994.

*Graduate Research Assistant, School of Mechanical and Aerospace Engineering. Student Member AIAA.

†Graduate Research Assistant, School of Mechenical and Aerospace Engineering.

‡Professor, Department of Aerospace Engineering. Associate Fellow AIAA.

The velocity vector $\boldsymbol{V}$ of a fluid particle is written relative to the motion of the dynamic grids,

$$\boldsymbol{V} = \{(u - x_t), (v - y_t), (w - z_t)\} \tag{4}$$

and the contravariant face speed is computed as

$$a_t = x_t \eta_x + y_t \eta_y + z_t \eta_z \tag{5}$$

In these equations, η_x, η_y, and η_t denote the components of the unit vector $\hat{\eta}$ which points normal to the cell face, and x_t, y_t, and z_t are the grid speed terms in the x, y, and z directions, respectively.

Spatial Discretization

Spatial discretization may be accomplished by either Roe's flux difference splitting[10] or van Leer's flux vector splitting.[11,12] For Roe's scheme, the flux across each cell face is calculated using a numerical flux formula which essentially represents a central difference of the inviscid fluxes plus an upwind correction. Roe's scheme used for stationary-boundary problems may be modified for moving meshes by redefining the Roe-averaged (denoted by $\sim$) contravariant velocity as

$$\tilde{U} = \tilde{u}\eta_x + \tilde{v}\eta_y + \tilde{w}\eta_z - a_t \tag{6}$$

For van Leer's scheme, the flux vectors are given in terms of the Mach number normal to the face. This results in the possibility of the flow being supersonic or subsonic through a face. Supersonic fluxes are evaluated as

$$\hat{F}^+ = [\boldsymbol{F}(Q) \cdot \hat{\eta}]^+, \qquad \hat{F}^- = [\boldsymbol{F}(Q) \cdot \hat{\eta}]^- = 0$$

$$M_n \geq 1 \tag{7}$$

$$\hat{F}^- = [\boldsymbol{F}(Q) \cdot \hat{\eta}]^-, \qquad \hat{F}^+ = [\boldsymbol{F}(Q) \cdot \hat{\eta}]^+ = 0$$

$$M_n \leq -1 \tag{8}$$

For subsonic flow, the fluxes are split into the following contributions:

$$F_k = \hat{F}^+(Q^-) + \hat{F}^-(Q^+) \tag{9}$$

where

$$\hat{F}^{\pm} = \begin{Bmatrix} f^{\pm}_{\text{mass}} \\ f^{\pm}_{\text{mass}}\{u + \eta_x(-\bar{U} \pm 2a)/\gamma\} \\ f^{\pm}_{\text{mass}}\{v + \eta_y(-\bar{U} \pm 2a)/\gamma\} \\ f^{\pm}_{\text{mass}}\{w + \eta_z(-\bar{U} \pm 2a)/\gamma\} \\ f^{\pm}_{\text{energy}} \end{Bmatrix} \tag{10}$$

with

$$f^{\pm}_{\text{mass}} = \pm\frac{\rho a}{4}(M_n \pm 1)^2 \tag{11}$$

$$f^{\pm}_{\text{energy}} = f^{\pm}_{\text{mass}}\left[\frac{(1-\gamma)\bar{U}^2 \pm 2(\gamma-1)\bar{U}a + 2a^2}{(\gamma^2-1)} + \frac{u^2+v^2+w^2}{2} + \frac{a_t(-\bar{U} \pm 2a)}{\gamma}\right] \tag{12}$$

In Eqs. (7–12), $\bar{U}$ is the adjusted contravariant velocity, which is the scalar product of the modified velocity in Eq. (4) with the unit normal vector to the face.

In a first-order scheme, the state of the primitive variables at each cell face is made equal to the cell-centered averages on both sides of the face. A higher order scheme is obtained by expanding the cell-centered solution to each cell face using a Taylor series expansion which may be expressed as

$$q(x, y, z) = q(x_c, y_c, z_c) + \nabla q_c \cdot \Delta \boldsymbol{r} + \vartheta(\Delta r^2) \tag{13}$$

where the solution gradient ∇q_c at the center of the cell is found using the geometric invariant features of triangles and tetrahedra. The expression for the solution gradient at the cell center may be written as

$$\nabla q_c \cdot \Delta \boldsymbol{r} = \left[\frac{\frac{1}{3}(q_{n1} + q_{n2} + q_{n3}) - q_{n4}}{4\Delta r}\right]\Delta r \tag{14}$$

where q_{n1}, q_{n2}, and q_{n3}, denote the primitive variables at the three nodes that constitute the face through which the flux passes, Δr is the distance from the centroid of the tretrahedra to the center of that face, and q_{n4} is the primitive variables at the fourth node of the tetrahedra.

Time Integration

Once the fluxes have been evaluated for each cell face using the preceding higher order scheme, the spatially discretized form of the governing equations are then integrated in time using the explicit fourth-order Runge-Kutta method. This method has second-order temporal accuracy and may be written as

$$Q_i^{(0)} = Q_i^n$$

$$Q_i^{(k)} = \left(\frac{V_i^n}{V_i^{n+1}}\right)Q_i^{(0)} - \alpha_k \frac{\Delta t}{V_i^{n+1}} R_i^{(k-1)}$$

$$\alpha_k = \frac{1}{m-k+1} \qquad k = 1, 2, \ldots, m \tag{15}$$

$$Q_i^{n+1} = Q_i^{(m)}$$

The indices n, m, and k indicate the time level, the order of the Runge-Kutta method, and a dummy index, respectively. V denotes the cell volume and R_i is the residual given by

$$R_i = \sum_{j=k(i)} F_{ij} A_{ij} \tag{16}$$

where A_{ij} is the area of the jth surface of ith cell and F_{ij} is the flux through A_{ij}.

Geometric Conservation Law

To avoid grid-motion induced errors when dynamic meshes are involved, the geometric conservation law (GCL) must be satisfied concurrently with the conservation of mass, momentum, and energy.[6,13,14] The integral statement of GCL may be written as

$$\frac{\mathrm{d}}{\mathrm{d}t}\iiint_{\Omega} \mathrm{d}V = \iint_{\partial\Omega} \boldsymbol{W} \cdot \hat{\eta}\, \mathrm{d}A \tag{17}$$

where $\boldsymbol{W}$ denotes the local velocity of the cell faces. Furthermore, to provide a self-consistent solution for the local cell volumes, the GCL should be integrated using the same scheme that is used for the fluid equations. A discretization of Eq. (17) has been expressed in Ref. 6 which is consistent with the preceding solution algorithm and is given by

$$V_i^{n+1} = V_i^n + \Delta t \sum_{j=k(i)} [a_t A]_{ij}^{n+1} \tag{18}$$

Thus, this equation is used to compute the local cell volumes at the current time level.

