Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes

John T. Batina*
NASA Langley Research Center, Hampton, Virginia 23665

Two algorithms for the solution of the time-dependent Euler equations are presented for unsteady aerodynamic analysis of oscillating airfoils. Both algorithms were developed for use on an unstructured grid made up of triangles. The first flow solver involves a Runge-Kutta time-stepping scheme with a finite-volume spatial discretization that reduces to central differencing on a rectangular mesh. The second flow solver involves a modified Euler time-integration scheme with an upwind-biased spatial discretization based on the flux-vector splitting of Van Leer. A significant aspect of the research is the implementation of a dynamic mesh algorithm that is employed for problems where the airfoil moves. Steady and unsteady results are presented for the NACA 0012 airfoil to demonstrate application of the Euler solvers and dynamic mesh algorithm. The unsteady flow results were obtained for the airfoil pitching harmonically about the quarter chord. The two sets of calculated instantaneous pressure distributions and lift and moment coefficients during a cycle of motion compare well with each other and with the experimental data. The paper presents descriptions of the Euler solvers and dynamic mesh algorithm along with results which assess the capability.

Introduction

CONSIDERABLE progress has been made over the past two decades on developing computational fluid dynamics (CFD) methods for aerodynamic analysis.[1] Recent work in CFD has focused primarily on developing algorithms for the solution of the Euler and Navier-Stokes equations. Also, Edwards and Thomas[2] have recently reviewed the significant advances that have been made in the development of computational methods for predicting unsteady flows. This research has been highly focused on developing computer codes for the solution of the transonic small disturbance and full-potential equations, although efforts are currently underway at the higher equation levels as well. For unsteady aerodynamic and aeroelastic analysis with body-fitted meshes, these methods generally require that the mesh move to conform to the instantaneous position of the moving or deforming body under consideration. Many of the methods that are currently being developed assume that the mesh moves rigidly or that the mesh shears as the body deforms. The assumptions consequently limit the applicability of the procedures to rigid-body or small-amplitude motions. Furthermore, these methods of solution typically assume that the computational grid has an underlying geometrical structure. As an alternative, algorithms which make use of unstructured grids have been developed recently.[3-9] In two dimensions these grids are typically made up of triangles, and in three dimensions they consist of an assemblage of tetrahedra. The unstructured grid methods have the distinct advantage over structured grid methods in that they can easily treat the most complex of geometric configurations as well as flow conditions. For example, an inviscid transonic flow solution was presented by Jameson et al.[5] for the Boeing 747 aircraft, which was computed using an unstructured tetrahedral grid. In this calculation, the complete vehicle geometry was modeled including multiple lifting surfaces, the fuselage, underwing pylons, and flow-through engine nacelles. Furthermore, the unstructured grid-flow solvers easily lend themselves to solution-adaptive procedures where the grid is adapted to gradients of the solution to more accurately resolve the flow physics. As discussed by Lohner,[7] this may be accomplished using mesh movement, mesh enrichment, or remeshing, all of which pose no problems for most unstructured grid-flow solvers. In contrast, a structured grid-flow solver becomes relatively much more complicated when refinement procedures, such as embedded grids, are utilized.

Two algorithms for the solution of the time-dependent Euler equations, based on the use of unstructured grids, are described in the present paper. The algorithms were developed for the unsteady aerodynamic analysis of oscillating airfoils using unstructured grids made up of triangles. The main objective of the study was to investigate the applicability of the unstructured grid methodology for time-dependent problems with moving boundaries. The first flow solver that was developed involves a Runge-Kutta time-stepping scheme with a finite-volume spatial discretization that reduces to central differencing on a rectangular mesh. The solver uses explicitly added artificial dissipation to prevent oscillations near shock waves and to damp high-frequency uncoupled error modes. In Refs. 3 and 4, these procedures, including a multigrid method, were applied to steady flows about single-element and multi-element airfoils using unstructured triangular meshes. With the central-difference-type flow solver, shock waves are captured usually with two or three interior grid points. The second flow solver that was developed involves a modified Euler time-integration scheme with an upwind-biased spatial discretization based on the method of flux-vector splitting developed by Van Leer.[10] A distinct advantage of the upwind method is that it is naturally dissipative so that explicit artificial dissipation is not required. On structured grids with the upwind-difference-type flow solver, shock waves are captured with at most one interior grid point, as demonstrated by Anderson et al.[11] The second flow solver of the present study was therefore formulated using the procedures of Ref. 11, extended for application on unstructured meshes. Also, the flux splittings of the present work include the additional terms that are necessary for use on moving meshes, and they retain all of the properties of the original splitting.[12,13] A significant aspect of the present research is the implementation of a dynamic mesh algorithm that is employed for problems where

Received Jan. 1989; revision received Aug. 1989.

*Senior Research Scientist, Unsteady Aerodynamics Branch, Structural Dynamics Division. Senior Member AIAA.

the airfoil moves. The method is completely general in that it can treat realistic airfoil motions including aeroelastic transient-type motion. Steady and unsteady results are presented for the NACA 0012 airfoil to demonstrate application of the Euler solvers and dynamic mesh algorithm. The unsteady flow results were obtained for the airfoil pitching harmonically about the quarter chord. The paper presents descriptions of the Euler solvers and dynamic mesh algorithm along with results and comparisons which assess the capability.

Euler Equations

In the present study, the flow is assumed to be governed by the two-dimensional time-dependent Euler equations which may be written in integral form as

$$\frac{\partial}{\partial t}\iint_{\Omega} Q\,\mathrm{d}x\,\mathrm{d}y + \int_{\partial\Omega}(F\,\mathrm{d}y - G\,\mathrm{d}x) = 0 \qquad (1)$$

where the vector of conserved variables Q and the convective fluxes F and G are given by

$$Q = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{Bmatrix}, \quad F = \begin{Bmatrix} \rho U \\ \rho U u + p \\ \rho U v \\ (e+p)U + x_t p \end{Bmatrix}, \quad G = \begin{Bmatrix} \rho V \\ \rho V u \\ \rho V v + p \\ (e+p)V + y_t p \end{Bmatrix} \qquad (2)$$

The contravariant velocities U and V are defined by

$$U = u - x_t \qquad V = v - y_t \qquad (3)$$

where x_t and y_t are the grid speeds in the x and y directions, respectively, and the pressure p is given by the equation of state for a perfect gas

$$p = (\gamma - 1)[e - {\textstyle\frac{1}{2}}\rho(u^2 + v^2)] \qquad (4)$$

The preceding equations have been nondimensionalized by the freestream density ρ_∞ and the freestream speed of sound a_∞.

