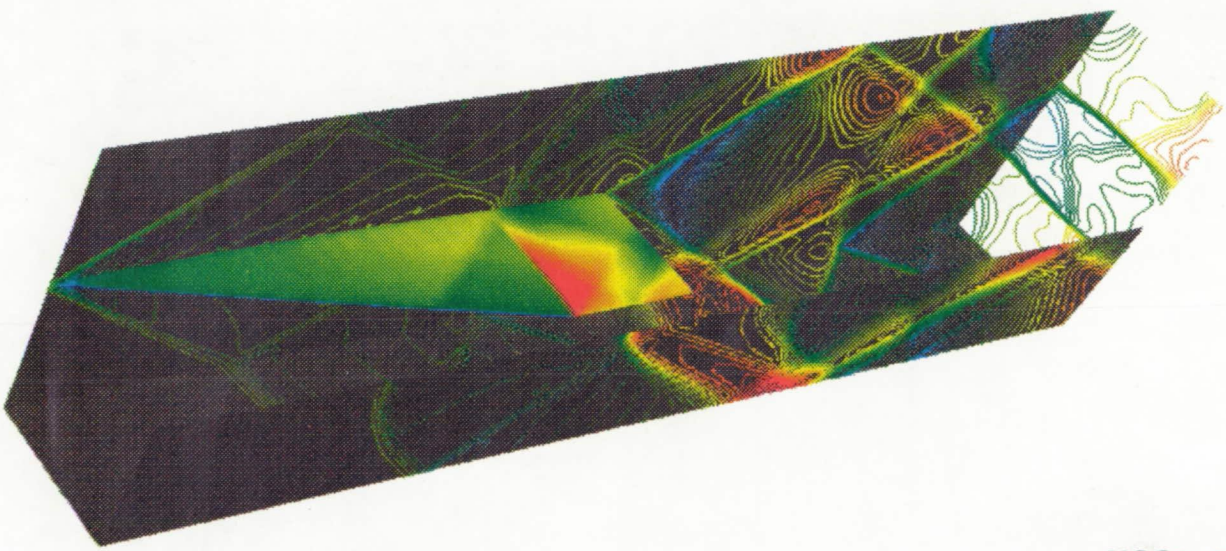
MCAT Institute
Final Report
93-14

CNSFV Code Development, Virtual Zone Navier-Stokes Computations of Oscillating Control Surfaces and Computational Support of the Laminar Flow Supersonic Wind Tunnel

Goetz H. Klopfer



NCC2-616

May 1993

MCAT Institute
3933 Blue Gum Drive
San Jose, CA 95127

CNSFV Code Development, Virtual Zone Navier-Stokes Computations of Oscillating Control Surfaces and Computational Support of the Laminar Flow Supersonic Wind Tunnel

Goetz H. Klopfer

MCAT Institute
Final Report
93-14

Corporate Agreement NCC2-616

May 1993

MCAT Institute
3933 Blue Gum Drive
San Jose, CA 95127

CNSFV Code Development, Virtual Zone Navier-Stokes Computations of Oscillating Control Surfaces and Computational Support of the Laminar Flow Supersonic Wind Tunnel

Abstract

The work performed during the past year on this cooperative agreement covered two major areas and two lesser ones. The two major items included further development and validation of the CNSFV code and providing computational support for the Laminar Flow Supersonic Wind Tunnel (LFSWT). The two lesser items involve a Navier-Stokes simulation of an oscillating control surface at transonic speeds and improving the basic algorithm used in the CNSFV code for faster convergence rates and more robustness. The work done in all four areas is in support of the High Speed Research Program at NASA Ames Research Center.

Introduction

The numerical simulation of the Navier-Stokes equations for complex configuations at realistic flight conditions is still limited by several problems. Included are the lack of adequate grid resolution, robust and efficient flow solvers, transition prediction techniques, and turbulence models. The work covered in this report involves the first three items only. The grid resolution and robust flow solver problems are included in the CNSFV code development work and the transition prediction technique is included in the work involving flow analysis of the laminar flow supersonic wind tunnel currently under development at NASA Ames. The flow analysis is by numerical simulation of the Navier-Stokes equations with the CNSFV code. In the following sections the work performed during the past year in the four different categories are described in more detail.

CNSFV Code Development

The CNSFV code is a cell-centered finite volume version of the finite difference code, CNS. The reason for developing the finite volume code was to facilitate the implementation of conservative zonal interface boundary condition. The finite difference form is not practical for conservative interfacing. The CNSFV code was originally developed three years ago (see ref. 1) and has been under continuous improvement since its inception. During the past year further improvements have been made. These include simplifying the type and amount of input data required, implementing general boundary conditions, increasing the code efficiency in terms of vectorization and algorithmic improvements, and developing special zoning capabilities (called virtual zones) to ease the problem of generating grids with multiple zones about complex and dynamic aerodynamic configurations. An example of an application of the virtual zone technology is given in reference 2, where a complete wing-body configuration with control surfaces is simulated with the CNSFV code.

The diagonal ADI algorithm in the original finite difference code is a fast and robust scheme. However when the code was converted to finite volume form a sharp drop in the maximum allowable stable time step was noticed. Various types of local time step scalings were tried and the allowable times steps improved dramatically

but the time steps originally possible with the finite difference form were never achieved. The scaled allowable time steps were still too small however and the diagonal ADI scheme was replaced with the lower-upper symmetric Gauss Seidel (LU-SGS) scheme. This scheme is unconditional stable and arbitrarily large time steps can now be used. Most of the the work on implementing and validating the LU-SGS scheme in the CNSFV code is reported in an AIAA paper presented at the AIAA Fluid Dynamics Conference, ref.3. This paper is included as Appendix A to which the reader is referred for a complete discussion.

A preliminary version of an user's manual (ref. 4) for the CNSFV code has been written and is presently undergoing user testing. This work is still underway and expected to be completed by the end of this year.

Computational Support for the LFSWT

For the past 18 months computational support has been provided for the Laminar Flow Supersonic Wind Tunnel (LFSWT). The objective of the effort is to develop computational tools so that the design of a test model and its placement within the test section of the LFSWT can be verified by numerical simulation of the Navier-Stokes equations before the model is constructed. For transition studies in the supersonic Mach regime it is important to know the extent of clean and undisturbed flow over the test model.

For the simulation, modified versions of the Upwind Parabolized Navier-Stokes (UPS, ref. 5) and the Compressible Navier-Stokes, Finite Volume (CNSFV) codes were used to solve the thin-layer Navier-Stokes equations for the laminar flow about the test model inside the LFSWT. The surface and flow field grids were generated with GRIDGEN (ref. 6). The faster UPS code was used for the higher Mach numbers investigated and the multi-zonal CNSFV code for the lower supersonic Mach regime.

Computations have been performed of flow fields about a NACA64A010 wing with a 70^0 leading edge sweep mounted on the top wall of the LFSWT for inviscid and viscous flows with the UPS and CNSFV codes, respectively. Various other model locations were studied to verify that the top wall mounted position provides for the largest extent of undisturbed flow on the model. Figure 1 shows the inviscid shock pattern obtained with the UPS code. Shown are the impinging shocks on the tunnel walls as well as on the model itself. The undisturbed region on the model is the triangular region in front of the reflected shock wave impinging on the model. The flowfield behind the impinging shock wave is no longer undisturbed and, hence, that part of the model behind the shock is useless for any natural transition study.

Another series of computations were carried out on the NACA64A010 wing with four different leading edge sweep angles. The mean flow results were used to validate a Parabolized Stability Equation (PSE) code being developed by another group. In addition a full scale sized portion of the F16XL2 passive glove is presently being gridded and a flow solution is being obtained in preparation for the wind tunnel tests scheduled for early 1994. The results of the computations are being used to design the wind tunnel model of the F16XL2 wing. The wind tunnel tests are designed to validate the wind tunnel with flight tests.

An accurate prediction of the flow field inside the LFSW with various test models is important for maximizing the usefulness of the tunnel, especially when relatively small test sections are considered. Numerical simulations can also be used for designing tunnel modification and innovative passive and active tunnel devices to minimize the impact of reflected waves on the test model. This is a cost-effective means of increasing the usable size of the LFSWT.

Virtual Zone Navier-Stokes Computations of Oscillating Control Surfaces

Another area of effort conducted during the past year was to implement the virtual zone concept into a time accurate finite difference code, ENSAERO (ref. 7), for application to an oscillating control surface mounted on a clipped delta wing at transonic speeds. For a time accurate computation it is essential that the search procedure to find the interpolation coefficients for the inter-zonal communication be at least as efficient as the flow solver. Otherwise the code is not practical enough for routine use. Much of the work involved developing more efficient search procedures. The results of this effort were presented at the AIAA 11th CFD Meeting (ref. 8). The paper is included in this report as Appendix B.

Algorithm Development

As mentioned the CNSFV code development section, the LU-SGS scheme is unconditionally stable. However it does suffer slower convergence rates with increasing Reynolds number for viscous dominated flows. For this reason several modifications of the basic LU-SGS scheme were implemented. One modification succeeded in improving the convergence rates by as much as a factor of four. A typical example of the improved convergence rate for a compressible flat plate boundary layer flow at Mach 2 is shown in Figure 2. This work is still underway and the complete results will be reported at a future date (AIAA 25th Fluid Dynamics Conference in July 1994).

References

1. Klopfer, G. H. and Molvik, G. A.; "Conservative Multizonal Interface Algorithm for the 3-D Navier-Stokes Equations", AIAA Paper 91-1601, AIAA 10th CFD Conference, 24-27 June 1991, Honolulu, Hawaii.

2. Chaussee, D. S. and Klopfer, G. H.; "The Numerical Study for 3-D Flow Past Control Surfaces," AIAA Paper 92-4650, August 1992.

3. Klopfer, G. H. and Yoon, S.; "Multi-Zonal Navier-Stokes Code with the LU-SGS Scheme," AIAA Paper 93-2965, AIAA 24th Fluid Dynamics Conference, 6-9 July 1993, Orlando, Florida.

4. Klopfer, G. H.; "Preliminary Version of User's Manual for CNSFV," MCAT Report, May 1993.

5. Lawrence, S. L.; "Development of a Three-Dimensional Upwind Parabolized Navier-Stokes Code," AIAA J., Vol. 26, No. 6, June 1990, pp. 971-972.

6. Steinbrenner, J. P., Chawner, J. R., and Fouts, C. L.; "The GRIDGEN 3D Multiple Block Grid Generation System, Vol. I, Final Report," General Dynamics Corporation, WRDC-TR-90-3022, July 1990.

7. Obayashi, S. and Guruswamy, G. P.; "Navier-Stokes Computations for For Oscillating Control Surfaces," AIAA Paper 92-4431, August 1992.

8. Klopfer, G. H. and Obayashi, S.; "Virtual Zone Navier-Stokes Computations For Oscillating Control Surfaces," AIAA Paper 93-3363, AIAA 11th CFD Conference, 6-9 July 1993, Orlando, Florida.