Dynamic Mesh Algorithm

Adaptation Method

The adaptation method used here has been previously reported by Batina.[6] The unstructured mesh about the body (or bodies) of interest is considered as a system of interconnected springs. This system is constructed by representing each edge of each triangle by a tension spring. Various attempts at determining the optimum relationship for specifying the spring stiffness have been made by Chakravarthy and Szema.[16] In the present work, however, the spring

stiffness is assumed inversely proportional to the length of its edge and may be written as

$$k_{ij} = 1.0/\left[(x_i - x_j)^2 + (y_i - y_j)^2\right]^{p/2} \tag{19}$$

where p (generally taken as unity) is a parameter used to control the stiffness of the spring. Then, for each mesh point, the external forces due to the connecting springs are summed and resolved into Cartesian components. The resulting set of linear systems are solved for the displacements of each node using several Jacobi iterations

$$\Delta x_j^{n+1} = \sum k_{ji} \Delta x_i^n \Big/ \sum k_{ji} \tag{20}$$

$$\Delta y_j^{n+1} = \sum k_{ji} \Delta y_i^n \Big/ \sum k_{ji} \tag{21}$$

where i is summed over all edges connected to node j. The positions of the interior points are then updated using the determined displacements.

This iterative method has the advantage of not requiring an excessive amount of memory but does require an initial guess. For the present system, only the displacements at the current time level are stored, and the initial guesses of the displacements are the displacements at the previous time level. Since the system being solved is diagonally dominant (the diagonal of each row being the sum of the spring stiffness of every node involved in that equilibrium equation) a relaxation factor may be introduced to accelerate convergence. Hence, using this successive over-relaxation method, an acceptable mesh movement is achieved in 4–6 iterations.

Adaptive Window Procedure

With the use of the described method for adapting the unstructured mesh, computational efficiency can be improved by limiting the size of the adaptation region. Limiting the size of this region is advantageous since only a small area of the mesh needs to be stored and adapted. The method used in the present work to restrict the size of the adaptation region is to create a window around the physical domain of interest. The nodal points inside this window are considered as the spring network and, thus, allowed to adapt to the body movement.

This procedure can provide significant savings in both CPU time and memory; for example, the spatial adaptation procedure can be responsible for 3–7.5% of the total CPU time for an unsteady simulation,[17] depending on whether or not local enrichment or refinement is utilized in the adaptation. Nevertheless, for unsteady flows, where it is understood that a large number of iterations are performed, small savings per iteration results in large overall savings. The present adaptation procedure, irrespective of window construction, costs 9.5 μs/node/iteration of CPU time and requires a storage of $2 + 2^*m$ per node, where m denotes the maximum number of springs connected to a node. Therefore, CPU time and memory savings may be obtained by reducing the number of nodes being adapted per itertation; for example, by only adapting 30% of the total number of nodes, a 70% savings is realized.

Creating the window may be carried out by either specifying a normal distance from the body or choosing a basis shape (circle, ellipse, etc.) around it. The entire domain is searched to locate the points which fall within the window, and those which do are flagged as window points. The window points are allowed to be adapted from one time step to the next. The next search is for the mesh points which are connected to the outermost window points. These points are flagged as window frame points. Mesh points exterior to the window and the window frame points are spatially fixed in time.

For problems in which the body has small or no translational movement, creation of the window takes place only once. However, for problems in which large movements are encountered, the window may need to be constructed on several occasions during the body's trajectory. Thus, window construction must be a quick, reliable, and automated process. In the present work a basis shape is used to specify the window, and a critical displacement is chosen to determine when a new window is needed.

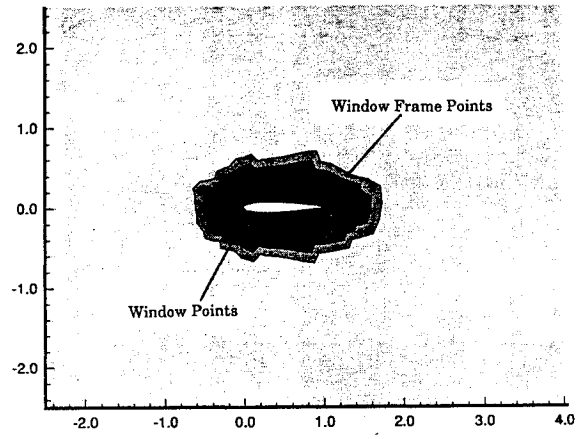


Fig. 1 Initial adaptation window for NACA 0012 airfoil.

An example illustrating the adaptive window method for moving body problems is presented: a NACA 0012 airfoil sinusoidally oscillating about its quarter chord with an amplitude of 35 deg. The window constructed about this airfoil is shown in Fig. 1. This mesh contains 1577 nodes and 3042 cells, but the adaptation window contains 569 nodes and 1180 cells. Hence, only about 30% of the original mesh is being adapted. Detailed views of the adapted mesh are given in Fig. 2. To ensure the integrity of the mesh around the airfoil, the stiffness of the springs in this region are increased by increasing p from a value of unity to 1.8 in Eq. (19).

Rigid Body Dynamics

Coupled in the overall solution algorithm are the governing equations of rigid body dynamics. These equations are used to aerodynamically determine the 6-DOF trajectory of a body from a given flowfield. That is, given the force and moment fields from the solution of the flowfield equations [Eq. (1)], the translational and the rotational motions of the body are determined by solving the Eulerian equations of the rigid body motion.[19] The displacement $\boldsymbol{r}$, velocity $\boldsymbol{v}$, and acceleration $\boldsymbol{a}$ fields are given by the following equations:

$$\boldsymbol{r} = \boldsymbol{R} + \boldsymbol{\rho} \tag{22a}$$

$$\boldsymbol{v} = \dot{\boldsymbol{r}} = \dot{\boldsymbol{R}} + (\dot{\boldsymbol{\rho}})_r + \boldsymbol{\omega} \times \boldsymbol{\rho} \tag{22b}$$

$$\boldsymbol{a} = \ddot{\boldsymbol{r}} = \ddot{\boldsymbol{R}} + \dot{\boldsymbol{\omega}} \times \boldsymbol{\rho} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \boldsymbol{\rho}) + (\ddot{\boldsymbol{\rho}})_r + 2\boldsymbol{\omega} \times (\dot{\boldsymbol{\rho}})_r \tag{22c}$$

where $\boldsymbol{R}$ is the position vector of the inertial frame's origin with respect to the noninertial coordinate system, $\boldsymbol{\rho}$ is the position vector of a point on the body with respect to noninertial coordinate system, $\boldsymbol{\omega}$ is the angular velocity of the body, and subscript r indicates an operation in the inertial frame. The three-dimensional force and moment equations are given by

$$I\dot{\boldsymbol{\omega}} + \iint_{\partial\Omega} (\boldsymbol{\omega} \cdot \boldsymbol{\rho}) \cdot (\boldsymbol{\rho} \times \boldsymbol{\omega})\, \mathrm{d}\Omega = \sum \boldsymbol{\rho} \times \boldsymbol{F} \tag{23}$$

$$m\dot{\boldsymbol{v}}_{\text{c.g.}} = \sum \boldsymbol{F} = m\boldsymbol{g} - \iint_{\partial\Omega} p\boldsymbol{n}\, \mathrm{d}\Omega \tag{24}$$

Thus, there are six equations that must be solved [for a planar motion, the second term in Eq. (23) vanishes]. An outline of the solution method is itemized as follows.