Central-Difference Solution Algorithm

In this section an algorithm is presented for the solution of the two-dimensional time-dependent Euler equations, which reduces conceptually to central differencing on a rectangular mesh. Thus, the algorithm is referred to as a central-difference-type flow solver, as described below.

Spatial Descretization

The Euler equations in integral form [Eq. (1)] are solved using a finite-volume algorithm that was developed for analysis with an unstructured grid made up of triangles. The algorithm is a nodal scheme whereby the flow variables are stored at the vertices of the triangles. As shown in Fig. 1 by the dashed lines, the control volume at a given node i is taken to be the neighboring triangles, which have a vertex at that node. The boundary integral in Eq. (1) is approximated as follows. First, along each edge of the control volume boundary the fluxes at the two end points are averaged. Then, the contribution to the boundary integral is determined by taking the product of the averaged fluxes with the directed length of the edge. Along edge j-k in Fig. 1, for example, the contribution to the boundary integral in Eq. (1) is given by

$$\tfrac{1}{2}(F_j + F_k)(y_k - y_j) - \tfrac{1}{2}(G_j + G_k)(x_k - x_j)$$

Artificial Dissipation

The unsteady Euler equations are a set of nondissipative hyperbolic conservation laws whose numerical solution requires some form of artificial dissipation to prevent oscillations near shock waves and to damp high frequency uncoupled error modes. On structured meshes, an adaptive blend of second and fourth differences of the conserved variables has been shown to be an effective form of dissipation.[14] In the present unstructured mesh Euler solver, a combination of harmonic and biharmonic operators is employed corresponding to second and fourth differences, respectively. Similar to that of Jameson and Mavriplis,[3] Mavriplis,[4] and Jameson et al.,[5] the biharmonic operator provides a background dissipation to damp high frequency errors, and the harmonic operator prevents oscillations near shocks. The harmonic operator is multiplied by a pressure switch, which is first-order accurate near shocks and is second-order accurate in smooth regions of the flow. The biharmonic operator is third-order accurate and is adaptively turned off to prevent overshoots in regions of shock waves.

Time Integration

The Euler equations are integrated in time by assuming that the conserved variables represented by Q are constant within a control volume which yields

$$\frac{\mathrm{d}}{\mathrm{d}t}(A_i Q_i) + C(Q) - D(Q) = 0 \qquad (5)$$

where C and D are the convective and dissipative operators, respectively, and A_i is the area of the control volume surrounding node i. These equations are integrated in time using an explicit four-stage, Runge-Kutta time-stepping scheme given by

$$\begin{aligned}
Q^{(0)} &= Q^n \\
Q^{(1)} &= \frac{A^n}{A^{n+1}}Q^{(0)} - \frac{1}{4}\frac{\Delta t}{A^{n+1}}[C(Q^{(0)}) - D(Q^{(0)})] \\
Q^{(2)} &= \frac{A^n}{A^{n+1}}Q^{(0)} - \frac{1}{3}\frac{\Delta t}{A^{n+1}}[C(Q^{(1)}) - D(Q^{(0)})] \\
Q^{(3)} &= \frac{A^n}{A^{n+1}}Q^{(0)} - \frac{1}{2}\frac{\Delta t}{A^{n+1}}[C(Q^{(2)}) - D(Q^{(0)})] \\
Q^{(4)} &= \frac{A^n}{A^{n+1}}Q^{(0)} - \frac{\Delta t}{A^{n+1}}[C(Q^{(3)}) - D(Q^{(0)})] \\
Q^{n+1} &= Q^{(4)}
\end{aligned} \qquad (6)$$

In this scheme, the convective operator is evaluated at each stage and, for computational efficiency, the dissipative operator is evaluated only at the first stage. The Runge-Kutta

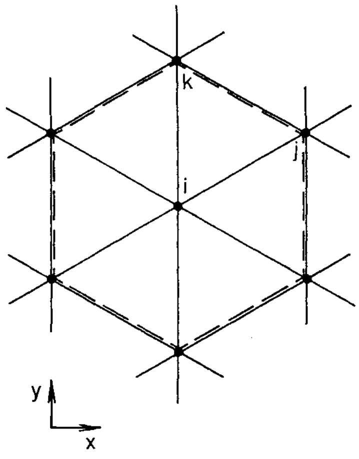


Fig. 1 Definition of the control volume in the central-difference solution algorithm.

scheme represented by Eqs. (6) is second-order accurate in time and includes the necessary terms to account for changes in cell areas due to a moving or deforming mesh.

Implicit Residual Smoothing

The explicit time-integration scheme described in the preceding section has a step size that is limited by the Courant-Friedricks-Lewy (CFL) condition corresponding to a CFL number of $2\sqrt{2}$. To accelerate convergence to steady-state, the CFL number may be increased by averaging the residual R_i with values at neighboring grid points. This is accomplished by replacing R_i by the smoothed residual $\bar{R}_i$ given by

$$\bar{R}_i - \epsilon \nabla^2 \bar{R}_i = R_i \tag{7}$$

where ϵ is a constant selected typically in the range between zero and one which controls the amount of smoothing and ∇^2 is an undivided Laplacian operator. These implicit equations are solved approximately using several Jacobi iterations similar to that which is done in Refs. 3 and 4. Convergence to steady state is further accelerated using enthalpy damping[14] and local time stepping. The local time stepping uses the maximum allowable step size at each grid point as determined by a local stability analysis. For unsteady applications, however, a global time step must be used because of the time-accuracy requirement. The maximum allowable global time step may be increased to a value that is larger than that dictated by the CFL condition by using a time-accurate version of Eq. (7) similar to that of Ref. 15. In this procedure the constant ϵ in Eq. (7) is replaced by a parameter that varies from grid point to grid point, which is defined by

$$\epsilon = \max\left[\frac{1}{4}\left(\frac{\Delta t^2}{\Delta t^2_{\rm CFL}} - 1\right), 0.0\right] \tag{8}$$

In Eq. (8), Δt is the time step taken, and $\Delta t_{\rm CFL}$ is the locally allowable time step for the four-stage, Runge-Kutta time-stepping scheme.