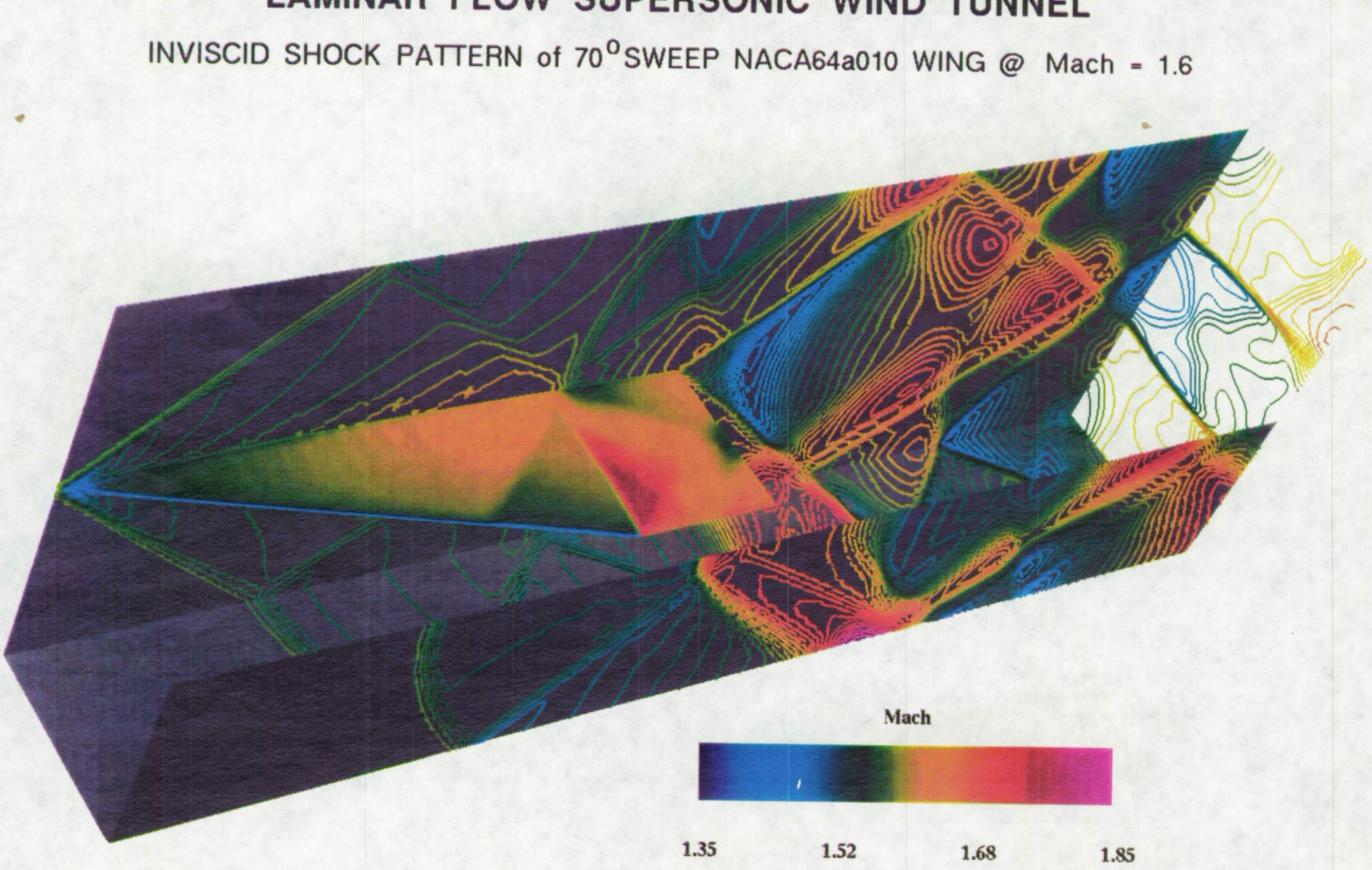


Figure 1. Shock pattern on a NACA64A010 wing in the LFSWT.

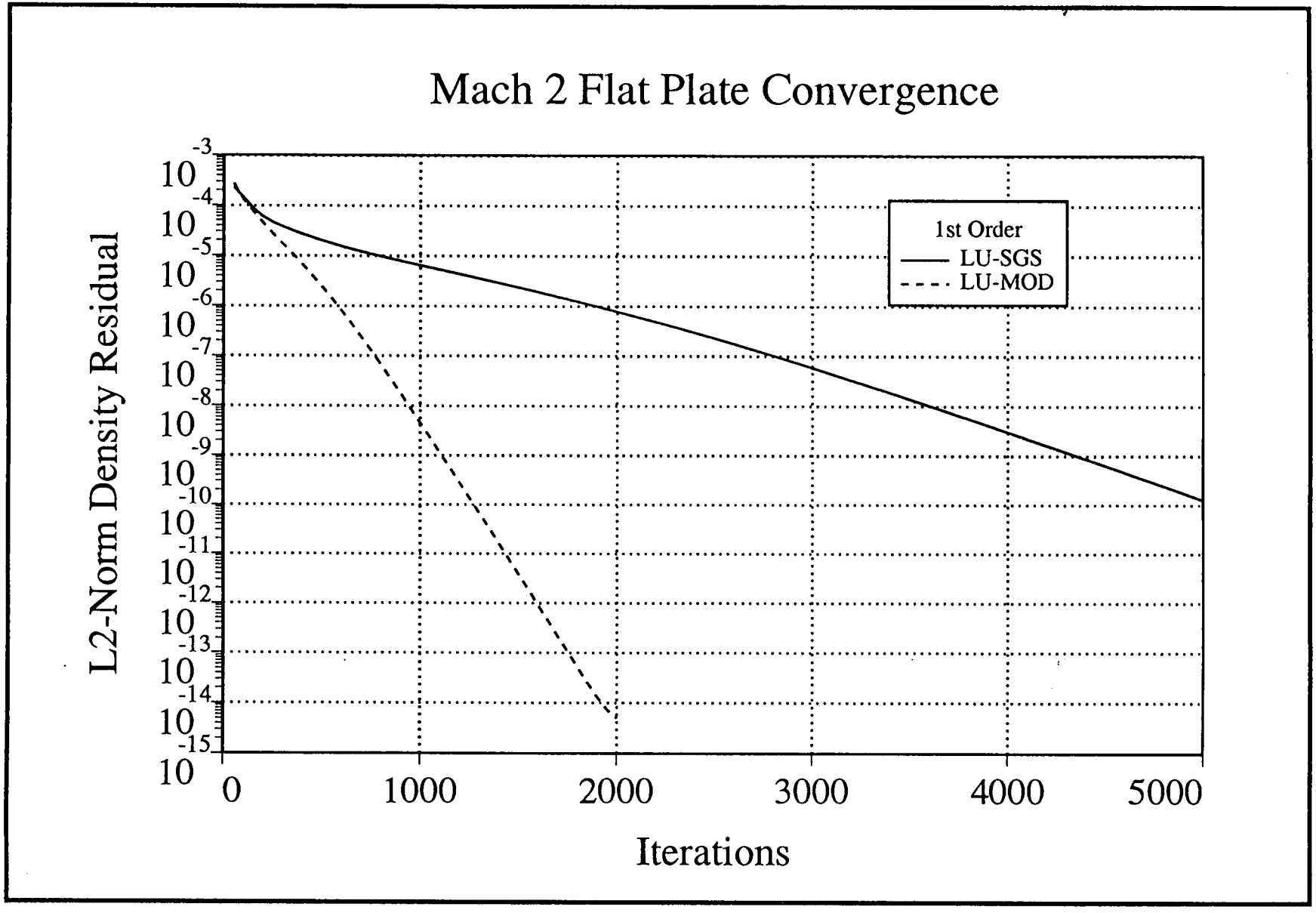


Figure 2. Comparison of convergence rates between standard and modified LU-SGS schemes for a flat plate boundary flow.

Appendix A

AIAA 93–2965
Multizonal Navier–Stokes Code with the LU–SGS Scheme

G. H. Klopfer and S. Yoon
MCAT Institute
Moffett Field, CA

AIAA 24th
Fluid Dynamics Conference
July 6-9, 1993 / Orlando, FL

MULTI-ZONAL NAVIER-STOKES CODE
with the LU-SGS SCHEME

G.H. Klopfer* and S. Yoon*
NASA Ames Research Center
Moffett Field, CA 94035-1000

Abstract

The LU-SGS (lower upper symmetric Gauss Seidel) algorithm has been implemented into the Compressible Navier-Stokes, Finite Volume (CNSFV) code and validated with a multizonal Navier-Stokes simulation of a transonic turbulent flow around an Onera M6 transport wing. The convergence rate and robustness of the code have been improved and the computational cost has been reduced by at least a factor of 2 over the diagonal Beam-Warming scheme.

Introduction

The numerical simulation of the Navier-Stokes equations about complex and realistic aerodynamic configurations with a structured grid requires the use of zonal methods. A popular and fairly efficient numerical scheme used to solve the Navier-Stokes equations is the diagonal implicit Beam-Warming algorithm [1,2]. A finite volume multi-zonal Navier-Stokes code, CNSFV, was developed using this scheme. However the Beam-Warming scheme uses approximate factorization to simplify the matrix inversion process. As a result the convergence properties of the scheme are dependent on the time step and time step scaling chosen, and much user input is required to determine the optimum time step and time step scaling. For a numerical simulation of a realistic aerodynamic configuration, several dozen zones may be required. Choosing the optimum time step and scaling for each of these zones becomes a tedious and time consuming process. For reasons not understood yet, the finite volume formulation of the diagonal Beam-Warming scheme suffers a severe time step restriction. Typically for constant time steps, the largest CFL number that could be exercised were less than ten and much less than that for the initial transients. With a judicious time step scaling, the maximum CFL can be as high as 50 to 75. The time step restriction is not so severe with the finite difference formulation of the diagonal Beam-Warming scheme, where the maximum CFL numbers can be as high as 500. The time step restriction for the finite volume form of the diagonal Beam-Warming scheme is severe enough that the time step is too small to be practical for an unsteady Navier-Stokes code.

A numerical scheme without the above mentioned time step restrictions is the LU-SGS (lower upper symmetric Gauss Seidel) algorithm [3,4]. This scheme has been implemented into the CNSFV code and validated with a multizonal Navier-Stokes simulation of a transonic flow around an Onera M6 transport wing.

Navier-Stokes Equations

The three-dimensional thin-layer Navier-Stokes equations in strong conservation law form in curvilinear coordinates are

$$\partial_\tau \hat{V} Q + \partial_\xi F + \partial_\eta G + \partial_\zeta H$$

$$= Re^{-1}(\partial_\xi F_v + \partial_\eta G_v + \partial_\zeta H_v) \qquad (1)$$

where

$$Q = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{pmatrix}, F = \begin{pmatrix} \rho U \\ \rho u U + \xi_x \hat{V} p \\ \rho v U + \xi_y \hat{V} p \\ \rho w U + \xi_z \hat{V} p \\ (e+p)U - \xi_t \hat{V} p \end{pmatrix},$$

$$G = \begin{pmatrix} \rho V \\ \rho u V + \eta_x \hat{V} p \\ \rho v V + \eta_y \hat{V} p \\ \rho w V + \eta_z \hat{V} p \\ (e+p)V - \eta_t \hat{V} p \end{pmatrix}, H = \begin{pmatrix} \rho W \\ \rho u W + \zeta_x \hat{V} p \\ \rho v W + \zeta_y \hat{V} p \\ \rho w W + \zeta_z \hat{V} p \\ (e+p)W - \zeta_t \hat{V} p \end{pmatrix} \qquad (2)$$

The contravariant velocity components are defined as

$$\begin{aligned} U &= \hat{V}(\xi_t + \xi_x u + \xi_y v + \xi_z w), \\ V &= \hat{V}(\eta_t + \eta_x u + \eta_y v + \eta_z w), \\ W &= \hat{V}(\zeta_t + \zeta_x u + \zeta_y v + \zeta_z w) \end{aligned} \qquad (3)$$

and the viscous fluxes are given by

$$F_v = \mu \hat{V} \begin{pmatrix} 0 \\ m_1 u_\xi + m_2 \xi_x \\ m_1 v_\xi + m_2 \xi_y \\ m_1 w_\xi + m_2 \xi_z \\ m_1 m_{0\xi} + m_2(\xi_x u + \xi_y v + \xi_z w) \end{pmatrix},$$

$$G_v = \mu \hat{V} \begin{pmatrix} 0 \\ m_3 u_\eta + m_4 \eta_x \\ m_3 v_\eta + m_4 \eta_y \\ m_3 w_\eta + m_4 \eta_z \\ m_3 m_{0\eta} + m_4(\eta_x u + \eta_y v + \eta_z w) \end{pmatrix},$$

---

* Senior Research Scientist, MCAT Institute, San Jose, CA 95127

$$H_v = \mu\hat{V}\begin{pmatrix} 0 \\ m_5 u_\zeta + m_6\zeta_x \\ m_5 v_\zeta + m_6\zeta_y \\ m_5 w_\zeta + m_6\zeta_z \\ m_5 m_{0_\zeta} + m_6(\zeta_x u + \zeta_y v + \zeta_z w) \end{pmatrix} \quad (4)$$

with

$$\begin{aligned} m_0 &= (u^2+v^2+w^2)/2 + (Pr(\gamma-1))^{-1}(a^2), \\ m_1 &= \xi_x^2 + \xi_y^2 + \xi_z^2, \\ m_2 &= (\xi_x u_\xi + \xi_y v_\xi + \xi_z w_\xi)/3, \\ m_3 &= \eta_x^2 + \eta_y^2 + \eta_z^2, \\ m_4 &= (\eta_x u_\eta + \eta_y v_\eta + \eta_z w_\eta)/3, \\ m_5 &= \zeta_x^2 + \zeta_y^2 + \zeta_z^2, \\ m_6 &= (\zeta_x u_\zeta + \zeta_y v_\zeta + \zeta_z w_\zeta)/3 \end{aligned} \quad (5)$$

The pressure is given by the equation of state

$$p = (\gamma-1)(e - \rho(u^2+v^2+w^2)/2) \quad (6)$$

where γ is the ratio of specific heats. The sound speed is denoted by a. The nondimensional parameters are the Reynolds number Re and the Prandtl number Pr. The coefficient of viscosity μ and thermal conductivity k are decomposed into laminar and turbulent contributions as follows:

$$\mu = \mu_l + \mu_t$$

$$\frac{k}{Pr} = \frac{\mu}{Pr} = \left(\frac{\mu_l}{Pr_l} + \frac{\mu_t}{Pr_t}\right)$$

where Pr_l and Pr_t are the laminar and turbulent Prandtl numbers and $k = \mu$ with the nondimensionalization used in this paper. The standard Baldwin-Lomax turbulent eddy viscosity model [5] is chosen for this study.