1) Given an initial value of the grid speed, the governing equations of flow [Eq. (1)] are solved to obtain the force and moment fields in the inertial frame. Then, moment and force fields are transformed to a noninertial frame (body coordinate system) using the Euler angles,[18] so that they can be used in solving Eqs. (23) and (24).

2) The moment equations [Eqs. (23)] are solved simultaneously to obtain the angular acceleration. Then, the angular acceleration is integrated in time using a fourth-order Runge-Kutta method to obtain the angular velocity field.

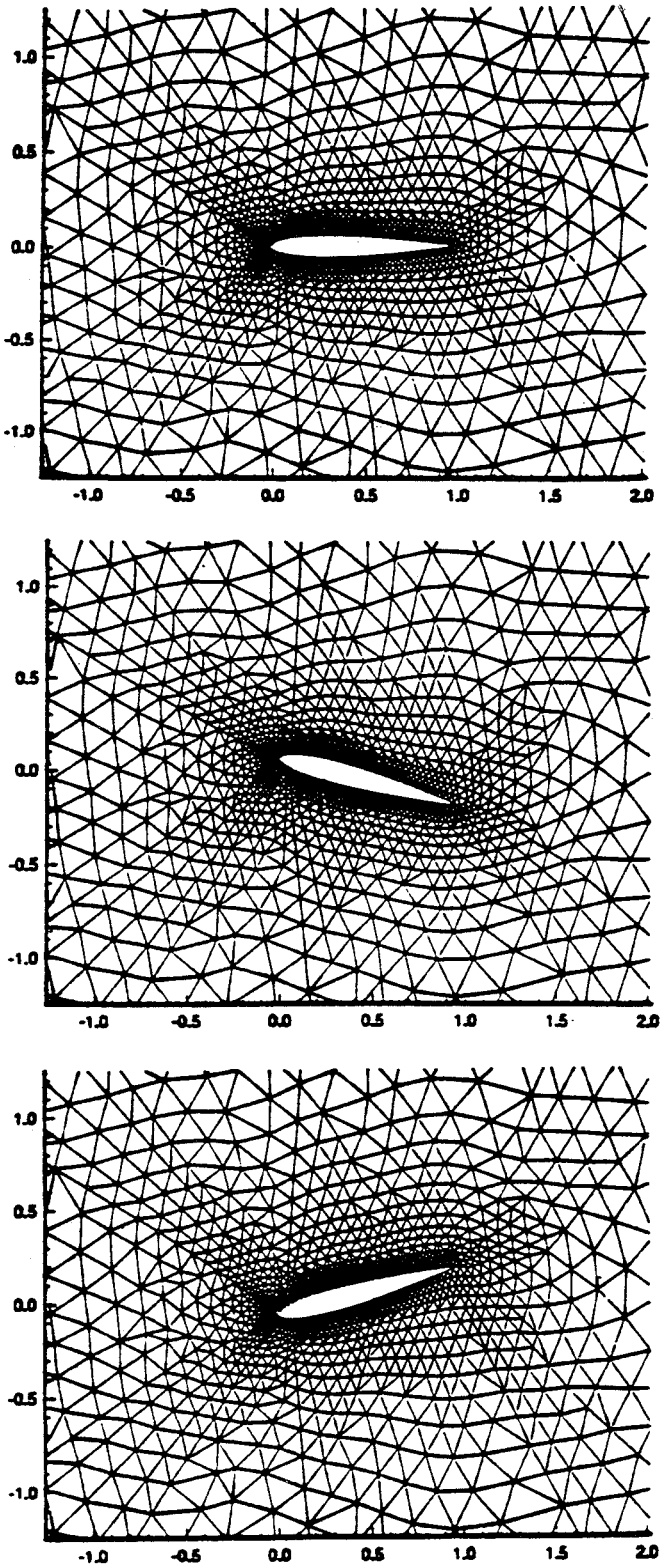


Fig. 2 Adaptation for the sinusoidally oscillating airfoil.

3) Using the obtained angular velocities, the angular rate equations,

$$\dot{\psi} = (\omega_y \sin\phi + \omega_z \cos\phi)/\cos\theta$$

$$\dot{\theta} = \omega_y \cos\phi - \omega_z \sin\phi \tag{25}$$

$$\dot{\phi} = \omega_x + \omega_y \sin\phi \tan\theta + \omega_z \cos\phi \tan\theta$$

are integrated in time to obtain the angular displacements.

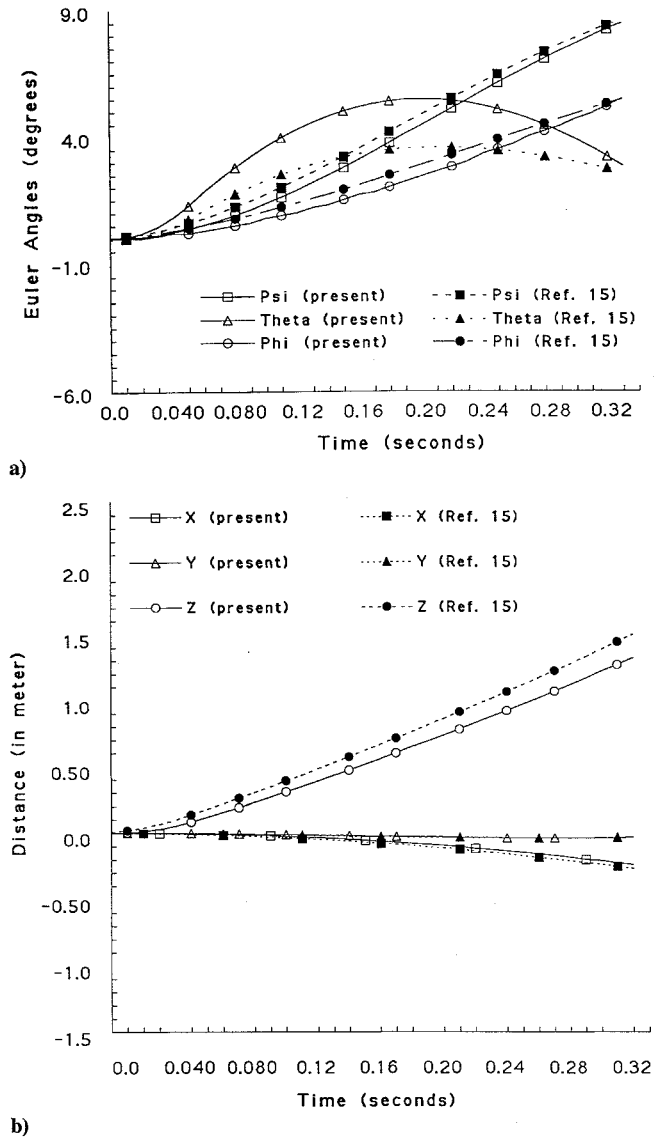


Fig. 3 Comparison of trajectory predictions: a) Euler angles and b) displacements.

4) With the obtained angular velocities and angular displacements, the force equations [Eqs. (24)] may now be solved simultaneously to obtain the translational accelerations. These accelerations are then integrated to give the translational velocity field.