Upwind-Difference Solution Algorithm

In this section a second algorithm is presented for the solution of the Euler equations that makes use of upwind differencing and flux-vector splitting similar to schemes developed for use on structured meshes. Thus, the algorithm is referred to as an upwind-difference-type flow solver, as described below.

Flux-Vector Splitting

Similar to the central-difference solution algorithm, the upwind method solves the Euler equations in integral form [Eq. (1)] based on a finite-volume discretization on an unstructured grid made up of triangles. The upwind algorithm is also a nodal scheme, which has a control volume at a given point i as defined by the dashed lines in Fig. 2. The edges of the control volume are constructed within each triangle, which has a vertex at point i by connecting the centroid of the triangle with the midpoints of the two edges which share point i. In general, however, the line connecting the centroids of adjacent triangles is not straight. The boundary integral of Eq. (1) is approximated by using the flux-vector splitting of Van Leer.[10] In this method the flux vectors are split into forward and backward contributions, which are continuously differentiable even at sonic and stagnation points. The scheme is derived as follows. For an edge of the control volume a-b shown in Fig. 2, the fluxes are first rotated into a locally Cartesian coordinate system $\bar{x}$-$\bar{y}$ with the principal direction being perpendicular to the edge. The flux in this direction is defined as

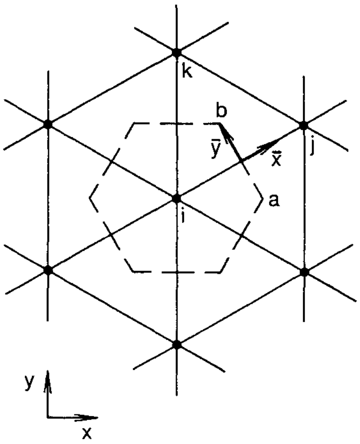


Fig. 2 Definition of the control volume in the upwind-difference solution algorithm.

$$H\Delta s = T(F\Delta y - G\Delta x) = \begin{Bmatrix} \rho \bar{U} \\ \rho \bar{U}\bar{u} + p \\ \rho \bar{U}\bar{v} \\ e\bar{U} + p\bar{u} \end{Bmatrix} \Delta s \tag{9}$$

where the transformation matrix T is given by

$$T = \frac{1}{\Delta s}\begin{bmatrix} \Delta s & 0 & 0 & 0 \\ 0 & \Delta y & -\Delta x & 0 \\ 0 & \Delta x & \Delta y & 0 \\ 0 & 0 & 0 & \Delta s \end{bmatrix} \tag{10}$$

In Eqs. (9) and (10), Δx and Δy are the directed lengths of edge a-b in the x and y coordinate directions, respectively, and $\Delta s^2 = \Delta x^2 + \Delta y^2$. Also, $\bar{u}$ and $\bar{v}$ are the Cartesian velocity components perpendicular and parallel to the edge defined by

$$\bar{u} = u\frac{\Delta y}{\Delta s} - v\frac{\Delta x}{\Delta s} \tag{11a}$$

$$\bar{v} = u\frac{\Delta x}{\Delta s} - v\frac{\Delta y}{\Delta s} \tag{11b}$$

and $\bar{U}$ and $\bar{V}$ are the corresponding contravariant velocities

$$\bar{U} = (u - x_t)\frac{\Delta y}{\Delta s} - (v - y_t)\frac{\Delta x}{\Delta s} \tag{12a}$$

$$\bar{V} = (u - x_t)\frac{\Delta x}{\Delta s} + (v - y_t)\frac{\Delta y}{\Delta s} \tag{12b}$$

The flux vector H is split in a one-dimensional fashion into forward (H^+) and backward (H^-) vectors for $|\bar{U}| \leq a$ as

$$H = H^+ + H^- \tag{13}$$

where

$$H^{\pm} = \begin{Bmatrix} h^{\pm}_{\rm mass} \\ h^{\pm}_{\rm mass} \quad [(-\bar{U} \pm 2a)/\gamma + \bar{u}] \\ h^{\pm}_{\rm mass} \quad \bar{v} \\ h^{\pm}_{\rm energy} \end{Bmatrix} \tag{14}$$

and

$$h^{\pm}_{\rm mass} = \pm\frac{\rho}{4a}(\bar{U} \pm a)^2 \tag{15a}$$

$$h^{\pm}_{\text{energy}} = h^{\pm}_{\text{mass}}\left[\frac{-(\gamma-1)\bar{U}^2 \pm 2(\gamma-1)\bar{U}a + 2a^2}{\gamma^2-1}\right.$$
$$\left. + \frac{\bar{u}^2+\bar{v}^2}{2} + \left(x_t\frac{\Delta y}{\Delta s} - y_t\frac{\Delta x}{\Delta s}\right)(-\bar{U} \pm 2a)/\gamma\right] \qquad (15b)$$