The nondimensional parameters chosen for this code are the same as those in the CNS code [6]. Denoting dimensional quantities with a $\tilde{()}$, the normalizing parameters are the freestream density $\tilde{\rho}_\infty$, the freestream sound speed $\tilde{a}_\infty$, the freestream viscosity coefficient $\tilde{\mu}_\infty$, and a characteristic length $\tilde{l}$.

The metrics used above have a different meaning for a finite volume formulation compared to the finite difference formulation of [6]. Referring to a typical finite volume cell as shown in figure 1, the finite volume metrics are defined (see, for example, Vinokur [7]) as

$$\mathbf{s}_{j+\frac{1}{2}} = s_{x,j+\frac{1}{2}}\mathbf{i} + s_{y,j+\frac{1}{2}}\mathbf{j} + s_{z,j+\frac{1}{2}}\mathbf{k}$$

$$= \frac{1}{2}[(\mathbf{r}_7 - \mathbf{r}_4)\times(\mathbf{r}_8 - \mathbf{r}_4) + (\mathbf{r}_3 - \mathbf{r}_4)\times(\mathbf{r}_7 - \mathbf{r}_4)]$$

$$\mathbf{s}_{k+\frac{1}{2}} = s_{x,k+\frac{1}{2}}\mathbf{i} + s_{y,k+\frac{1}{2}}\mathbf{j} + s_{z,k+\frac{1}{2}}\mathbf{k}$$

$$= \frac{1}{2}[(\mathbf{r}_7 - \mathbf{r}_2)\times(\mathbf{r}_3 - \mathbf{r}_2) + (\mathbf{r}_6 - \mathbf{r}_2)\times(\mathbf{r}_7 - \mathbf{r}_2)]$$

$$\mathbf{s}_{l+\frac{1}{2}} = s_{x,l+\frac{1}{2}}\mathbf{i} + s_{y,l+\frac{1}{2}}\mathbf{j} + s_{z,l+\frac{1}{2}}\mathbf{k} \quad (7)$$

$$= \frac{1}{2}[(\mathbf{r}_7 - \mathbf{r}_5)\times(\mathbf{r}_6 - \mathbf{r}_5) + (\mathbf{r}_8 - \mathbf{r}_5)\times(\mathbf{r}_7 - \mathbf{r}_5)]$$

The finite volume metrics represent the cell face area normals in each of the curvilinear coordinates (ξ, η, ζ). They are related to the metrics introduced in equations (1 - 5) as follows

$$\begin{aligned} \xi_{x_i}\hat{V} &= s_{x_i,j+\frac{1}{2}} \\ \eta_{x_i}\hat{V} &= s_{x_i,k+\frac{1}{2}} \\ \zeta_{x_i}\hat{V} &= s_{x_i,l+\frac{1}{2}} \end{aligned} \quad (8)$$

where $x_i = x, y, z$ for $i = 1, 2, 3$. The volume of the computational cell is given by

$$\begin{aligned} \hat{V} = \frac{1}{6}[&(\mathbf{r}_4 - \mathbf{r}_2)\times(\mathbf{r}_3 - \mathbf{r}_1)\cdot(\mathbf{r}_7 - \mathbf{r}_3) \\ &+(\mathbf{r}_2 - \mathbf{r}_5)\times(\mathbf{r}_6 - \mathbf{r}_1)\cdot(\mathbf{r}_7 - \mathbf{r}_6) \\ &+(\mathbf{r}_5 - \mathbf{r}_4)\times(\mathbf{r}_8 - \mathbf{r}_1)\cdot(\mathbf{r}_7 - \mathbf{r}_8)] \end{aligned} \quad (9)$$

and is the finite volume equivalent of the inverse Jacobian of the coordinate transformation in the finite difference formulation of [6].

Numerical Method

The governing equations are integrated in time for both steady and time accurate calculations. The unfactored linear implicit scheme is obtained by linearizing the flux vectors about the previous time and dropping second and higher order terms. The resulting scheme in finite volume form is given by

$$[I + \hat{V}^{-1}\hat{h}(\delta_\xi A^n + \delta_\eta B^n + \delta_\zeta C^n - Re^{-1}(\delta_\xi L^n + \delta_\eta M^n + \delta_\zeta N^n))]$$

$$\cdot\Delta Q^n = -\hat{V}^{-1}\hat{h}R^n \quad (10)$$

where the residual R^n is

$$R^n = [\delta_\xi F^n + \delta_\eta G^n + \delta_\zeta H^n - Re^{-1}(\delta_\xi F_v^n + \delta_\eta G_v^n + \delta_\zeta H_v^n)] \quad (11)$$

The convective flux jacobians A, B, C and the viscous flux jacobians L, M and N are defined in the appendix of [1].

Solving the above equation set by a direct matrix inversion is still not practical for three dimensional problems. However there are several indirect or approximate methods available, including the diagonal Beam-Warming scheme and the lower-upper (LU) factorized scheme of Yoon and Jameson. In a previous study [8,9], the diagonal Beam-Warming scheme was the basic flow solver algorithm for the multi-zonal compressible Navier- Stokes finite volume code, CNSFV. While the scheme worked well for finite difference codes, the performance deteriorated for the finite volume code. The cause of the deterioration is not known, but it is not isolated to the CNSFV code; several other finite volume codes using ADI schemes seem to suffer from the same type of deterioration. Because of this problem, a LU scheme which does not seem to suffer the same problem as the ADI schemes is incorporated into the CNSFV code. A brief description of both the diagonal Beam-Warming and the LU scheme is presented.

Diagonal Beam-Warming Algorithm

With the use of approximate factorization and diagonalization of the flux jacobian matrices, a scalar pentadiagonal algorithm [2] can be derived from eqn (10) as

$$T_\xi[I + \hat{V}^{-1}\hat{h}\delta_\xi\Lambda_\xi]N[I + \hat{V}^{-1}\hat{h}\delta_\eta\Lambda_\eta]P[I + \hat{V}^{-1}\hat{h}\delta_\zeta\Lambda_\zeta]$$

$$\cdot T_\zeta{}^{-1}\Delta Q^n = R^n \quad (12)$$

where δ_ξ is a central difference operator and $\Delta Q^n = Q^{n+1} - Q^n$ with $Q^{n+1} = Q(t^n + \hat{h})$. The viscous terms are not included in the implicit side. The artificial dissipation is included in both sides and is derived below.

The inviscid flux jacobians are diagonalized as follows:

$$\partial_Q F = A = T_\xi \Lambda_\xi T_\xi^{-1}$$

$$\partial_Q G = B = T_\eta \Lambda_\eta T_\eta^{-1}$$

$$\partial_Q H = C = T_\zeta \Lambda_\zeta T_\zeta^{-1}$$

The T_ξ, T_η, T_ζ are the eigenvector matrices of A, B, C, respectively with $\Lambda_\xi, \Lambda_\eta, \Lambda_\zeta$ as the respective eigenvalues.

We also have

$$N = T_\xi^{-1} T_\eta$$

$$P = T_\eta^{-1} T_\zeta$$

Each of the factors of the implicit operator of equation (12) has an artificial dissipation term added to stabilize the central difference operator. The added term is based on Jameson's nonlinear second and fourth order dissipation and for the ξ-operator takes the form

$$\hat{h}\hat{V}^{-1}\nabla_\xi[\bar{\sigma}_{j+\frac{1}{2}}(\epsilon^{(2)}\Delta_\xi \cdot - \epsilon^{(4)}\Delta_\xi\nabla_\xi\Delta_\xi\cdot)]\Delta Q^n \quad (13)$$

with

$$\epsilon^{(2)} = \kappa_2 max(\gamma_{j+1}, \gamma_j, \gamma_{j-1}) \quad (14)$$

$$\gamma_j = \frac{|p_{j+1} - 2p_j + p_{j-1}|}{|p_{j+1} + 2p_j + p_{j-1}|} \quad (15)$$

$$\epsilon^{(4)} = max(0, \kappa_4 - \epsilon^{(2)}) \quad (16)$$

where κ_2, κ_4 are constants of o(1), and Δ_ξ, ∇_ξ are the forward and backward difference operators. $\bar{\sigma}_{j+\frac{1}{2}}$ is a modified spectral radius defined as

$$\bar{\sigma}_{j+\frac{1}{2}} = \bar{\sigma}_j + \bar{\sigma}_{j+1}$$

$$\bar{\sigma}_j = \sigma_j\left(1 + \sqrt{max(\sigma_k/\sigma_j, \sigma_l/\sigma_j)}\right), \quad (17)$$

$$\sigma_j = [|U| + a\sqrt{s_x^2 + s_y^2 + s_z^2}]_j \quad (18)$$

and where cell centered surface areas are used, e.g.

$$s_{x,j} = \frac{1}{2}(s_{x,j+\frac{1}{2}} + s_{x,j-\frac{1}{2}}) \quad (19)$$

The modified form of the spectral radius, equation (17), is suggested by Turkel [10] to account for large aspect ratio computational cells as for example in a viscous layer. Similar dissipation terms are obtained for the $\eta-$ and ζ-operators. The dissipation terms added to the right hand side of equation (10) are identical to those given above except that ΔQ^n is replaced by Q^n.

LU-SGS Scheme

The unfactored scheme of eqn (10) is given in terms of central differences for the implicit operator. It can also be represented in terms of upwinded differences. Dropping

the viscous terms in the implicit operator, equation (10) is in terms of upwind differences as follows:

$$[I+\hat{V}^{-1}\hat{h}(\delta_\xi^- A^+ + \delta_\xi^+ A^- + \delta_\eta^- B^+ + \delta_\eta^+ B^- + \delta_\zeta^- C^+ + \delta_\zeta^+ C^-)]$$

$$\cdot\Delta Q^n = -\hat{V}^{-1}\hat{h}R^n \tag{20}$$

where δ_ξ^+ and δ_ξ^- are the forward and backward difference operators. Similarly, A^+ and A^- are the Jacobian matrices which contain non-negative and non-positive eigenvalues, respectively.