5) Using the translational velocities and the angular velocities, the velocities of the off-surface mesh points (these points move with respect to the noninertial body axes) are calculated by solving Eq. (22b). The velocities of the surface points are also found from Eq. (22b), but with $\dot{\rho}_r = 0$ since the body is rigid.

6) With the velocity field at the current time step, the coordinates of the grid points and the body's center of mass are updated.

7) The position of the body is checked to see if a new window is needed; if so, a new window is created about the body in the current position.

8) The mesh is then adapted to the body movement.

9) The solution is advanced by one time step, and items 1–8 are then repeated.

An attempt has been made to validate the present 6-DOF trajectory method by comparing its results with the available data. As reported in Ref. 15, the separation of a store from a wing was wind-tunnel tested in a quasisteady mode. That is, steady-state measurements were obtained when the store was placed at sequential positions on its trajectory, which was computed based on the measured aerodynamic forces and moments. These experimentally measured aerodynamic forces and moments were also used as the input to the present method. Then, the present trajectory was compared with the trajectory reported in Ref. 15 (see Fig. 3). It should

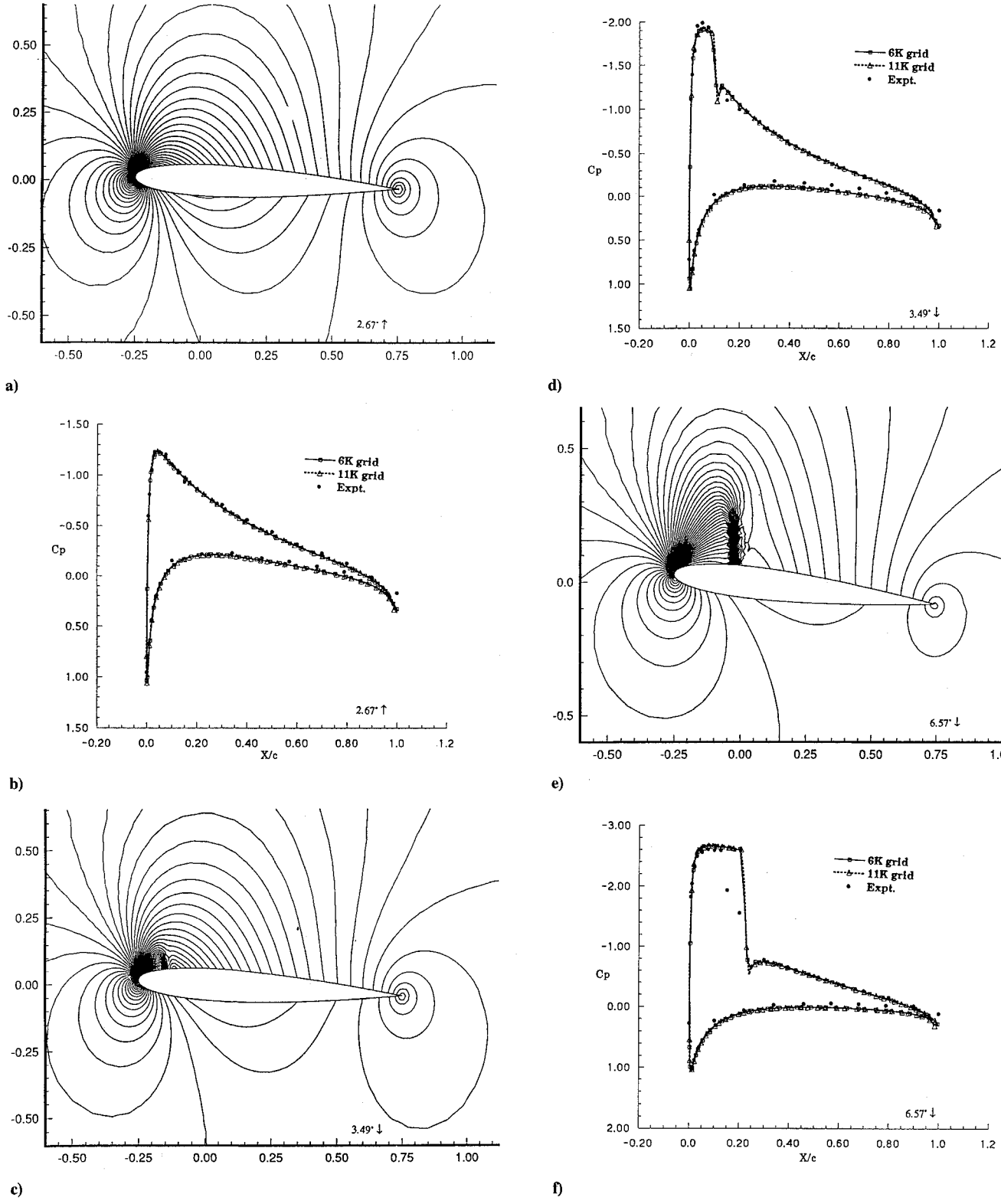


Fig. 4 Instantaneous pressure contours and surface pressure coefficient distributions for the sinusoidally oscillating airfoil.

be noted, however, the manner in which the store was ejected from its carriage position was not available, hence, a guessed value was used herein; an estimated ejector force was applied until the store dropped a specified distance, after which it was released to free fall. As shown in Fig. 3, three of the translational components and two of the rotational components compare very well. The trend of the third rotational component is also in agreement, however, the magnitudes differ by a maximum of 3 deg. The discrepancies in the results should be attributed to the possibly mismatched ejection characteristics.

Boundary Conditions

For steady inviscid flow, the velocity components used in the surface boundary conditions are written as

$$u_{\text{corrected}} = u - \eta_x U \tag{26a}$$

$$v_{\text{corrected}} = v - \eta_y U \tag{26b}$$

where U is the contravariant velocity given by

$$\boldsymbol{V} \cdot \hat{\boldsymbol{\eta}} = U = u\eta_x + v\eta_y \tag{27}$$

For unsteady moving boundary problems, however, the given conditions must be adjusted since the boundary faces now possess a discernible velocity. The expressions for the unsteady corrected velocity components remain the same as in Eq. (28) except for the velocity vector V, which now has to take into account the grid-speed term. The expression for the velocity vector is given by Eq. (4) and it is obtained by subtracting the grid speed from the respective components of the velocity vector V. For a moving-boundary problem, the pressure gradient in the normal direction is nonzero and can be derived from the normal momentum equation as

$$\frac{\partial p}{\partial n} = -\rho \hat{n} \cdot \boldsymbol{a} \tag{28}$$

where $\hat{n}$ is the unit normal to the boundary face and $\boldsymbol{a}$ is the acceleration of the body given by Eq. (22c). In the far field, characteristic boundary conditions are employed based on the locally one-dimensional Riemann invariants.