The resulting split fluxes are finally rotated back into the original coordinate system so that

$$F\Delta y - G\Delta x = T^{-1}[H^+(Q^-) + H^-(Q^+)] \qquad (16)$$

where $H^+(Q^-)$ and $H^-(Q^+)$ denote the fluxes $H^{\pm}$ evaluated using upwind-biased interpolations of the conserved variables given by

$$Q^- = Q_i + ½[(1-\kappa)(\Delta_-)_i + \kappa(\Delta_+)_i] \qquad (17)$$

$$Q^+ = Q_j + ½[(1-\kappa)(\Delta_-)_j + \kappa(\Delta_+)_j] \qquad (18)$$

where

$$(\Delta_-)_i = Q_{x_i}(x_j - x_i) + Q_{y_i}(y_j - y_i) \qquad (19a)$$

$$(\Delta_+)_i = Q_j - Q_i \qquad (19b)$$

$$(\Delta_-)_j = Q_{x_j}(x_i - x_j) + Q_{y_j}(y_i - y_j) \qquad (20a)$$

$$(\Delta_+)_j = Q_i - Q_j \qquad (20b)$$

In Eqs. (19a) and (20a), Q_{x_i}, Q_{y_i}, Q_{x_j}, and Q_{y_j} are computed by applying Green's theorem and taking the value of Q along an edge of the control volume to be the average of the endpoint values. For example, the contribution to Q_{x_i} along edge j-k in Fig. 2 is given by

$$\frac{1}{2A_i}(Q_j + Q_k)(y_k - y_j)$$

Also, the parameter κ controls a family of difference schemes by appropriately weighting Δ_- and Δ_+. On structured meshes it is easy to show that $\kappa = -1$ yields a fully upwind scheme, $\kappa = 0$ yields Fromm's scheme, and $\kappa = 1$ yields central differencing. The value $\kappa = ⅓$ leads to a third-order-accurate, upwind-biased scheme, although the third-order accuracy is strictly correct only for one-dimensional calculations.[11] Nevertheless, $\kappa = ⅓$ was used in the upwind-difference calculations presented herein. Also, in calculations involving upwind-biased schemes, oscillations in the solution near shock waves are expected to occur. To eliminate these oscillations flux limiting is usually required. In the present study, the continuously differentiable flux limiter described in Ref. 11 was used to modify the upwind-biased interpolations for Q^- and Q^+ given by Eqs. (17) and (18).

Time Integration

The Euler equations are integrated in time using a two-step-modified, Euler time-integration scheme given by

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

$$Q^{(1)} = \frac{A^n}{A^{n+1}}Q^{(0)} - \frac{1}{2}\frac{\Delta t}{A^{n+1}}T^{-1}[H^+(Q^{-(0)}) + H^-(Q^{+(0)})]$$

$$Q^{(2)} = \frac{A^n}{A^{n+1}}Q^{(0)} - \frac{\Delta t}{A^{n+1}}T^{-1}[H^+(Q^{-(1)}) + H^-(Q^{+(1)})]$$

$$Q^{n+1} = Q^{(2)} \qquad (21)$$

Explicit artificial dissipation is not added to the equations since the upwind-biased spatial differencing is naturally dissipative. Also, similar to the Runge-Kutta time-stepping of Eqs. (6), the scheme represented by Eqs. (21) is second-order accurate in time and includes the necessary terms to account for changes in cell areas due to a moving or deforming mesh. The Runge-Kutta scheme was not used to time integrate the Euler equations for the upwind difference solution algorithm since it was shown by Turkel and Van Leer[16] that the allowable CFL number is reduced from $2\sqrt{2}$ to approximately 1.4 when using Fromm's scheme. It was therefore concluded that upwind differencing is not efficient when used in combination with Runge-Kutta time stepping.[16] The two-step modified Euler scheme was chosen as an alternative to the four-stage Runge-Kutta scheme since it requires half of the work per time step, and calculations were found to be numerically stable for a CFL number of 1.8. To accelerate convergence to steady slate, local time stepping was used similar to that in the central-difference algorithm. Enthalpy damping was not used since it is inappropriate when upwind differencing is employed.

Boundary Conditions

To impose the no-flow-through boundary condition along the surface of the body, the boundary integral of Eq. (1) is first rewritten in terms of the flux velocity defined by $\bar{U}\Delta y - \bar{V}\Delta x$. Then, for control volumes with edges along the body surface, the flux velocity along these edges, which is proportional to the velocity normal to the edge, is set equal to zero.

In the far field, a characteristic analysis based on Riemann invariants, similar to that used in Refs. 3, 4, and 14, is used to determine the values of the flow variables on the outer boundary of the grid. This analysis correctly accounts for wave propagation in the far field, which is important for rapid convergence to steady state and serves as a "nonreflecting" boundary condition for unsteady applications.

Dynamic Mesh Algorithm

A significant aspect of the present research is the implementation of a dynamic mesh algorithm that is employed for problems where the body moves. The mesh is moved to conform to the instantaneous position of the body by modeling each edge of each triangle by a spring. The spring stiffness for a given edge i-j is taken to be inversely proportional to the length of the edge as

$$k_m = 1/[(x_j - x_i)^2 + (y_j - y_i)^2]^{½} \qquad (22)$$

Grid points on the outer boundary of the mesh are held fixed, and the instantaneous locations of the points on the inner boundary (body) are prescribed. At each time step, the static equilibrium equations in the x and y directions are solved iteratively at each interior node i of the grid for the displacements δ_{x_i} and δ_{y_i}. This is accomplished by using a predictor-corrector procedure, which first predicts the displacements according to a linear extrapolation given by

$$\tilde{\delta}_{x_i} = 2\delta^n_{x_i} - \delta^{n-1}_{x_i} \qquad \tilde{\delta}_{y_i} = 2\delta^n_{y_i} - \delta^{n-1}_{y_i} \qquad (23)$$

and then corrects these displacements using several Jacobi iterations of the static equilibrium equations using

$$\delta^{n+1}_{x_i} = \frac{\Sigma k_m \tilde{\delta}_{x_m}}{\Sigma k_m} \qquad \delta^{n+1}_{y_i} = \frac{\Sigma k_m \tilde{\delta}_{y_m}}{\Sigma k_m} \qquad (24)$$

In Eqs. (24) the summations are performed over all edges of the triangles that have node i as an end point. The new locations of the interior nodes are then determined by

$$x^{n+1}_i = x^n_i + \delta^{n+1}_{x_i} \qquad y^{n+1}_i = y^n_i + \delta^{n+1}_{y_i} \qquad (25)$$

The predictor-corrector procedure has been found to be more efficient than simply performing Jacobi iterations because far fewer iterations are required to achieve acceptable convergence. In practice, it has been found that only one or two iterations are sufficient to move the mesh accurately.