The Yoon and Jameson version of the LU scheme can be obtained from the above equation by a simple reordering of the matrix elements and approximately factoring into two matrices. Define D, $\tilde{L}$, and $\tilde{U}$ to be matrices which contain the diagonal, sub-diagonal and super-diagonal elements of the implicit operator of equation (20), respectively.

$$[D+\tilde{L}+\tilde{U}]\Delta Q^n = -\hat{V}^{-1}\hat{h}R^n$$

which can be factored into

$$[D+\tilde{L}]D^{-1}[D+\tilde{U}]\Delta Q^n = -\hat{V}^{-1}\hat{h}R^n$$

Redefine $L = D+\tilde{L}$ and $U = D+\tilde{U}$ to yield the final form of the LU-SGS scheme.

$$LD^{-1}U\Delta Q^n = -\hat{V}^{-1}\hat{h}R^n \tag{21}$$

where

$$L = I + \hat{V}^{-1}\hat{h}(\delta_\xi^- A^+ + \delta_\eta^- B^+ + \delta_\zeta^- C^+ - A^- - B^- - C^-)$$

$$D = I + \hat{V}^{-1}\hat{h}(A^+ - A^- + B^+ - B^- + C^+ - C^-) \tag{22}$$

$$U = I + \hat{V}^{-1}\hat{h}(\delta_\xi^+ A^- + \delta_\eta^+ B^- + \delta_\zeta^+ C^- + A^+ + B^+ + C^+)$$

A variety of LU-SGS schemes can be obtained by different choices of the numerical dissipation models and Jacobian matrices of the flux vectors. In this study we use the same artificial dissipation described for the diagonal Beam-Warming scheme. For robustness and to ensure that the scheme converges to a steady state, the matrices should be diagonally dominant. To ensure diagonal dominance, the Jacobian matrices can be constructed so that + matrices have nonnegative eigenvalues and − matrices have nonpositive eigenvalues. For example, the diagonalization used for the Beam-Warming scheme can be used to obtain the ± matrices.

$$A^\pm = T_\xi \Lambda_\xi^\pm T_\xi^{-1}$$

$$B^\pm = T_\eta \Lambda_\eta^\pm T_\eta^{-1} \tag{23}$$

$$C^\pm = T_\zeta \Lambda_\zeta^\pm T_\zeta^{-1}$$

Another method to obtain diagonal dominance is to construct approximate Jacobian matrices:

$$A^\pm = [A \pm \tilde{\rho}(A)I]/2$$

$$B^\pm = [B \pm \tilde{\rho}(B)I]/2 \tag{24}$$

$$C^\pm = [C \pm \tilde{\rho}(C)I]/2$$

where $\tilde{\rho}(A) = max[|\lambda(A)|]$ and represent a spectral radius of the Jacobian matrix A with the eigenvalues $\lambda(A)$. These eigenvalues are, e.g., $\lambda(A) = [U, U, U, U \pm a\sqrt{s_x^2 + s_y^2 + s_z^2}]_j$, where the metric terms are again defined at cell centers by Eq. (19). Other methods of increasing the diagonal dominance of the LU operator include the addition some approximation of the viscous terms and the artificial dissipation in the implicit operator [11].

The inversion of equation (21) is done in three steps. The block inversion along the diagonal is eliminated if the approximate Jacobian of Eqn (24) are used instead of Eqn (23). A Newton-type of iteration is obtained simply by setting $\hat{h} = \infty$. Eliminating the time step from the algorithm provides the practical advantage of side-stepping the need to find an optimal CFL number or the time step scaling to reduce the overall computational effort to achieve steady state solutions. As mentioned previously, this is an important consideration for a multizonal code. If first-order one-sided differences are used, Eq. (22) reduces to

$$L = \tilde{\rho}I - A^+_{j-1,k,l} - B^+_{j,k-1,l} - C^+_{j,k,l-1}$$

$$D = \tilde{\rho}I \tag{25}$$

$$U = \tilde{\rho}I + A^-_{j+1,k,l} + B^-_{j,k+1,l} + C^-_{j,k,l+1}$$

where

$$\tilde{\rho} = \tilde{\rho}(A) + \tilde{\rho}(B) + \tilde{\rho}(C)$$

This algorithm requires only scalar diagonal inversions since the diagonal of L or $U = D$. The true Jacobians matrices of Eq. (23) may permit better convergence rates, but require block diagonal inversions with approximately twice the computational effort per iteration. This study considers the scalar version without the viscous or artificial dissipation only.

The scheme is completely vectorizable on $j+k+l =$ constant oblique planes of sweep, Fig. 2. This can be achieved by reordering the three dimensional arrays into two-dimensional arrays as follows:

$$P(i_p, p) = P(j, k, l)$$

where i_p is the serial number of the oblique plane of sweep and p is the address of point (j, k, l) in that plane. The specific value of p in terms of (j, k, l) is

$$p = j + k + l - 2$$

and the number of points in the plane p is

$$n_p = \frac{1}{2}p(p-1)$$

$$-\frac{1}{4}[(p - j_{max})(p - j_{max} + 1)(1 + isign(1, p - j_{max} - 1))$$

$$+(p - k_{max})(p - k_{max} + 1)(1 + isign(1, p - k_{max} - 1))$$

$$+(p - l_{max})(p - l_{max} + 1)(1 + isign(1, p - l_{max} - 1))$$

$$-(p - k_{max} - l_{max})(p - k_{max} - l_{max} + 1)$$

$$\cdot(1 + isign(1, p - k_{max} - l_{max} - 1))$$

$$-(p - j_{max} - l_{max})(p - j_{max} - l_{max} + 1)$$

$$\cdot(1 + isign(1, p - j_{max} - l_{max} - 1))$$

$$-(p - j_{max} - k_{max})(p - j_{max} - k_{max} + 1)$$

$$\cdot(1 + isign(1, p - j_{max} - k_{max} - 1))]$$

The maximum number of points in the planes is $n_{p,max} = n_p(p_{\frac{1}{2}})$, where $p_{\frac{1}{2}} = (p_{max} + 1)/2$. The vector lengths change from 1 at the beginning of the inversion sweep to a maximum of $n_{p,max}$ at the center of the sweep and back to 1 at the end. While $n_{p,max}$ can be large, the vector lengths at the beginning and end of the sweep are very small and inhibit efficient vectorization. Nevertheless for reasonably-sized zones, $n_{p,max}$ is sufficiently large so that good vectorization can be achieved. Other reordering sequences are possible, but additional sweeps will then be necessary and the overall performance of the LU-SGS inversion is less. In addition, the oblique planes of sweep allow for very efficient autotasking procedures on multi-processor computers; the efficiency is less with other sweep directions.

Boundary Conditions and Zonal Interfaces

To complete the equation set boundary conditions must be specified. With the use of curvilinear coordinates the physical boundaries have been mapped into computational boundaries, which simplifies the application of boundary conditions. The boundary conditions to be implemented for external viscous or inviscid flows include (1) inflow or far field, (2) outflow, (3) inviscid and (4) viscous impermeable wall, and (5) symmetry conditions. For external three-dimensional flow fields about closed bodies, the topology of the grid usually introduces (6) grid singularities which require special boundary conditions. The use of zonal methods can avoid the generation of grid singularities, but requires (7) special zonal interface boundary conditions. For compressible flows these zonal boundary conditions must be conservative to maintain global conservation. In this study we consider nonconservative interfaces. However, conservative interzonal communication has been developed and reported in a previous study [8].

A patched zonal procedure is used for the grid system. The zonal interfaces are general three-dimensional surfaces. The zoning procedure has a general capability so that a face of one zone can be in contact with several other zones. Information is transfered between zones with a bilinear interpolation procedure. The search method for determining the interpolation coefficients is automatic in that no user input is required other than identifying the zonal faces in contact with each other. The grid topology of the various zones is arbitrary and the zonal interfaces are not, in general, mesh continuous. In this study we use the ghost cell concept to obtain boundary and interface fluxes. Although there is no grid overlap at the zonal interfaces, the cell-centered conserved variables are reflected across surface and zonal boundaries and are determined by interpolating from the the adjacent zonal values. To determine the dissipative fluxes for the fourth-order dissipation requires two ghost cells. However, here we simply reduce the fourth-order dissipation to second-order so that only one ghost cell is needed.

Results

The test case chosen to validate the LU-SGS scheme and the code is the attached turbulent transonic flow over an ONERA M6 wing. This case has good experimental data [12] available and has been used extensively for Navier-Stokes code validation. The reason for choosing an attached flow case instead of a separated flow is that the emphasis of this study is to validate the flow solver and not the turbulence model. For attached flows the Baldwin-Lomax model [5] is sufficiently accurate. The flow conditons are $M_\infty = 0.84$, $\alpha = 3.06^0$, $Re_{mac} = 11.72 \times 10^6$. The wing surfaces are adiabatic and the fluid is a perfect gas with $\gamma = 1.4$. The computational grid consists of a C-O mesh split into four zones. There are two zones around the wing, one each on the upper and lower surface of the wing. Similarly, the wake region is split into an upper and lower grids. This grid, in the single zone version, seems to be the standard grid used for various validations of Navier-Stokes codes, see, e.g., ref. 13. Figure 3 shows a partial view of the grids and the zones. The total grid size is $193 \times 34 \times 49$ points, and the individual zones consist of $73 \times 34 \times 49$ and $25 \times 34 \times 49$ points for the wing and wake zones, respectively. The wall normal grid spacing at the surface is on the order of $y^+ = 4$. This is sufficient to properly resolve the boundary layer.

The converged results in terms of pressure contours on the upper surface of the wing are compared with experimental data [12] are shown in Fig. 4. The agreement between computation and experiment is good for the four span stations shown. These results are comparable to the results reported in ref. 13. The numerical predicted shocks are slightly smeared as compared to experiment. This is to

be expected since the numerical scheme uses central differencing with nonlinear scalar dissipation. Increased mesh resolution or an upwind scheme will sharpen the shock wave structure.

Figure 5 presents the performance comparison between the diagonal Beam-Warming scheme and the LU-SGS scheme. These views show the convergence rate for the upper wing zone with the two schemes. The first view shows that the LU-SGS scheme is not that much faster than the diagonal Beam-Warming scheme in terms of iterations required to reach convergence. However, in terms of cpu times, the LU-SGS scheme is at least twice as efficient, as shown in the second view.

Most of the gains in efficiency with the LU-SGS scheme is in the reduced operation count. Table 1 shows the unit cpu time (μsec/ gridpoint/ iteration) for both schemes, as measured on the NASA-Ames Cray C90 computer. The unit cpu time for the right hand side (RHS) includes the calculation required for the RHS fluxes, boundary conditions, interface interpolation, swapping data in/out of machine core for each zone at each time step, writeout of convergence data, and other miscellaneous operations. This unit time is the same for both schemes at $4.5\mu sec$. The left hand side (implicit) unit cpu time is substantially different. The times are $5.9\mu sec$ for the diagonal Beam-Warming and $2.2\mu sec$ for the LU-SGS scheme. The total times are $6.7\mu sec$ and $10.4\mu sec$ for the LU-SGS and diagonal Beam-Warming schemes, respectively. Furthermore, the diagonal Beam-Warming scheme required user intervention to obtain the optimum time step size as well as the time step scaling. The LU-SGS scheme uses no time step scaling and an infinitely large time step as explained in LU-SGS section above.

Conclusions

A multi-zonal compressible Navier-Stokes code has been improved by replacing the implicit diagonal Beam-Warming algorithm with the lower-upper symmetric Gauss Seidel scheme. With the new scheme the code is now much more robust, requires no user intervention to determine the optimum time step or time step scaling, converges faster, and requires only half the cpu time to obtain the same level of convergence. Most of the gain in efficiency is due to the lower operation count required by the LU-SGS algorithm. The algorithm and code have been validated with a turbulent simulation of an attached transonic flow around an Onera M6 wing.

Acknowledgements

The first author's work was supported by NASA Grant NCC 2-616 and the second author's by NCC 2-505.

References

1. Beam, R. and Warming, R. F.; "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations," AIAA Journal, Vol. 16, Apr. 1978, pp. 393-402.

2. Pulliam, T. H. and Chaussee, D. S.; "A Diagonal Form of an Implicit Approximate Factorization Algorithm," Journal of Computational Physics, Vol. 39, 1981, pp. 347-363.

3. Yoon, S. and Jameson, A.; "Lower-Upper Symmetric Gauss Seidel Method for the Euler and Navier-Stokes Equations," AIAA Journal, Vol. 26, Sept. 1987, pp. 1025-1026.

4. Yoon, S. and Kwak, D.; "An Implicit Three-Dimensional Navier-Stokes Solver for Compressible Flow," AIAA Paper 91-1555-CP, June 1991.

5. Baldwin, B. S. and Lomax, H.; "Thin-Layer Approximation and Algebraic Model for Separated Turbulent Flow," AIAA Paper 78-0257, Jan. 1978.

6. Flores, J. and Chaderjian, N. M.; "Zonal Navier-Stokes Methodology for Flow Simulations about Complete Aircraft," Journal of Aircraft, Vol. 27, No. 7, July 1990, pp. 583-590.