Computational Results

To validate the present dynamic solution algorithm and adaptive window procedure, a case with published experimental data[8] has been considered: a NACA 0012 airfoil sinusoidally oscillating about its quarter chord with a mean incidence of $\alpha_m = 4.86$ deg. For this oscillatory motion, the amplitude, reduced frequency, and freestream Mach number are $\alpha_0 = 2.44$ deg, $\omega = 0.0810$, and $M_\infty = 0.6$, respectively. To study the grid independence of the solution, a fine mesh with 6,084 cells and a finer mesh with 11,344 cells are generated. Both meshes cover the same amount of computational domain which ends at about 10 chords from the airfoil. Using a relatively small time step (2.0E-3 time units as normalized by the local speed of sound and the chord length), one cycle of the airfoil motion requires 64,642 steps and about 6 CPU h on a Cray Y-MP computer. Since, this time step is at least an order of magnitude smaller than those used in other computational studies reporting acceptable results (e.g., Refs. 1, 6, and 12), it is deemed appropriate for the present computations.

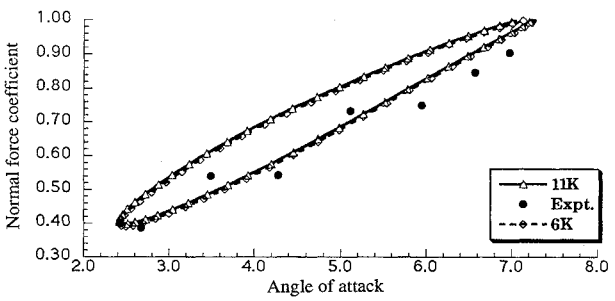


Fig. 5 Normal force cofficient vs angle of attack for the sinusoidally oscillating airfoil computed on 6-K and 11-K grids, and plotted in the counterclockwise direction.

Using Roe's flux-difference-splitting method, computations are performed until the periodic results are attained during the third cycle (limiting cycle). Presented in Fig. 4 are the pressure contours (fine-mesh results) and the surface pressure coefficient distributions (results from both meshes), in comparison with the experimental data, at three instants during the airfoil's motion. As the airfoil oscillates, a shock is formed on the upper surface which migrates toward the leading edge as the angle of attack is increased. As the angle of attack is decreasing, this shock migrates away from the leading edge, becoming nonexistent as the angle approaches the mean incidence and at all subsequent angles of attack below the mean incidence. Good agreement is observed between the computed and experimental results at all three positions. Small discrepancies over the first 5–10% of the airfoil's upper surface are believed to be related to the viscous effects, which are neglected during these computations. Finally, note that the fine-mesh and the finer-mesh solutions are almost identical. Hence, the fine-mesh solution is deemed grid independent for this case. Depicted in Fig. 5 is the variation of the normal force coefficient with the angle of attack, computed on all three meshes and in comparison with the data from Ref. 8. As expected in inviscid computations, the agreement with the data improves as the angle of attack decreases. Also, the grid-independence contention from Fig. 4 is fortified with Fig. 5 as the solutions on both levels of meshes are almost indistinguishable.

The next case considered in the present study also involves a prescribed motion of a NACA 0012 airfoil, but this time the body traverses a 3-DOF trajectory, streamwise translation of the center of gravity (quarter chord) and the angle of attack vary linearly with

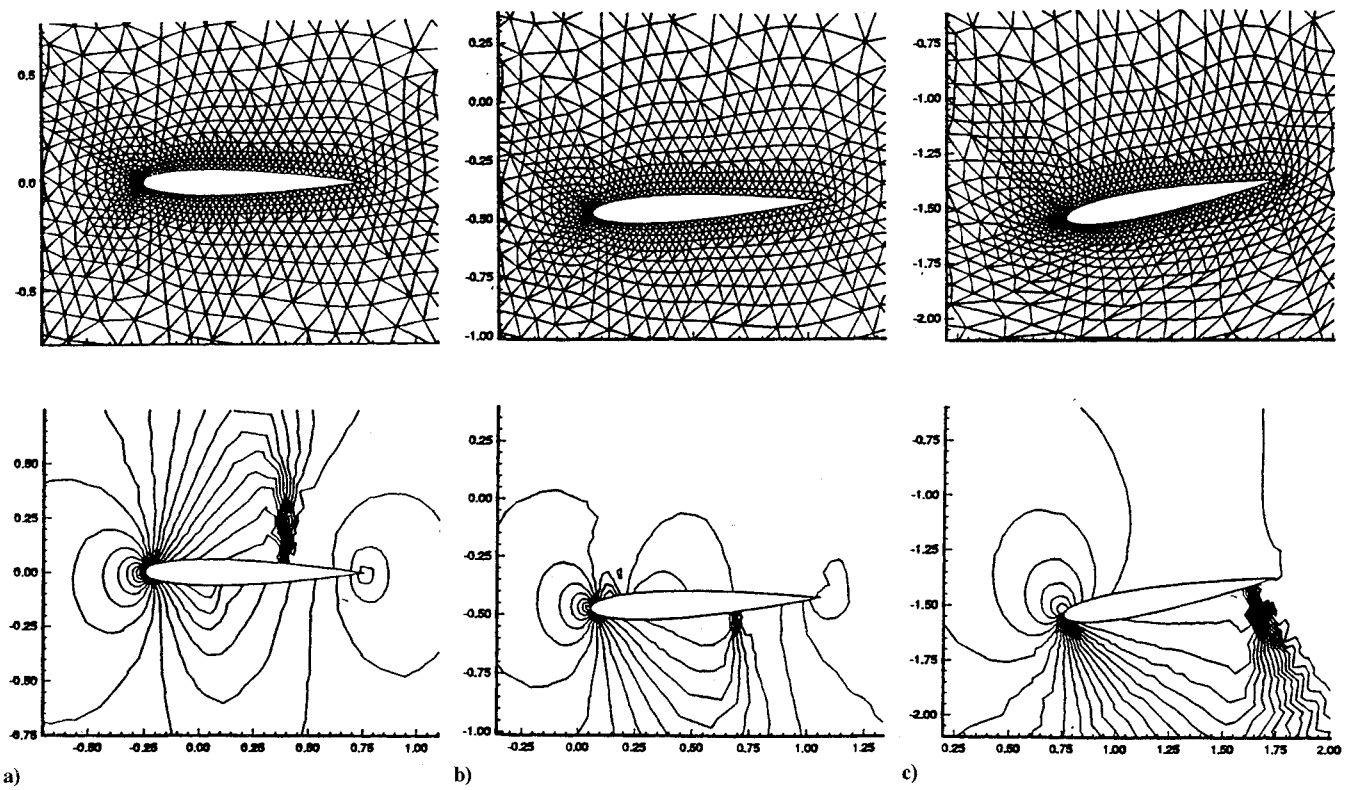


Fig. 6 Instantaneous grids and pressure contours for the airfoil in 3-DOF motion.