To demonstrate mesh movement using the dynamic mesh algorithm, consider the coarse grid about a NACA 0012 airfoil that is shown in part a of Fig. 3. The grid, which contains 288 nodes, was generated using the Delaunay triangulation method of Ref. 4. It is used only to illustrate how the mesh moves. In this example the airfoil was pitched about the quarter chord for one cycle of sinusoidal motion with an amplitude of 15 deg. As mentioned, the grid points on the outer boundary of the mesh are held fixed. The points on the airfoil are also fixed relative to the airfoil. The mesh at the maximum pitch angle of 15 deg is shown in part b of Fig. 3, and the mesh at the minimum pitch angle of -15 deg is shown in part c of Fig. 3. The mesh moves smoothly as the airfoil pitches, and the procedure is completely general in that it can treat realistic airfoil motions including aeroelastic transient-type motion.

Geometric Conservation Law

To avoid errors induced by the moving mesh, a geometric conservation law needs to be satisfied numerically in addition to the mass, momenta, and energy conservation laws which govern the physics of the flow. In most of the unsteady flow computations that have been published previously, however, this additional conservation law has not been satisfied. As discussed by Thomas and Lombard,[17] the geometric conservation law is of the same integral form as the mass conservation law and is defined by

$$\frac{\partial}{\partial t}\iint_{\Omega} \mathrm{d}x\,\mathrm{d}y - \int_{\partial\Omega}(x_t\,\mathrm{d}y - y_t\,\mathrm{d}x) = 0 \tag{26}$$

This geometric conservation law must be solved numerically using the same scheme that is used to integrate the conservation laws of the fluid to provide a self-consistent solution for the local cell areas.[17] Discretization of Eq. (26), consistent with the solution algorithms already presented, yields

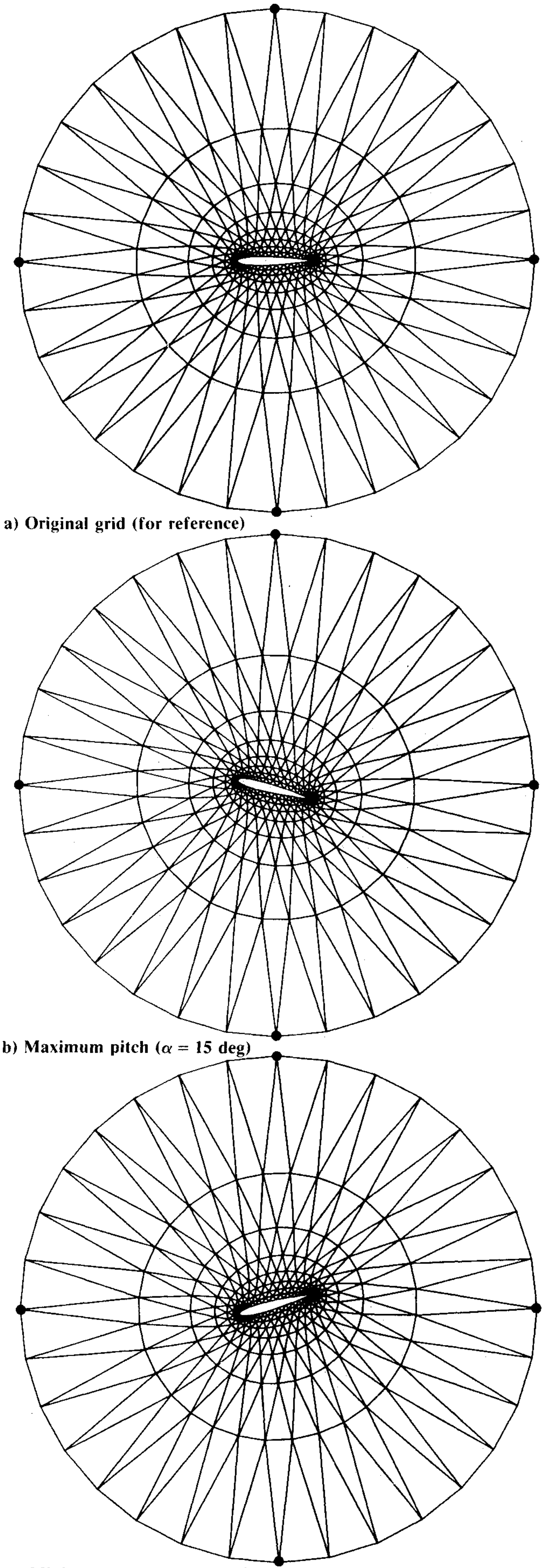
$$A_i^{n+1} = A_i^n + \Delta t\Sigma(x_{t_m}^{n+1}\Delta y_m - y_{t_m}^{n+1}\Delta x_m) \tag{27}$$

where the summation is taken over all edges of the control volume with endpoints at node i.

It is easy to demonstrate that the addition of the geometric conservation law is important for computations involving moving meshes by considering a uniform steady-state flow. Conservative finite-volume schemes with a fixed mesh have the property that a uniform flow is an exact solution of the numerical algorithm. For a moving mesh, the uniform flow is preserved only if the cell areas are computed using the discrete form of the geometric conservation law. This is true even if the mesh moves rigidly and the cell areas are not changing geometrically. Similarly for nonuniform flows, the cell areas need to be recomputed at each time step for rigidly moving meshes or for deforming meshes to prevent grid-motion-induced errors in the numerical solution. Therefore, Eq. (27) was used in the present study to compute the local cell areas at time level $(n + 1)$, as required by the time-integration schemes of Eqs. (6) and (21). For the cases considered in the present study, however, the effects of using the geometric conservation law in comparison with the geometrical determination of the cell areas were small.

Programming Considerations

Flow solvers based on unstructured grids have additional programming considerations that are not necessary in solvers based on structured grids. For example, connectivity information is required to determine which nodes make up a given triangle, and an indirect addressing system is used to access this information. Furthermore, special programming strategies, which make extensive use of gather and scatter operations, are used to achieve full vectorization of the solution algorithm.[18] Vectorization is accomplished primarily by sorting the elements into groups such that within each group no



a) Original grid (for reference)

b) Maximum pitch ($\alpha = 15$ deg)

c) Minimum pitch ($\alpha = -15$ deg)

Fig. 3 Sequence of grids about the NACA 0012 airfoil which illustrate how the mesh moves for a pitching airfoil.