7. Vinokur, M.; "An Analysis of Finite Difference and Finite Volume Formulation of Conservation Laws," NASA CR-177416, June 1986.

8. Klopfer, G. H. and Molvik, G. A.; "Conservative Multizonal Interface Algorithm for the 3-D Navier-Stokes Equations," AIAA Paper 91-1601-CP, June 1991.

9. Chaussee, D. S. and Klopfer, G. H.; "The Numerical Study of 3-D Flow Past Control Surfaces," AIAA Paper 92-4650, August 1992.

10. Turkel, E. and Vatsa, V. N.; "Effect of Artificial Viscosity on Three-Dimensional Flow Solutions," AIAA Paper 90-1444, June 1990.

11. Shih, T. I-P., Steinthorsen, E., and Chyu, W. J.; "Implicit Treatment of Diffusion Terms in Lower-Upper Algorithms," AIAA J., Vol. 31. No. 4, 1993, pp. 788-791.

12. Schmitt, V. and Charpin, F.; "Pressure Distributions on the ONERA M6 Wing at Transonic Mach Numbers," AGARD-AR-138, Report of the Fluid Dynamics Panel, May 1979.

13. Rumsey, C. L. and Vatsa, V. N.; "A Comparison of the Predictive Capabilities of Several Turbulent Models Using Upwind and Central-Difference Computer Codes," AIAA Paper 93-0192, Jan. 1993.

Scheme	CNSFV - diag BW	CNSFV - LUSGS
Total	10.4μsec (340 mflop)	6.7μsec (330 mflop)*
LHS	5.9μsec xlinv etainv zetainv vpenta, etc	2.2μsec lusgs (oblique planes of sweeps)
RHS, etc.	4.5μsec	4.5μsec

* average vector length ~70, optimum length ~128

Table 1. Perftrace performance comparison of diagonal Beam-Warming and LU-SGS schemes on Cray C90 computer. (unit cpu time μsec/grid point/iteration)

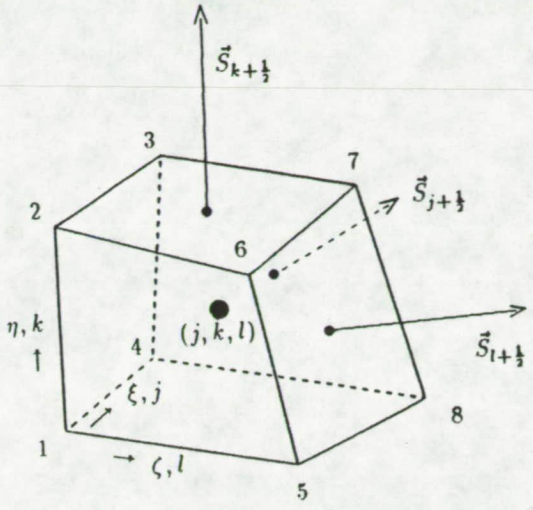


Fig. 1. Finite volume cell nomenclature.

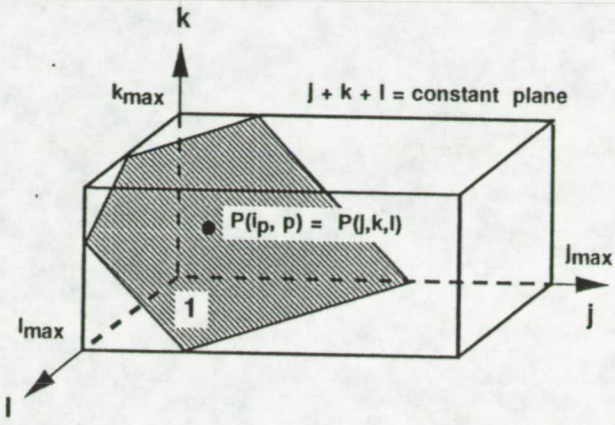


Fig. 2. Oblique plane of sweep for vectorization.

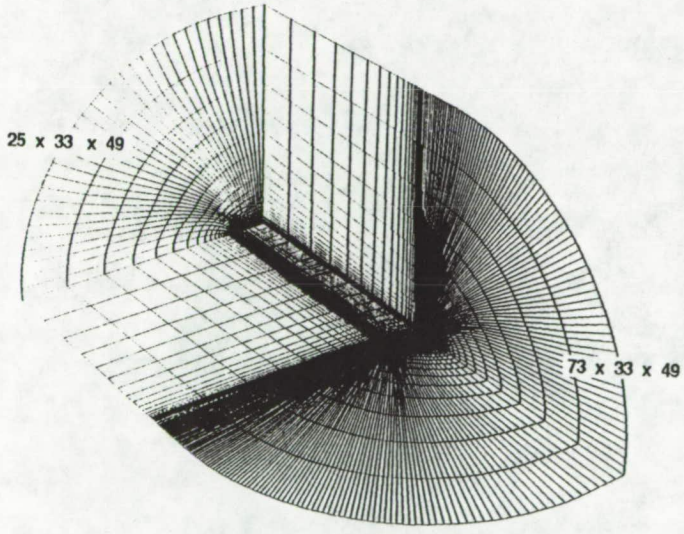
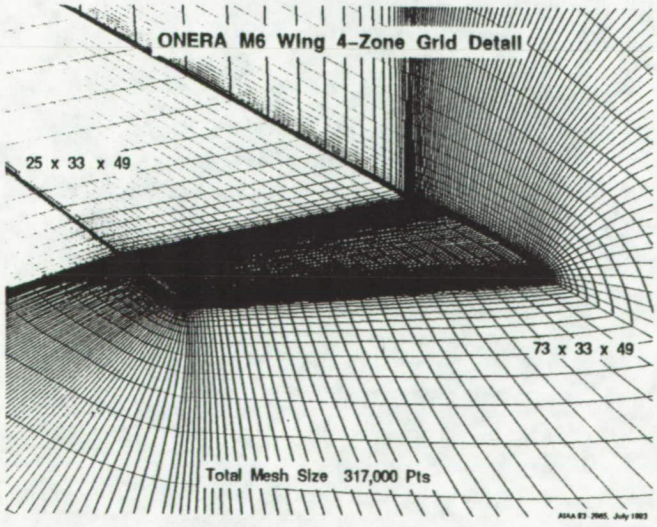


Fig. 3. Four zone C-O Grid for the ONERA M6 wing; only the two zones on the upper surface of wing and wake are shown.



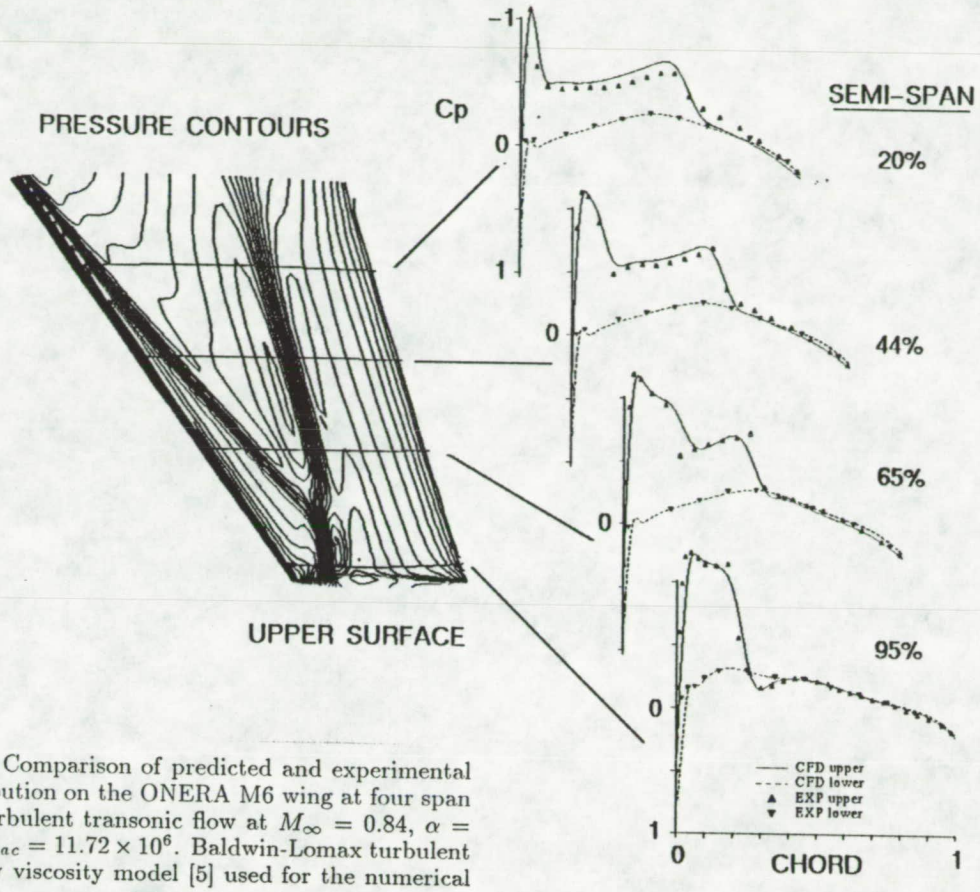


Fig. 4. Comparison of predicted and experimental [12] C_p distribution on the ONERA M6 wing at four span locations. Turbulent transonic flow at $M_\infty = 0.84$, $\alpha = 3.06$, and $Re_{mac} = 11.72 \times 10^6$. Baldwin-Lomax turbulent algebraic eddy viscosity model [5] used for the numerical prediction.

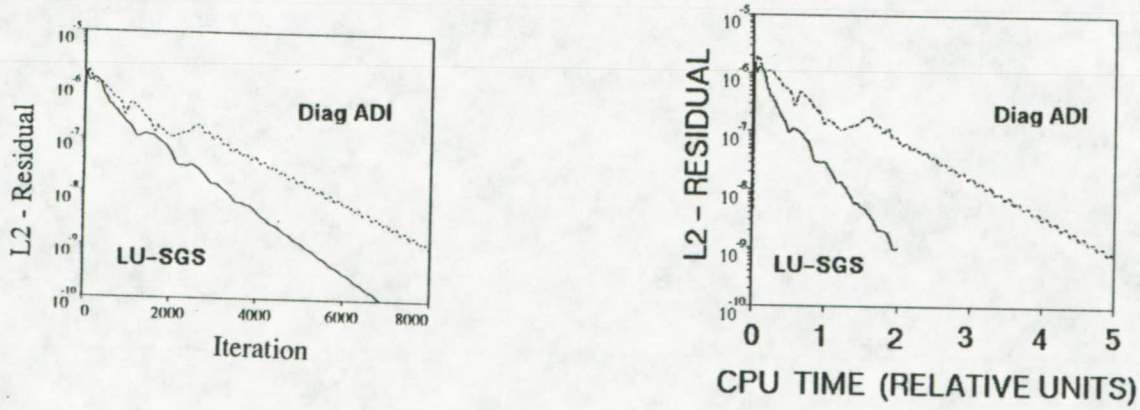


Fig. 5. Comparison of the convergence rates versus iteration and cpu times between the diagonal Beam-Warming and LU-SGS schemes.

Appendix B

Preprint of AIAA Paper 93-3363 presented at the AIAA 11th Computational Fluid Dynamics Conference, 6-9 July 1993, Orlando, Florida.

VIRTUAL ZONE NAVIER-STOKES COMPUTATIONS FOR OSCILLATING CONTROL SURFACES

G. H. Klopfer* and S. Obayashi*†
NASA Ames Research Center
Moffett Field, CA 94035-1000

Abstract

A new zoning method called "virtual zones" has been developed for application to an unsteady finite difference Navier-Stokes code. The virtual zoning method simplifies the zoning and gridding of complex configurations for use with patched multi-zone flow codes. An existing interpolation method has been extensively modified to bring the run time for the interpolation procedure down to the same level as for the flow solver. Unsteady Navier-Stokes computations have been performed for transonic flow over a clipped delta wing with an oscillating control surface. The computed unsteady pressure and response characteristics of the control-surface motion compare well with experimental data.