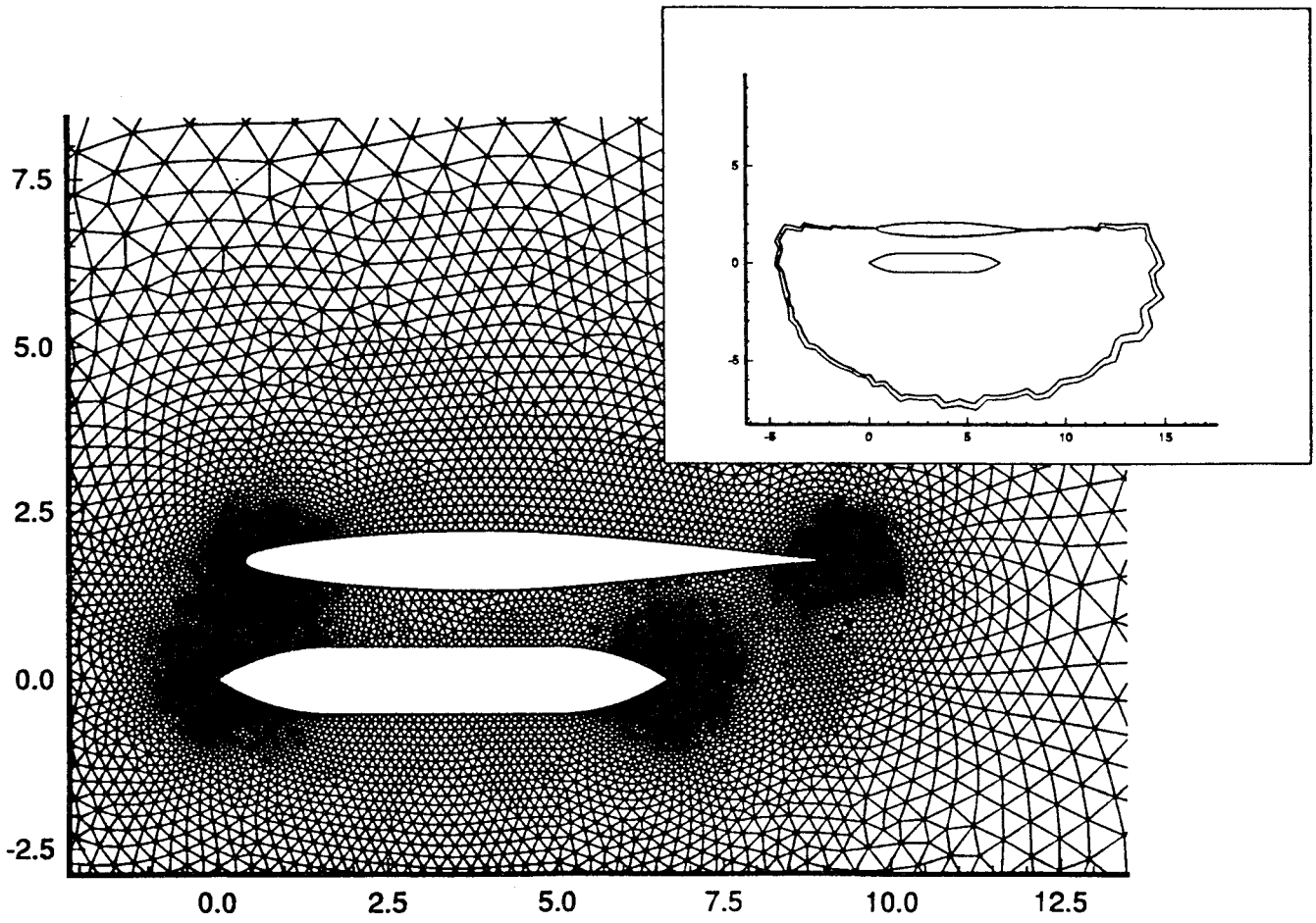


Fig. 7 Grid for the airfoil-store configuration; initial window for the aerodynamically determined motion shown as the inset.

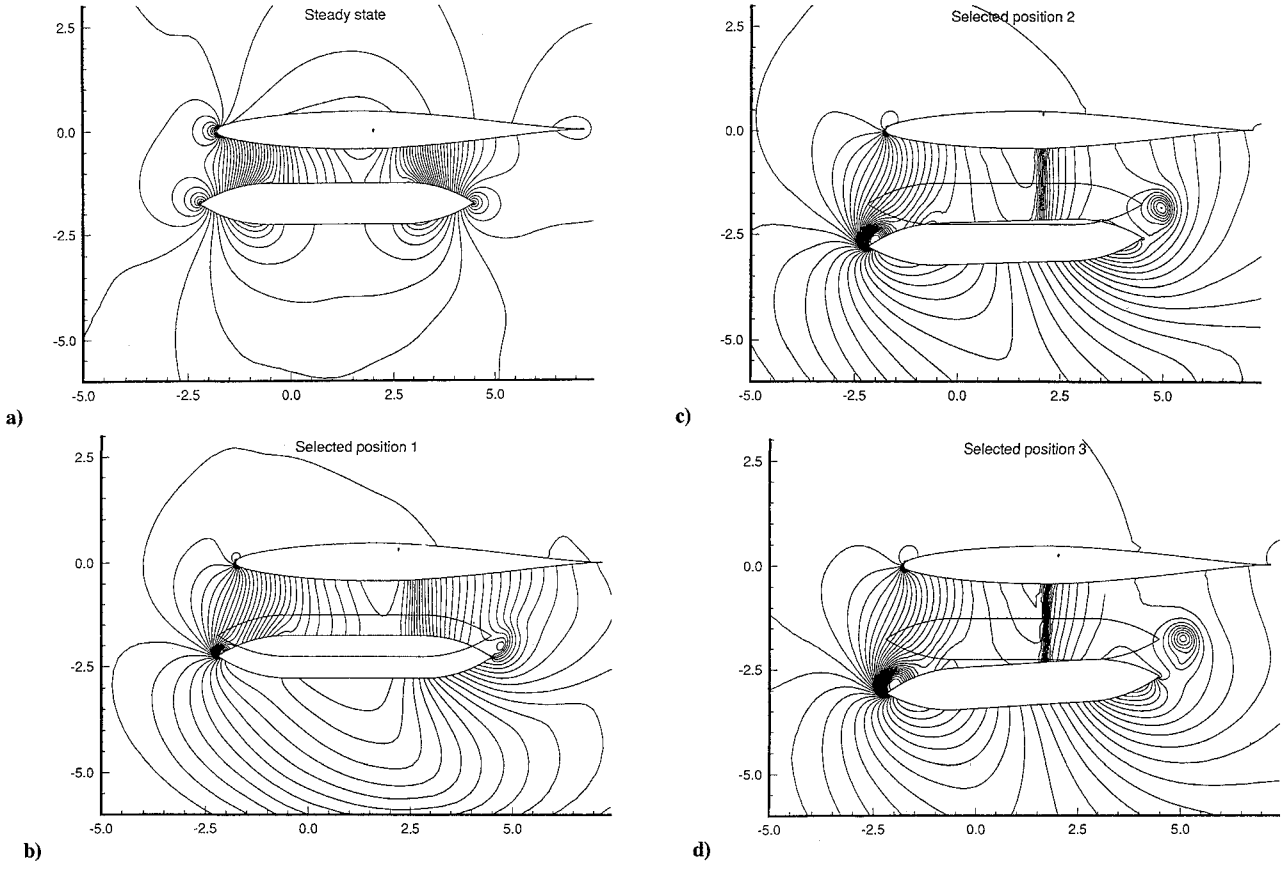


Fig. 8 Instantaneous pressure contours for the store separating from the airfoil.

time, whereas the vertical translation is a parabolic one in time. The freestream Mach number and the initial angle of attack are 0.8 and 1.25 deg, respectively.