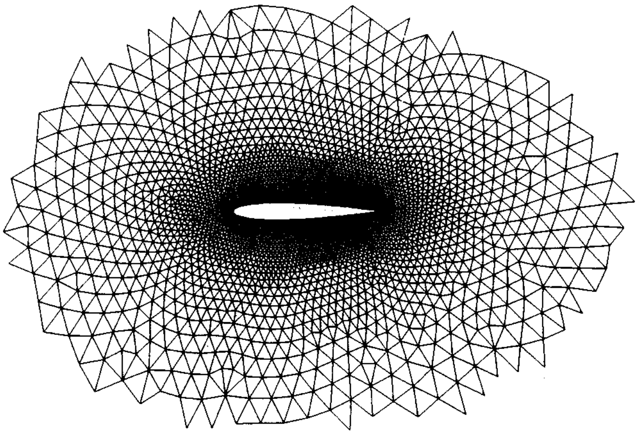


Fig. 4 Partial view of unstructured grid of triangles about the NACA 0012 airfoil.

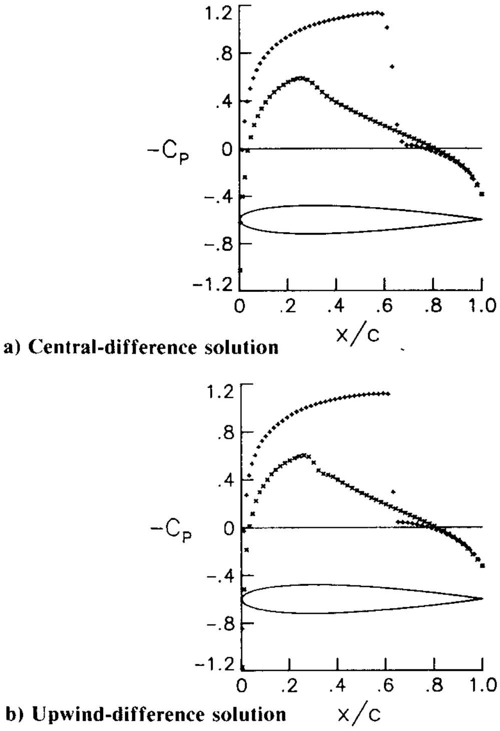
element accesses the same node. This avoids vector recurrences in the flux balance computation that would prevent vectorization. The programming procedure then "gathers" flow variable information from the elements into these groups, performs the flux balance separately within each group to solve the Euler equations, and finally "scatters" the resulting information back to the nodes which make up the grid.

Results and Discussion

To assess the Euler solvers and dynamic mesh algorithm, calculations were performed for the NACA 0012 airfoil. These results were obtained using the unstructured grid shown in Fig. 4, which was generated using the advancing front method.[6,19] The grid has 3298 nodes, 6438 triangles, and extends 20 chordlengths from the airfoil with a circular outer boundary. Also there are 110 points that lie on the airfoil surface. Steady-state calculations were performed for the airfoil at a freestream Mach number of $M_\infty = 0.8$ and an angle of attack of $\alpha_0 = 1.25$ deg. Unsteady calculations were performed for the airfoil pitching harmonically about the quarter chord with an amplitude of $\alpha_1 = 2.51$ deg and a reduced frequency based on semichord of $k = 0.0814$ at $M_\infty = 0.755$ and $\alpha_0 = 0.016$ deg. Two sets of steady and unsteady results were obtained corresponding to solutions computed using the central-difference and upwind-difference algorithms. These calculations are compared with the experimental unsteady data of Ref. 20.

Steady-Flow Results

Steady-flow results were obtained for the NACA 0012 airfoil with the central-difference algorithm using 2000 iterations and a CFL number of 5. Convergence to steady state was accelerated using local time stepping, implicit residual smoothing, and enthalpy damping. The solution residual was reduced by over eight orders of magnitude as determined by the reduction in the L_2 norm of the density residual. To engineering accuracy an acceptably converged solution can be obtained in less than 1000 iterations, which requires approximately 6 min of CPU time on the VPS-32 (CDC Cyber 205) computer at NASA Langley Research Center. Steady flow results were also obtained for the NACA 0012 airfoil with the upwind-difference algorithm using 4500 iterations and a CFL number of 1.8. Convergence to steady state was accelerated using local time stepping. The solution residual was reduced by five orders of magnitude, which cost approximately twice that of the central-difference solution on a per iteration basis. The increase in computational cost is due to the increased number of operations required by the flux-vector splitting in comparison with the central-difference algorithm.



a) Central-difference solution

b) Upwind-difference solution

Fig. 5 Comparison of steady pressure distributions for the NACA 0012 airfoil at $M_\infty = 0.8$ and $\alpha_0 = 1.25$ deg.

Steady pressure distributions for the NACA 0012 airfoil at $M_\infty = 0.8$ and $\alpha_0 = 1.25$ deg are shown in Fig. 5. The central-difference solution is shown in part a of Fig. 5, and the upwind-difference solution is shown in part b of Fig. 5. For this case there is a relatively strong shock wave on the upper surface of the airfoil near 62% chord and there is a relatively weak shock wave on the lower surface near 30% chord. The solution obtained using the central-difference algorithm indicates that the upper surface shock is captured with about three internal grid points and the lower surface shock is smeared. The solution obtained using the upwind-difference algorithm shows that there is only one interior grid point within the upper surface shock structure and that the lower surface shock is also more sharply captured. Away from the shocks, the two sets of pressure are in good agreement with each other. Furthermore, the steady pressure results of Fig. 6 are of comparable accuracy in comparison with the numerous published results for this case such as those reported in Ref. 21.