Introduction

Present transport aircraft as well as highly maneuverable fighter aircraft are often subject to unsteady aerodynamics. In this unsteady environment aircraft designers utilize active controls to achieve controllability and safety of the aircraft. Active control can also be used to suppress transonic flutter characteristics of high aspect ratio wings and thus reduce the structural weight to achieve more efficient flight conditions.

In the transonic flow regime active controls have a pronounced effect on the aerodynamic and aeroelastic performance of a wing. This effect can be used to improve the airplane performance by improved design of the active control surfaces. To do this successfully requires accurately predicting the aerodynamic and aeroelastic performance of a wing. Experimental prediction of unsteady aerodynamic and aeroelastic performance is costly, risky and time consuming; numerical simulation of the unsteady Navier-Stokes equations is a much more cost effective alternative for predicting the performance of an active control surface.

The physics of unsteady transonic flow around a control surface has been simulated with small disturbance theory [1,2]. Unsteady Navier-Stokes simulations have been performed in two dimensions [3,4]. A more recent study [5] conducted a three dimensional simulation of the unsteady thin-layer Navier-Stokes equations on the flow field surrounding a wing with a forced oscillating control surface. In that study an unsteady Navier-Stokes code, ENSAERO, was extended to simulate unsteady flows over a rigid wing with an oscillating trailing-edge flap.

* Senior Research Scientist, MCAT Institute, San Jose, CA 95127

† Senior Member AIAA

The numerical simulation of the unsteady Navier-Stokes equations about complex and realistic aerodynamic configurations requires the use of zonal methods. In this method the overall flow field domain is subdivided into smaller blocks or zones. In each of these zones the flow field is solved independently of the other zones. The boundary data for each zone is provided by the neighboring zones. A major difficulty of the zonal methods applied to oscillating control surfaces has been how to account for the variable exposure of the ends of the control surfaces to the flow field.

In Ref. 5 an algebraic grid generation technique was incorporated into the ENSAERO code. The grid moved at every time step to follow the deflection of the flap. The small unsteady deflections were handled using a sheared single mesh. The large mean deflections were handled using a zonal method [6]. The use of the single sheared grid did not permit the exact simulation of the unsteady flap-wing geometry. A gap had to be introduced between the ends of the flap and wing to allow sufficient space for the moving sheared mesh. The gap compromised the geometry and numerical predictions of the oscillating control surface flow field. The purpose of the present study is to rectify that compromise through the use of a new zoning technique called "virtual" zones.

The virtual zoning method, first implemented in a multizone finite volume code, CNSFV, [7], has been modified for application to the unsteady finite difference code, ENSAERO. The main purpose of these zones is to convert, for example, a solid wall boundary condition into an interface condition. The interface conditions are required for the interzonal communication. In a multi-zonal code the virtual zones are treated like real zones as far as boundary and interface conditions are concerned, however, no flow field computations are done within these zones. Hence, the name "virtual" zone is appropriate.

In addition to the introduction of the virtual zones, it is necessary to speed up the process of determining the interpolation coefficients required for the interzonal communication if the unsteady Navier-Stokes simulation is to be practical.

The present study considers the transonic vortical flow over a clipped delta wing. A view of the wing and the control surface is shown in Fig. 1. Unsteady Navier-Stokes computations for the clean wing were reported in Ref. 8. The forced oscillating control surface computations with the single zone sheared mesh were presented in Ref. 5.

Numerical Method

A brief description of the governing equations, grid and zonal system, and boundary conditions is given in this section. Most of the attention will be focussed on the virtual zoning concept and the unsteady interpolation procedure since these two items were crucial to the successful and practical application of the Navier-Stokes equations to

oscillating control surfaces.

Governing Equations and Discretization

The governing equations are the Reynolds-averaged thin-layer Navier-Stokes equations. The laminar viscosity is taken from the freestream laminar viscosity and is assumed to be constant for the transonic flow considered in this study. The turbulent viscosity is obtained with the Baldwin-Lomax algebraic eddy viscosity model [9] with the Degani-Schiff modification [10] to properly handle the leading edge separation as well as the control flap vortical flow.

The numerical algorithm, time dependent metrics of the curvilinear coordinate system, and performance characteristics of the ENSAERO code have been described previously and will not be repeated here. The interested reader is referred to Ref. 5.

Control Surface Grid and Zones

The primary focus of the present study is to demonstrate the feasibility of using dynamic zones for the oscillating control surface case rather than minimizing the cpu run time for the case to be presented. Hence the flow domain was split into only three real zones. Each of the zones consists of a C-H topology. The two zonal boundaries were placed at the span stations located at the ends of the control surface. Four additional virtual zones were placed in the two cuts separating the flap and the wing. One pair of the virtual zones remains fixed with the wing and the other pair is fixed with the flap and moves with the flap during the control surface motion. Figure 2 shows the seven zones used in this study.

The C-H grid around a deflected control surface can be obtained in two ways. One is to shear every grid line normal to the control surface with the local deflection. The other is to algebraically regenerate the entire C-H grid with the control surface deflected at every time step. A previous study [5], showed that the computed surface pressures did not show any significant differences between the two methods. Therefore, for this study, the grids around the control flap were regenerated with the simpler shearing method.

Virtual Zone

The zoning capability of the CNSFV code [6] allows the possibility of a single face of a zone to interact with several other zones. The procedure of determining the interpolation coefficients is automatic in that no additional information is required other than identifying the faces that are in contact with each other. To further extend the flexibility of the zoning method for the case of control surface aerodynamics, the idea of "virtual" zones was introduced in Ref. 7.

Virtual zones are zones of zero thickness (for a finite volume formulation) which serve to transfer solid wall (or other) boundary conditions to an interface condition. Thus multiple boundary conditions can be imposed on a block face with the same flexibility as an interface condition. Virtual zones also decouple the process of volume grid zoning from the surface grid patches which define the aerodynamic configuration under study. Surface grid patches are required in order to impose the proper boundary conditions. The volume grid zones should be set up to obtain the proper mesh qualities required for numerical accuracy. Another advantage of the decoupling is that much fewer zones are now needed, thus easing the effort and time required to generate the grids about complex and realistic configurations.

Perhaps the simplest way to explain the virtual zoning concept is through the use of a generic wing/flap configuration represented by two cylinders sliding past each other as shown in Fig. 3. In Fig. 3a(i) the two cylinders are in contact and aligned. A convential zoning scheme has no difficulties for this configuration. In Fig. 3a(iii) the two cylinders are completely separated from each other and again a convential zoning scheme is adequate. However a convential zoning scheme may not be possible at the instance where the two cylinders are in partial contact as in Fig. 3a(ii). The topology of the grids and zones in these three instances is not the same, and thus it is not possible to conveniently use the convential zoning procedure for such dynamic configurations.

On the other hand, as Fig. 3b illustrates, with the virtual zone concept the topology of the grids and zones does need to not change during the movement of the upper cylinder over the lower one. By covering the cut ends of the cylinders with a virtual zone so that the variable solid wall boundary conditions become interface conditions, the upper zone communicates with the lower zone only through the interface condition irrespective of the relative position of the two cylinders. As also shown in the figure, the grid topology of the virtual zones can be, and usually is, different from the topology of the volume grid in contact with the virtual zone. By this simple idea of converting boundary conditions into interface conditions through the use of virtual zones, dynamic configurations which were previously difficult to zone and grid with patched grids become tractable.

The virtual zones also allow the zonal boundaries to cut through the configuration surfaces, which is an important property of control surfaces. The region of the configuration that intersects the zonal face is covered with a virtual zone to convert that region into another interface condition. Once a zone has been defined along with its associated virtual zones, its definition is complete and is not influenced by any of its neighboring zones. In other words, the real zone communicates with the other zones (real or virtual) only through the interface conditions. Thus a particular zone can be altered or substituted with another zone without any need to redefine the interface conditions of the other zones. For example, a zonal grid can be set up for a wing with control flaps with one zone for each (say, undeflected) flap. For the deflected flap case, only the flap zone needs to be replaced with a zone containing a deflected flap. The boundary and interface conditons of all other zones remain unchanged, even though they may now have a solid surface exposed to the flow field, e.g., the edge of the exposed end of the wing and flap.

The original zoning capability of the ENSAERO code was extended by including the above capability of multiple interface conditions on a single block face. Since the code is a finite difference code the zones required an overlap at the boundaries of the zones to allow for the proper interblock communication (i.e., interfacing). In this study a one-cell overlap was chosen. For this kind of zoning the virtual zones of zero thickness used for the finite volume

formulation are not appropriate. Instead, the thickness of the virtual zones had to be expanded to include the extent of the overlap of the zones and, thus, the virtual zones for the present formulation are now one cell thick. This formulation results in a slight mismatch of one-half cell thickness between the location of the actual solid wall and the virtual zone boundary. The solid wall boundary condition is applied at the proper location and the slight mismatch has no discernable influence on the overall flow field, especially in the case where the ends of the flaps and wings are treated with the no-slip viscous boundary condition, rather than with the inviscid tangency condition.

An example of the virtual zones required for the present case is shown in Figs. 4-6 for the inboard end of the control surface. The two virtual zones slide through each other with the control surface motion. It can be seen in the figures that different regions of the virtual zones are then exposed to, or in contact with, the real zones surrounding the wing and the flap. The virtual zones transfer the solid wall boundary condition to an interface condition and thus allows for the automatic inclusion of the variable exposure of the ends of the flap and wing to the flow field. The area where the wing and flap virtual zones overlap represents the unexposed portions of the flap and wing. Since the flow field is not updated in the virtual zones by the flow solver, nothing happens in the virtual zone overlap region nor does it influence the rest of the flow field.

Zonal Interface Interpolation

The original interpolation procedure used in CNSFV and ENSAERO is based on a global area search. Even though it is highly vectorized, it still requires 5000 μsec per target point on a Cray YMP on an interface with 2500 points for both the target and base domains. In this paper a target point is the point to be interpolated using the data from the base points. The global area search is, thus, much too slow for a dynamic interpolation procedure where the interpolation coefficients have to be updated at every time step. By replacing the area search with a clipping search procedure based on a polygon clipping algorithm (also called window clipping, Ref. 11), the search time to find the interpolants is reduced by two orders of magnitude. This improvement brought the cpu run time for determining the interpolation coefficients down to the same level as required for the flow solver. In addition to the clipping search, the nested loop (or shell) search, directed hunt, and range limiting search are used to speed up the process and are described below.

The clipping search is based on the well-known polygon clipping algorithm. However, instead of clipping a polygon to fit inside a window, only one point designated the target points is used. The window is formed from the four base points and we are trying to find the four base points that surround the target point. In general the four base points do not form a rectangular window and the "coarse clip" circumscribes the four base points as shown in Fig. 7. The clipping proceeds as follows:

1. Circumscribe the four base points with a rough window.

2. Target point to the right of left boundary? If so, continue with 3. If not, go to next set of base points since target cannot be inside this window.

3. Target point above bottom boundary? If not, go to next set of base points; if so continue with 4.

4. Target point to the left of right boundary? If not, go to next set of base points; if so continue with 5.

5. Target point below top boundary? If not, go to next set of base points; if so target point inside the rough window and the "coarse clip" is successful, continue with 6.

6. Transform the four base points into a unit square aligned with the coordinate axis using a bilinear transformation, as sketched in Fig. 7. Locate the target point in the transformed space and repeat the search procedure 2-5. If this "fine clip" is successful then have located the four base points that surround the target point and can now go to 7.

7. Determine the bilinear interpolation coefficients for this target point and go to 8.

8. Proceed to next target point and go to 1.

The efficiency of the clipping search comes from two factors: the first is that non-candidate base points are rejected as quickly as possible with a minimal amount of operations performed. Secondly, the bilinear transformation is done only if the target point is inside the rough window. The bilinear transformation is relatively expensive to compute compared to setting up a rough window which requires only a few minmax function operations on the four base points, especially when the degenerate cases of the four base points collapsing into triangles must be considered.

In the best-case scenario where the first guess of the base points is the correct one, the Cray YMP cpu time per target point is 35 μsec for the above clipping search procedure. This includes the coarse clip, bilinear transformation, fine clip, and determining the interpolation coefficients. The worst case is when the target points are not within the domain of the base points. In this case assuming grids with 2500 points for both the target and base grids, about 600 μsec are required. The reason that the times are not larger is that only the coarse clip procedure needs to be executed. In spite of the slow times, the worst case times are about 10 times faster than the vectorized global area search procedure.