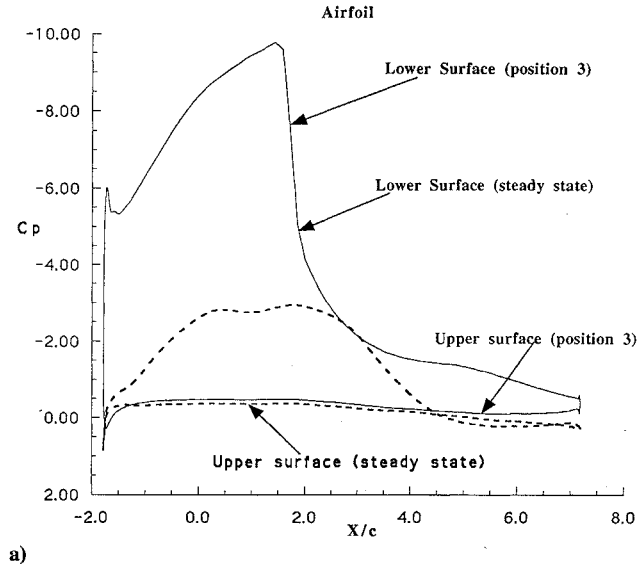
Three selected instants from the computed unsteady solution (Roe's flux difference splitting is used) are shown in Fig. 6. Note that, in order to zoom in on the near field at each instant, the abscissa and ordinate scales are different for these figures. As the airfoil begins to plunge down, the strong upper surface shock moves forward and the weak lower surface shock strengthens and moves aft. By the time the airfoil reaches third selected position, only a strong lower surface shock exists with a mild expansion on the upper surface. This case serves as a test of the robustness of the present solution algorithm and its adaptive window procedure for large displacements. The initial position has the center of gravity at the origin with a 1.25-deg angle of attack, whereas in position 3, the center of gravity is 1.5 chords below and 1.25 chords aft of the origin with a −15-deg angle of attack.

So far, the trajectory prediction method and the dynamic solution algorithm with the adaptive window procedure have been validated separately. In the final case, they are used together in achieving the primary objective of this research: determining the trajectory of released bodies in flight based on the computed aerodynamic loads as well as the gravitational force. Although the methodology has been developed for three-dimensional flows and for 6-DOF motions, only a two-dimensional test case that limits the trajectory to a 3-DOF motion has been chosen, mainly for computational efficiency; i.e., unsteady flow about an airfoil-store (AS) configuration, where the store has been released during a $M_\infty = 0.3$ flight. This configuration has been adapted from the three-dimensional wing-store separation problem detailed in Refs. 15 and 19. The wing section is a NACA 64A010 airfoil, and the store has an ogive-cylinder-ogive cross section. The computational domain size is 50 store diameters (all dimensions are normalized by the store diameter), and it contains 19,707 cells and 10,073 nodes. The grid and the initial adaptive window are shown in Fig. 7. Setting the computational time step (normalized using local speed of sound and the store diameter) of this case to 1.0E-3, 2.5 Cray Y-MP h are used to generate the sequence to be discussed next.

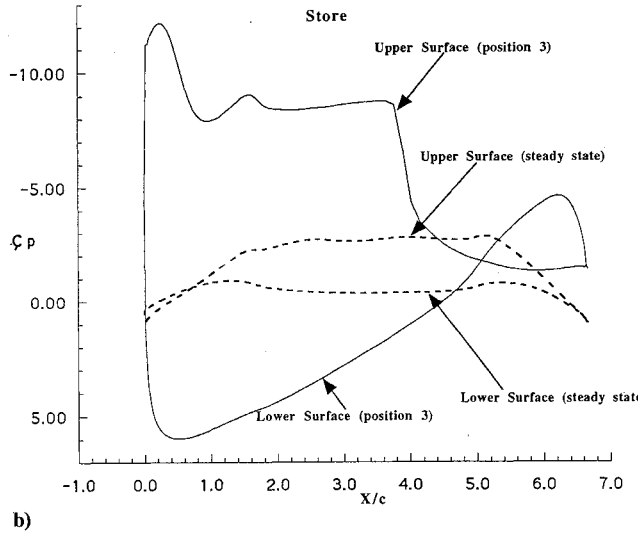
Depicted in Fig. 8 are the off-surface pressure contours for the initial position (steady-state flow of static AS) and three selected instants from the separation sequence. Position 1 displays the beginning of a compression region near the store's lower surface. This compression is caused by the moving-body-induced force and the subsequent flow. It should be noted that this simulation is two dimensional, which does not allow the lateral relieving effect of axisymmetric or three-dimensional flows. Hence, a nozzle-like flow behavior is observed between the airfoil and the store. At position 2, a strong compression is developed between the airfoil and the store. Furthermore, near the store's trailing edge, a vortex is formed. By the time the store drops to position 3, the compression region between the store and the wing develops into a strong normal shock, the trailing-edge vortex is enlarged and moved downstream, and the store-induced downward flow is strengthened.

Displayed in Fig. 9 are the pressure coefficient distributions on the airfoil and the store surfaces. To compare the changes in the pressure distributions as a result of the separation, initial steady-state distributions are superimposed on the instantaneous distribution at position 3. As expected, there is little change on the upper airfoil surface. However, the lower airfoil surface now indicates an expansion terminated by a shock. The upper store surface also displays the effect of the shock and the following compression. The lower store surface experiences a significant distribution change in time; from a predominantly constant distribution, it transforms to a large compression at the leading edge followed by an expansion extending to the trailing edge. The 3-DOF trajectory of this separation sequence is plotted in Fig. 10. The translational motion is primarily downward (y direction) with a small streamwise displacement (x direction). The rotational motion (θ direction) is an initially gradual, but then significantly increasing, pitch-down motion.

It should be noted that unsteady experimental data do not exist for this configuration, hence, such a comparison is not possible. As mentioned earlier, the experimental data provided in Ref. 15 is for



a)



b)

Fig. 9 Surface pressure coefficient distributions for the airfoil-store configuration.

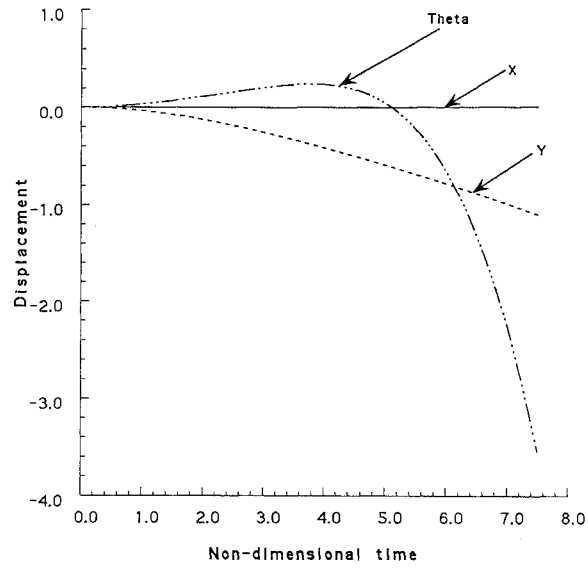


Fig. 10 Store separating from the airfoil, 3-DOF trajectory.