Unsteady-Flow Results

Unsteady results were obtained for the pitching NACA 0012 airfoil using step sizes of $\Delta t = 0.0204$ and 0.0051 for the central-difference and upwind-difference algorithms, respectively. A smaller step size was used in the upwind method since a smaller CFL number is required, and the implicit residual smoothing was not used. In each case, three cycles of motion were computed to obtain a periodic solution.

Calculated instantaneous pressure distributions at eight points in time during the third cycle of motion are shown in Fig. 6 for comparison with the experimental data. In each pressure plot, the instantaneous angle of attack $\alpha(\tau)$ and the angular position in the cycle $k\tau$ are noted where τ is time, nondimensionalized by airfoil semichord and the streamwise freestream speed. During the first part of the cycle, there is a shock wave on the upper surface of the airfoil, and the flow over the lower surface is predominantly subcritical. During the latter part of the cycle, the flow about the upper surface is subcritical, and a shock forms along the lower surface. The pressure distributions indicate that the shock position oscillates over approximately 25% of the chord along each surface and, in general, that the two sets of calculated results compare well with the experimental data. Similar to the steady-flow results, the upwind-difference algorithm captures shock waves

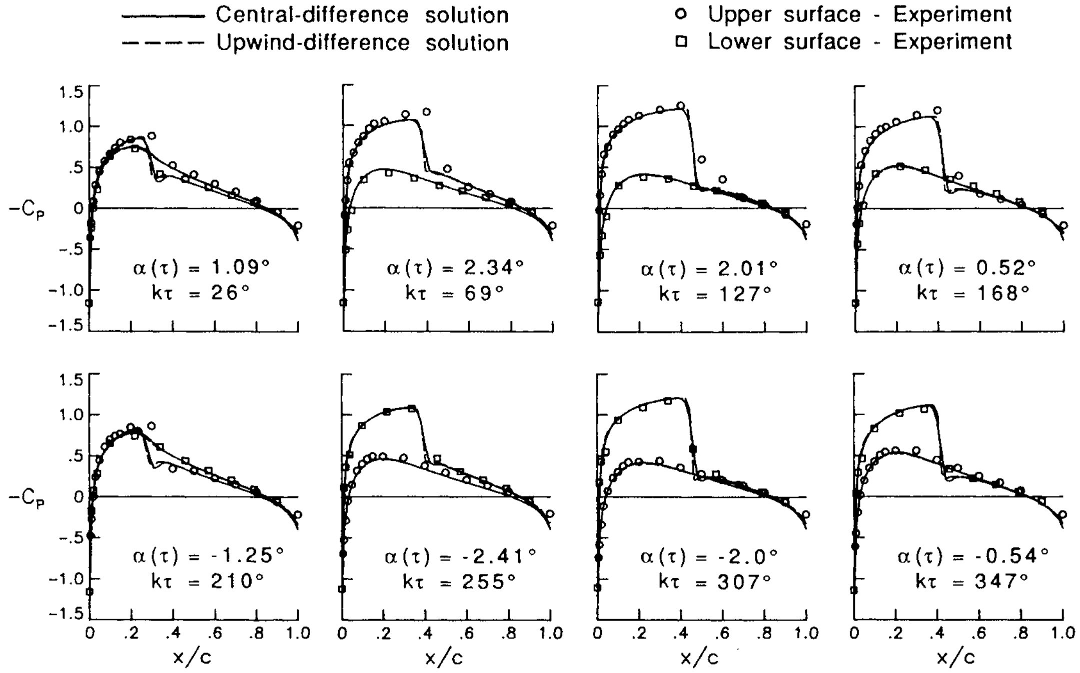
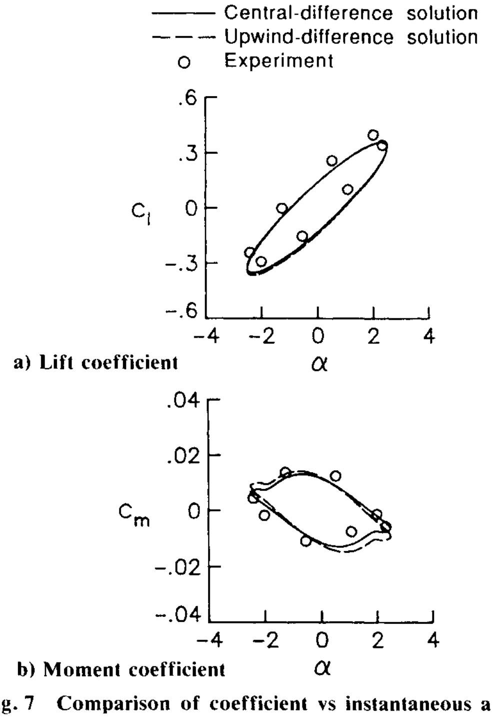


Fig. 6 Comparison of instantaneous pressure distributions for the NACA 0012 airfoil pitching at $M_\infty = 0.755$, $\alpha_0 = 0.016$ deg, $\alpha_1 = 2.51$ deg, and $k = 0.0814$.



a) Lift coefficient

b) Moment coefficient

Fig. 7 Comparison of coefficient vs instantaneous angle of attack for the NACA 0012 airfoil pitching at $M_\infty = 0.755$, $\alpha_0 = 0.016$ deg, $\alpha_1 = 2.51$ deg, and $k = 0.0814$.

that are sharper in comparison with the shocks predicted by the central-difference algorithm. The calculated results, however, show the expected symmetry in the flow in that the upper surface pressure distribution during the first half of the cycle is very similar to the lower surface pressure distribution during the second half of the cycle. The experimental data therefore appears to have been obtained at a slightly higher steady-state angle of attack than that reported in Ref. 20.