To obtain the times of the best case, it is clearly necessary to obtain the best guess possible for the base points so that only a small neighborhood needs to searched. For the steady-state problem a good starting guess for a new target point is the base points found from the previous target point. For unsteady cases, where the target grid is moving relative to the base grid, a good strategy for the starting base points is to use the base points found for this target point at the previous time step. For highly clustered meshes where a target point can move across several dozen base points during a time step, a better approach is to use the one for the steady state case. In this paper, however, we use the previous time step guess. With the starting guess for the base points, two procedures called the "shell search" and "directed hunt" are used with variable success. A third method called "range limiter" is developed to reduce the time required for the worst case or near worst case situation which happens quite often for the oscillating control surface configuration. All three methods are described below in further detail.

The shell search uses the clip search as the basic engine. The shell search refers to the sequence in which the four base points are tested against the clip search. The innermost shell depicted as $n = 1$ in Fig. 8 is the starting guess for the base points. If the clip test is unsuccessful, then the search proceeds to the next shell and loops around the inner shell until the clip test succeeds or until all the $8(n-1)$ sets of base points are tested. If again unsuccessful, the search proceeds to the next shell. The number of tests increases as $(2n-1)^2$ where n is the shell number. If the shell intersects a boundary of the base grid, then the shell is truncated at that boundary. The maximum number of shells searched is equal to the maximum dimension of the base grid. If the guessed starting set of base point is not good, then the shell search is not much better than searching the entire base grid with the clip search. However, if the starting guess is good and the target points have not moved across too many base points, then the shell search is successful within 2 or 3 shells, requiring at most 9 or 25 clip searches. Tests with various cases have shown that it is more efficient to stop after the second shell and do the range limiter (explained below) before proceeding with the next shell. Typical test cases show that the shell hunt cpu times vary from 35 μsec to 150 μsec per target point.

A potentially more efficient procedure than the shell search is the directed hunt. In this method, the best guess for the four base points and the target point is transformed into the unit square window. Let the transformed coordinates of the target point be (k_t, l_t) as shown in Fig. 9. If the coordinates are positive and less than one, the four base points have been found; if not, the next guess for the reference base point coordinates is given by

$$k_n = k_o + int(k_t - 1)$$

$$l_n = l_o + int(l_t - 1)$$

where k_o, l_o are the indices of the reference point of the four base points, as shown in Fig. 9. Typically, the directed hunt should find the base points with only 2 to 3 steps. However the bilinear transformation often breaks down because of degenerate base points, or the direction takes the search to the outside of the base domain boundaries (e.g., in crossing the wake boundary of a C-mesh). In the latter case one has to step along the boundary until the directed hunt can be continued. For these reasons the directed hunt has not been reliable and efficient enough for the oscillating control surface problem. Since the directed hunt has the potential of being 2 to 3 times faster than the shell search, especially if the first guess is poor, work is continuing on developing a reliable directed hunt procedure.

The cost of computing the searching procedure increases as the number of base points increases. What is needed is an efficient procedure to reduce the number of base points over which to search, or to eliminate them entirely if the target point is not within the domain of the base points as in the worst-case scenario mentioned above. The "range limiter" was developed for this purpose. As with the shell search the basic engine is again the clip test. The range limiter can be described with the help of Fig. 10 as follows:

1. Circumscribe the domain of the base points with the window shown in Fig. 10a.

2. Apply the clip test to the target point; if target is inside window, continue with 3; if not, target point cannot be interpolated with the given base points. Go on to next target point.

3. Subdivide the base domain in half and choose the subdivision which produces the smallest overlap area as shown in Fig. 10b and 10c. If overlap areas are equal, choose the partition which operates on the larger of the two dimensions; if the dimensions are the same, use the first of the two subdivisions.

4. Apply the clip test to the target point with both of the subdomains. If the target point is in one but not the other of the subdomains, then define the domain of base points to be the domain in which the target point lies and continue with 1. If the target point lies in both of the subdomains (i.e., it lies in the overlap area) then cut off the two subdomains at the ends opposite the overlap region and redefine the domain to be the central portion of the base point domain and continue with 1. The clip test is applied to the cutoff portions to make sure that the trimmed portions do not contain the target point.

5. The subdivision along any one direction stops if the dimension in that direction is less than 5, since at that point the shell search is more efficient.

Although the range limiter appears cumbersome, it is quite useful and efficient especially if the domain of base points is larger than the domain of target points, and it very quickly excises the portions of the base domain which are not close to the target point. Once the base domain has been trimmed, the shell search or directed hunt is used for the local search. Another way of trimming the base domain is to place search limits on the base domain beyond which the search will not be conducted. This last method is effective only if the search limits are known a priori, as for the present case of the oscillating control surface.

It is of interest to compare the present method with the so-called Domain Connectivity Function method of Ref. 12. The present method is just slightly faster than the dynamic mode of the DCF method. Compared to the static mode of the DCF method, where the inverse mapping needs to be determined, the present method is substantially faster (approx. 10 times). For the oscillating control surface problem, the DCF method would have to be run in the static mode and thus is too slow to be pratical.

Results

The test case considered in the present study is a clipped delta wing with an oscillating trailing edge control surface [13]. The wing planform is shown in Fig. 1. The wing has a leading edge sweep angle of 50.4 deg and a 6% thick circular arc airfoil section. At $M_\infty = 0.9$ and $\alpha = 3$ deg, both a leading edge vortex and a shock wave are present on the upper surface of the wing. The C-H grids of the three real zones consist of $151 \times 13 \times 34$, $151 \times 15 \times 34$, and $151 \times 20 \times 34$ points from inboard to outboard as shown in Fig. 2. Since the experiment was conducted using a Freon test medium, the ratio of specific heats, γ, is set to 1.135 in the present computations. As stated before, the modified Baldwin-Lomax model is used to account for the leading edge and control surface vortices. However, the boundaries normal to the edges of the control surface

and the wing cuts are treated with the solid wall tangency condition. Steady state and rigid pitching calculations of this wing were reported in Ref. 8.

Figure 11 shows the unsteady pressures coefficients with the control surface oscillating at a frequency of 8 Hz and an amplitude of 6.65 deg at $M_\infty = 0.9, \alpha = 3$ deg and $Re_c = 17 \times 10^6$ based on the root chord. The results are shown as the amplitude and phase angle of the upper surface pressure coefficients at the three span stations as indicated on the figure. In general, the agreement with the experimental results is good. Because the accuracy of experiment has its own limitations, the virtual zone results are also compared with the single grid results of Ref. 5. There is quite a discrepancy between the virtual zone results and the single grid results (the $151 \times 44 \times 34$ grid), especially in the amplitudes at the center of the control surface. This discrepancy is most likely due to the gap that was introduced between the flap and wing in the single grid case to accommodate the shearing grid. If the gap is reduced by increasing the spanwise resolution of the wing and control surface, the computational results of the refined single grid case ($151 \times 87 \times 34$) approach those of the virtual zone. This particular example demonstrates the importance of simulating the geometry of the control surface/wing configuration accurately.

Figure 12 shows the upper surface pressures as well as the instantaneous streamline traces emanating from the leading edge of the wing (black traces), the lower edges of the wing at the control surface cut (blue traces), and the upper edges of the control flaps (red traces). The interaction between the leading edge vortex and the outboard edge of the control surface vortices during the flap motion cycle is well demonstrated. Although not apparent in this figure, the computations also show that the flow on the leading side (defined as upper surface during the upstroke motion and lower surface during the downstroke motion) of the inboard section of the control surface is separated.

Figure 13 shows the velocity vectors at the inboard edge of the control surface as the flap is in the upstroke mode at the instant of zero deflection. The flow field is shown in much more detail at the trailing edge of the control surface. The detailed views show the velocities of the trailing edge of the control surface. As can be seen, these velocities are small, even when compared to the flow field in the inner regions of the boundary layer. Figure 14 shows the surface streamlines on the inboard edge and upper surface of the control surface. The particle traces are computed at each instant of time by freezing the flow field during the tracing procedure. The temporal changes in the surface particle traces as the control surface oscillation cycle is executed are quite apparent, both on the upper surface and the edge of the control surface as it is exposed and covered by the wing edge. The first view of Figure 14 shows that the flow is separated near the inboard trailing edge corner of the control surface. Preliminary studies with a viscous (albeit laminar) treatment of the control surface edges and wing cuts indicate that the separated region on the upper surface of the flap increases.

Conclusions

An unsteady interface algorithm based on the idea of virtual zones has been developed for a finite difference code, ENSAERO. The new "virtual" zoning technique simplifies the zoning of complex geometries, such as control surfaces, and makes possible the use of standard multizonal codes for complex configurations. A fast search routine based on a window clipping algorithm has also been developed and is fast enough for the interpolation coefficients to be recomputed at every time step. For the example presented in this study, the computational effort required for the interpolation coefficients is less than that for the flow solver.

Both of the above developments have made practical a complete unsteady Navier-Stokes simulation of a forced oscillating control surface on a clipped delta wing in the transonic flow regime. The method has been validated against experimental data as well as numerical simulations based on a shearing single-zone grid. The numerical result confirms that the accurate representation of geometry by "virtual zones" is superior to the single zone computations of the same grid size.

Acknowledgements

The first author's work was supported by NASA Grant NCC 2-616 and the second author's by NCC 2-605.

References

1. Ballhaus, W., Goorjian, P. M., and Yoshihara, H.; "Unsteady Force and Moment Alleviation in Transonic Flow," AGARD Conference Proceeding No. 227, May 1981.

2. Guruswamy, G. P. and Tu, E. L.; "Transonic Aeroelasticity of Fighter Wings with Active Control Surface," Journal of Aircraft, Vol. 26, No. 7, July 1989, pp. 682-684.

3. Steger, J. L. and Bailey, H. E.; "Calculation of Transonic Aileron Buzz," AIAA Journal, Vol. 18, March 1980, pp. 249-255.

4. Horiuti, K., Chyu, W. J., and Buell, D. A.; "Unsteady Transonic Flow Computation for an Airfoil with an Oscillating Flap," AIAA Paper 84-1562, June 1984.

5. Obayashi, S. and Guruswamy, G.; "Navier-Stokes Computations for Oscillating Control Surfaces," AIAA Paper No. 92-4431, August 1992.

6. Klopfer, G. H. and Molvik, G. A.; "Conservative Multizonal Interface Algorithm for the 3-D Navier-Stokes Equations," AIAA Paper No. 91-1601CP, June 1991.

7. Chaussee, D. S. and Klopfer, G. H.; "The Numerical Study of 3-D Flow Past Control Surfaces," AIAA Paper No. 92-4650, August 1992.

8. Obayashi, S. and Guruswamy, G.; "Unsteady Shock-Vortex Interaction on a Flexible Delta Wing," Journal of Aircraft, Vol. 29, No. 5, Sept-Oct 1992, pp. 790-798.

9. Baldwin, B. S. and Lomax, H.; "Thin-Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-257, January 1978.

10. Degani, D. and Schiff, L. B.; "Computations of Turbulent Supersonic Flows Around Pointed Bodies Having Crossflow Separation," Journal of Computational Physics, Vol. 66, No. 1, September 1986, pp. 173-196.

11. Pavlidis, T.; **Algorithms for Graphics and Image Processing**, Computer Science Press, Inc., MD, 1982.

12. Meakin, R. L.; "A New Method For Establishing Inter-Grid Communication Among Systems of Overset Grids," AIAA Paper No. 91-1586CP, June 1991.

13. Hess, R. W., Cazier, F. W., Jr., and Wynne, E. C.; "Steady and Unsteady Transonic Pressure Measurements on a Clipped Delta Wing for Pitching and Control-Surface Oscillations," NASA TP-2594, Oct. 1986.