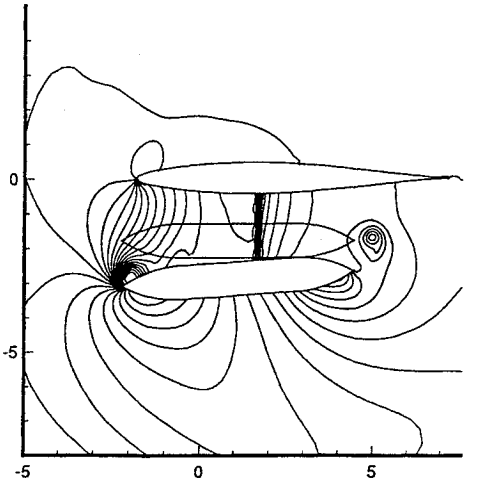


Fig. 11 Instantaneous pressure contours for the store separating from the airfoil obtained using Roe's scheme (position 3).

a quasisteady separation. In addition, the present computations are only two dimensional. However, in an attempt to check the preceding solution, this store-separation simulation is obtained once using van Leer's flux-vector-splitting (Figs. 8–10), then again using Roe's flux-difference-splitting method (Fig. 11). Although the variable between the two simulations is only a portion of the overall methodology, obtaining identical results from these two simulations is deemed as an important step toward a more thorough validation.

Concluding Remarks

A new dynamic, unstructured-mesh methodology was developed to simulate unsteady flows about single or multiple objects in relative motion and to compute the trajectory of their motion as determined by the aerodynamic loads. For this purpose, the method coupled the governing equations of the fluid dynamics and rigid body dynamics. The grid adaptation was performed within windows put around the moving boundaries to improve the computational efficiency. However, for very large amplitude motions, the window adaptation would require highly dense meshes for the major portion of the domain, otherwise a partial remeshing procedure would be necessary. This idea is currently under investigation.

Using the experimentally determined force and moment fields, the present trajectory code was used to compute the 6-DOF trajectory of a store separating from a wing. The results compared well with the trajectory used in the wind-tunnel tests. The flow past a sinusoidally oscillatory airfoil was computed and compared with the experimental data. The minor discrepancies were attributed to the viscous effects. Then, the same NACA 0012 airfoil was given a 3-DOF motion and immersed in a transonic freestream flow. This case demonstrated the robustness of the adaptive window procedure for large displacements as well as the ability of the flow solver to capture strong moving shock waves.

Finally, the overall methodology was demonstrated through a two-dimensional example: the carriage, separation, and the free-fall of a store from a wing section (airfoil). The motion and its trajectory were entirely determined by the instantaneous aerodynamic loads provided by the unsteady flowfield computations and the force of gravity. Unlike the quasisteady approaches, the present approach also captured the boundary-induced flow component, and the time-dependent, dynamic interference between a stationary body and a moving body. Thus, this study suggests that the present dynamic, unstructured-mesh method is a viable alternative to the dynamic, overlapped structured-grids approaches.[1–3]

Acknowledgments

The major support for this work was from NASA Langley Research Center, Grant NAG-1-1150, technical monitor D.S. Miller. It was also partially supported by The Armament Directorate of USAF Wright Laboratories, technical monitor L.B. Simpson. The authors would like to thank N.T. Frink and S. Pirzadeh for their help.

References

[1]Baysal, O., and Yen, G. W., "Kinematic Domain Decomposition To Simulate Flows Past Moving Bodies," AIAA Paper 91-0725, Jan. 1991.

[2]Yen, G. W, and Baysal, O., "Computing Unsteady High Speed Flows Past an Oscillating Cylinder Near a Vertical Wall, " *Journal of Spacecraft and Rockets*, Vol. 31, No. 4, 1994, pp. 630–635; also AIAA Paper 92-4653, Aug. 1992.

[3]Meakin, R. L., "Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations," AIAA Paper 93-3350, July 1993.

[4]Arabshahi, S., and Whitfield, D. L., "A Multiblock Approach for Solving the Three-Dimensional Unsteady Euler Equations about a Wing-Pylon-Store Configuration," AIAA Paper 89-3401, Aug. 1989.

[5]Trepanier, J. Y., Reggio, M., Paraschivoiu, M., and Camarero, R., "Unsteady Euler Solutions for Arbitrarily Moving Bodies and Boundaries," *AIAA Journal*, Vol. 31, No. 10, 1993, pp. 1869–1876.

[6]Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," AIAA Paper 89-0115, Jan. 1989.

[7]Lohner, R., "Adaptive Remeshing for Transient Problems with Moving Bodies," AIAA Paper 88-3736-CP, July 1988.

[8]Landon, R., "NACA 0012 Oscillatory and Transient Pitching," *Compendium of Unsteady Aerodynamic Measurements*, AGARD Rept. 702, Aug. 1982, pp. 3.3–3.25.

[9]Frink, N. T., "Upwind Scheme For Solving the Euler Equations on Unstructured Tetrahedral Meshes," *AIAA Journal*, Vol. 30, No. 1, 1992, pp. 70–77.

[10]Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372.

[11]Van Leer, B., "Flux Vector Splitting for the Euler Equations," *Lecture Notes in Physics*, Vol. 170, Springer-Verlag, Berlin, 1982.

[12]Anderson, W. K., Thomas, J. L., Rumsey, C. L., "Extension of Applications of Flux-Vector Splitting to Unsteady Calculations on Dynamic Meshes," AIAA Paper 87-1152, June 1987; also, "Parametric Study of Grid Size, Time Step and Turbulence Modeling on Navier-Stokes Computations Over Airfoils," *Validation of Computational Fluid Dynamics*, AGARD CP-437, Lisbon, Portugal, 1988, pp. 5.1–5.19.

[13]Thomas, P. D., and Lombard, C. K., "Geometric Conservation Law and its Application to Flow Computations on Moving Grids," *AIAA Journal*, Vol. 17, No. 10, 1979, pp. 1030–1037.

[14]Tamura, Y., and Fujii, K., "Conservation Law for Moving and Transformed Grids," AIAA Paper 93-3365, July 1993.

[15]Heim, E. R., "CFD Wing/Pylon/Finned Store Mutual Interference Wind Tunnel Experiment," Arnold Engineering and Development Center, AEDC-TSR-91-P4, Tullahoma, TN, Jan. 1991.

[16]Chakravarthy, S. R., and Szema, K.-Y., "Computational Fluid Dynamics Capability For Internally Carried Store Separation," Rockwell International Science Center, Tech. Rept. SC71039, Thousand Oaks, CA, Nov. 1991.

[17]Rausch, R. D., Batina, J. T., and Yang, T. Y., "Spatial Adaptation Procedures on Unstructured Meshes for Accurate Unsteady Aerodynamic Flow Computations," NASA TM 104039, March 1991.

[18]D'Souza, A. F., and Garg, V. K., *Advanced Dynamics—Modeling and Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1984, pp. 95–155.

[19]Newman, J. C., III, and Baysal, O., "Transonic Solutions of a Wing/Pylon/Finned Store Using Hybrid Domain Decomposition," AIAA Paper 92-4571, Aug. 1992.