Comparisons of calculated and experimental lift and moment coefficients vs the instantaneous angle of attack are presented in Fig. 7. The lift coefficient is shown in part a of Fig. 7, and the moment coefficient is shown in part b of Fig. 7. These coefficients show the variation as a function of angle of attack during a cycle of motion, and, in general, the two sets of calculated results compare well with the experimental data. The comparisons of lift coefficient further indicate that the data was probably obtained at a higher steady-state angle of attack since the experimental values are higher than the calculated values. Furthermore, the unstructured grid results of Figs. 6 and 7 are of comparable accuracy in comparison with published results obtained using structured grid methods for this case, such as those reported by Anderson and Batina.[21]

Concluding Remarks

Two algorithms for the solution of the time-dependent Euler equations were presented for unsteady aerodynamic analysis of oscillating airfoils. Both algorithms were developed for use on unstructured grids made up of triangles. The first flow solver involves a Runge-Kutta time stepping scheme with a finite-volume spatial discretization that reduces to central differencing on a rectangular mesh. The second flow solver involves a modified Euler time-integration scheme with an upwind-biased spatial discretization based on the flux-vector splitting developed by Van Leer. A significant aspect of the research is the implementation of a dynamic mesh algorithm that is employed for problems where the airfoil moves. The method is completely general in that it can treat realistic airfoil motions including aeroelastic transient-type motion.

Steady and unsteady flow results were presented for the NACA 0012 airfoil to demonstrate application of the Euler solvers and dynamic mesh algorithm. The unsteady results were obtained for the airfoil pitching harmonically about the quarter chord. Two sets of steady and unsteady results were obtained corresponding to solutions computed using the central-difference and upwind-difference algorithms. Comparisons of these solutions indicated that the shock waves predicted by the upwind method were more sharply captured than those predicted by the central-difference method, as expected. Also, for the unsteady case that was considered, the two sets of calculated instantaneous pressure distributions and lift and moment coefficients compared well with the experimental data.

The accuracy of the results presented and the ease with which the unstructured grid can be moved for realistic airfoil motions indicate that the unstructured grid methodology is attractive for time-dependent problems involving moving boundaries. Therefore, the current effort is being directed toward increasing the computational efficiency of the present procedures and developing three-dimensional flow solvers using tetrahedral grids for unsteady aerodynamic and aeroelastic analysis of complete aircraft configurations as reported in Ref. 22.

Acknowledgments

The author would like to thank Dimitri Mavriplis of the Institute for Computer Applications in Science and Engineering, Hampton, VA, for providing the Delaunay triangulation method grid generation program, and Paresh Parikh of Vigyan Research Associates, Hampton, VA, and Rainaid Lohner of George Washington University, Washington, D.C., for providing the advancing front method grid generation program.

References

[1] Jameson, A., "Successes and Challenges in Computational Aerodynamics," AIAA Paper 87-1184, Jan. 1987.

[2] Edwards, J. W., and Thomas, J. L., "Computational Methods for Unsteady Transonic Flows," AIAA Paper 87-0107, Jan. 1987.

[3] Jameson, A., and Mavriplis, D. J., "Finite Volume Solution of the Two-Dimensional Euler Equations on a Regular Triangular Mesh," *AIAA Journal,* Vol. 24, No. 4, 1986, pp. 611–618.

[4] Mavriplis, D. J., "Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes," *AIAA Journal,* Vol. 26, No. 7, 1988, pp. 824–831.

[5] Jameson, A., Baker, T. J., and Weatherill, N. P., "Calculation of Inviscid Transonic Flow Over a Complete Aircraft," AIAA Paper 86-0103, Jan. 1986.

[6] Morgan, K., and Peraire, J., "Finite Element Methods for Compressible Flow," Von Karman Institute for Fluid Dynamics Lecture Series 1987-04, Computational Fluid Dynamics, Belgium, March 2–6, 1987.

[7] Lohner, R., "Finite Elements in CFD: What Lies Ahead," *International Journal for Numerical Methods in Engineering,* Vol. 24, 1987, pp. 1741–1756.

[8] Morgan, K., Peraire, J., Thareja, R. R., and Stewart, J. R., "An Adaptive Finite Element Scheme for the Euler and Navier-Stokes Equations," AIAA Paper 87-1172, June 1987.

[9] Peraire, J., Peiro, J., Formaggia, L., and Morgan, K., "Finite Element Euler Computations in Three Dimensions," AIAA Paper 88-0032, Jan. 1988.

[10] Van Leer, B., "Flux-Vector Splitting for the Euler Equations," *Lecture Notes in Physics,* Vol. 170, 1982, pp. 507–512.

[11] Anderson, W. K., Thomas, J. L., and Van Leer, B., "Comparison of Finite Volume Flux Vector Splittings for the Euler Equations," *AIAA Journal,* Vol. 24, No. 9, 1986, pp. 1453–1460.

[12] Anderson, W. K., Thomas, J. L., and Rumsey, C. L., "Extension and Applications of Flux Vector Splitting to Unsteady Calculations on Dynamic Meshes," AIAA Paper 87-1152, June 1987.

[13] Parpia, I. H., "Van Leer Flux Vector Splitting in Moving Coordinates," *AIAA Journal,* Vol. 26, No. 1, 1988, pp. 113–115.

[14] Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes," AIAA Paper 81-1259, 1981.

[15] Venkatakrishnan, V., "Computation of Unsteady Transonic Flows Over Moving Airfoils," Ph.D. Thesis, Princeton University, Princeton, NJ, 1986.

[16] Turkel, E., and Van Leer, B., "Runge-Kutta Methods for Partial Differential Equations," Institute for Computer Applications in Science and Engineering Report, Hampton, VA, 1983.

[17] 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.

[18] Lohner, R., and Morgan, K., "Finite Element Methods on Supercomputers: The Scatter Problem," *Proceedings of the NUMETA 1985 Conference,* Rotterdam, 1985, pp. 987–990.

[19] Lohner, R., "Some Useful Data Structures for the Generation of Unstructured Grids," *Communications in Applied Numerical Methods,* Vol. 4, 1988, pp. 123–135.

[20] Landon, R. H., "NACA 0012. Oscillating and Transient Pitching," *Compendium of Unsteady Aerodynamic Measurements,* Data Set 3, AGARD-R-702, Aug. 1982.

[21] Anderson, W. K., and Batina, J. T., "Accurate Solutions, Parameter Studies, and Comparisons for the Euler and Potential Flow Equations," Paper No. 14 at the AGARD Symposium on Validation of Computational Fluid Dynamics, Lisbon, Portugal, May 2–5, 1988.

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