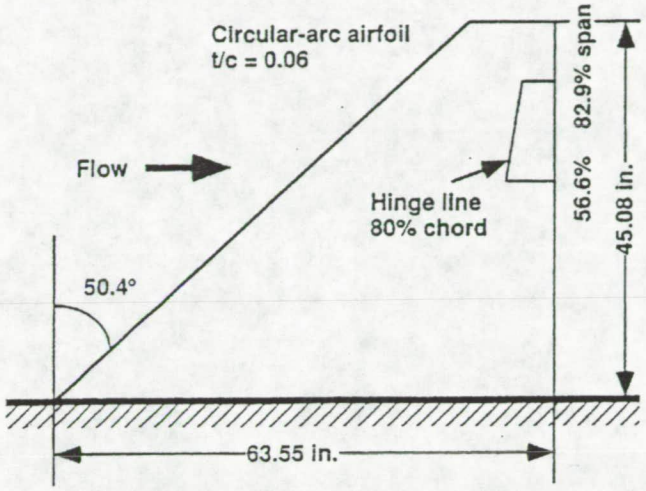


Fig. 1. Planform of clipped delta wing with trailing edge flap.

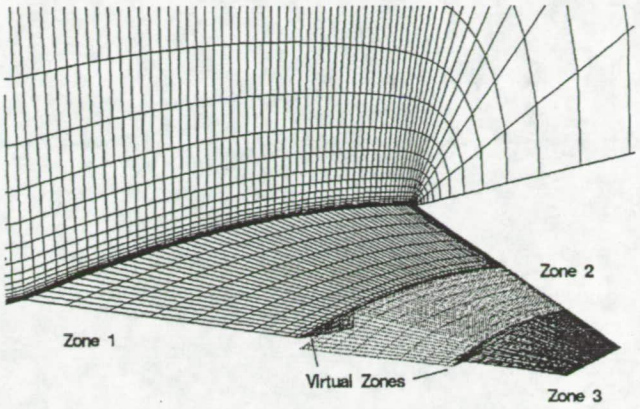


Fig. 2. Surface grids and zonal boundaries of wing/flap configuration.

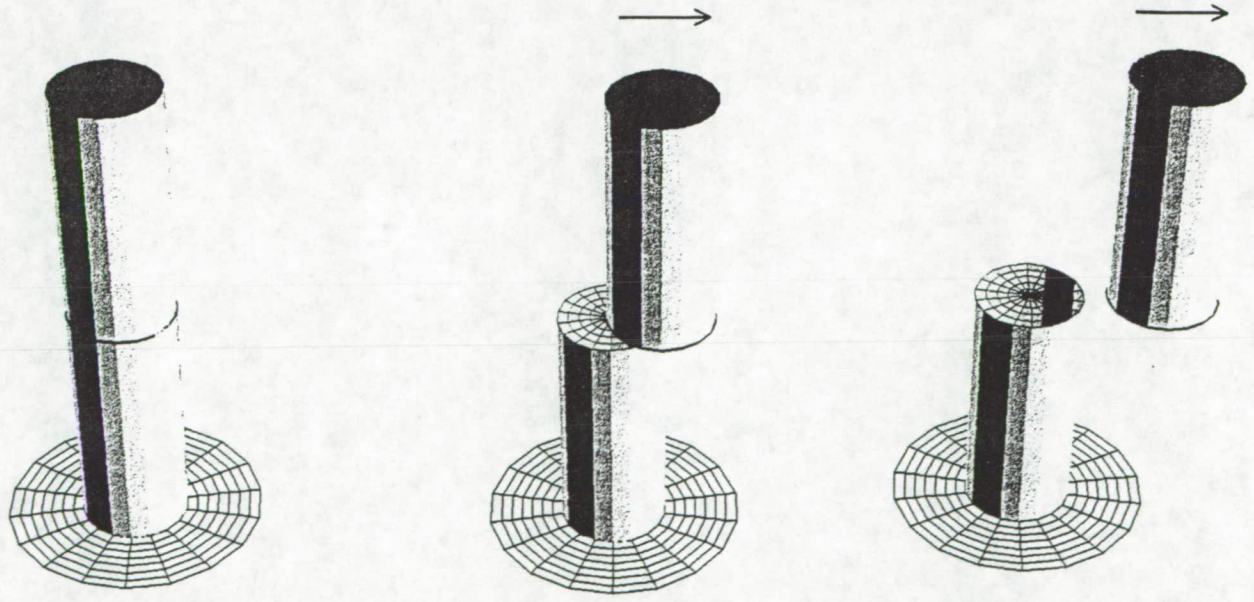


Fig. 3a. Generic wing/flap configuration with conventional zones. The top cylinder is sliding over the face of the bottom cylinder.

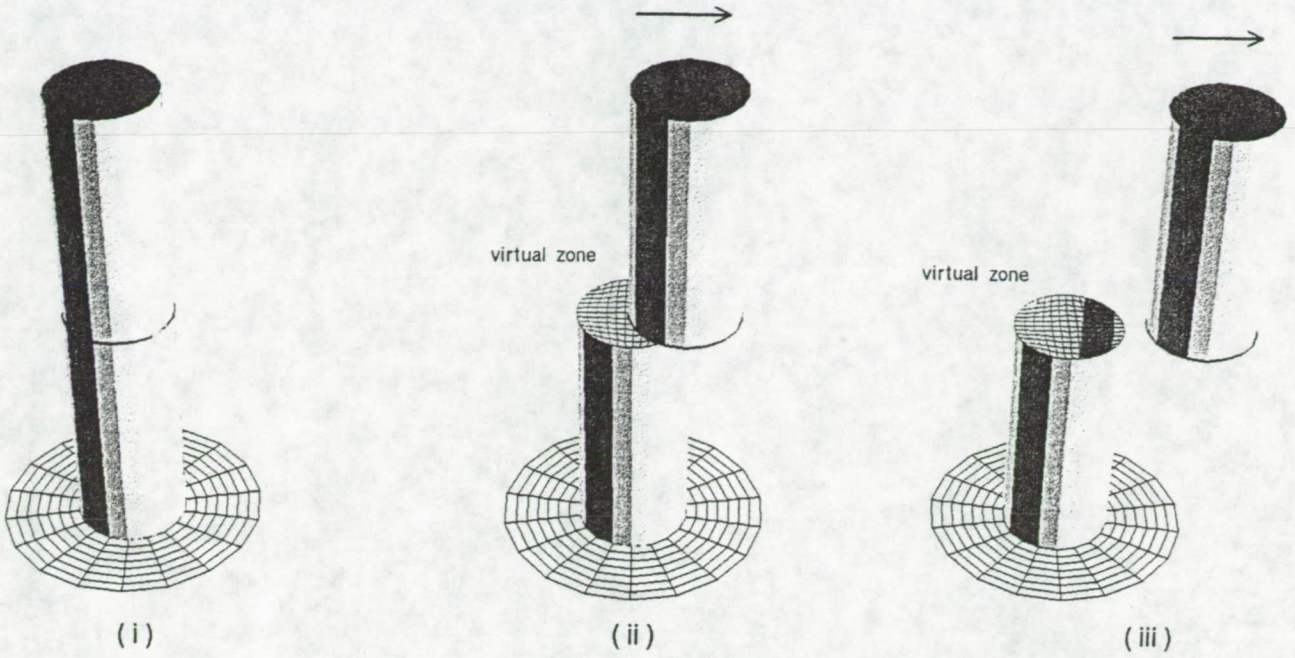


Fig. 3b. Generic wing/flap configuration demonstrating the virtual zone concept. The top cylinder is sliding over the face of the bottom cylinder.

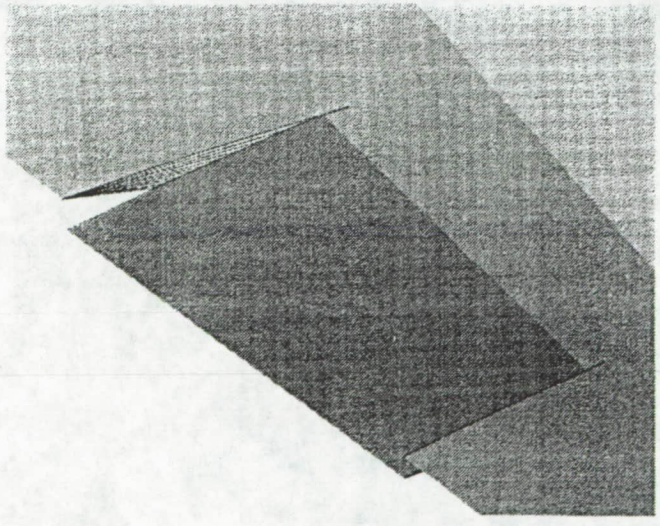


Fig. 4. Perspective view of trailing edge flap and wing.

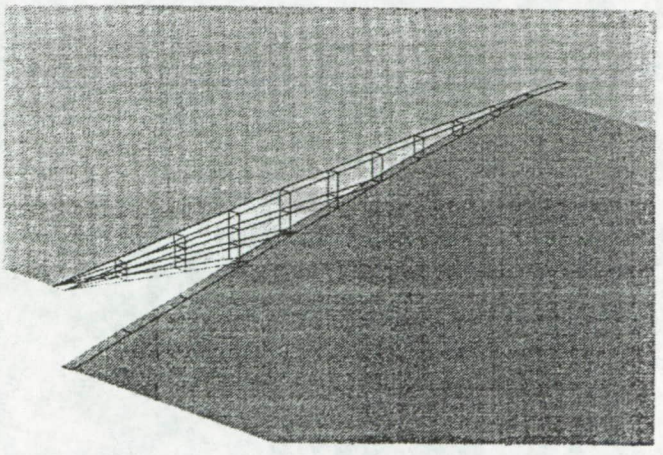


Fig. 5. Perspective view of trailing edge flap and wing with the inboard wing and flap virtual zones.

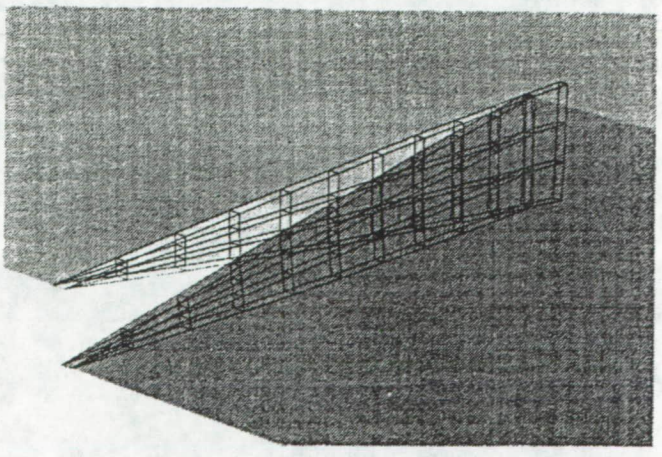


Fig. 6. Perspective phantom view of trailing edge flap and wing with the inboard wing and flap virtual zones.

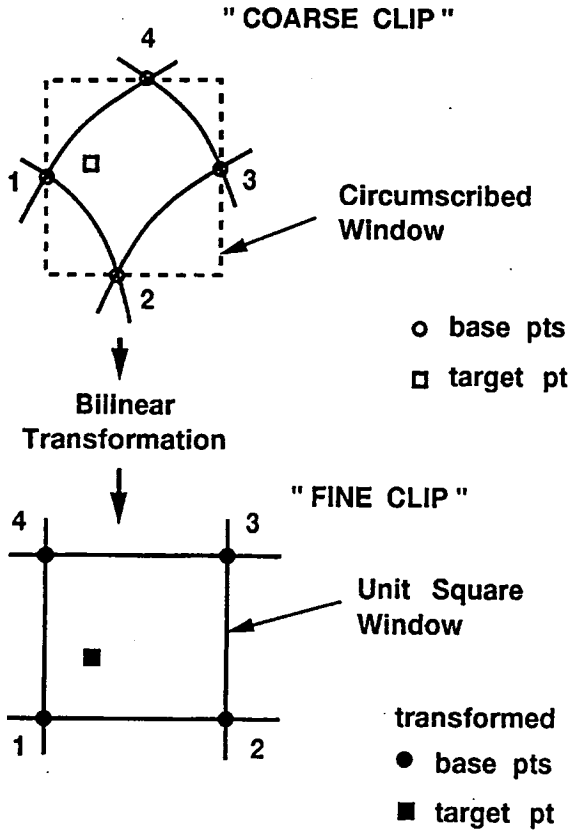


Fig. 7. Clip search procedure; Coarse clip, bilinear transformation and fine clip test.

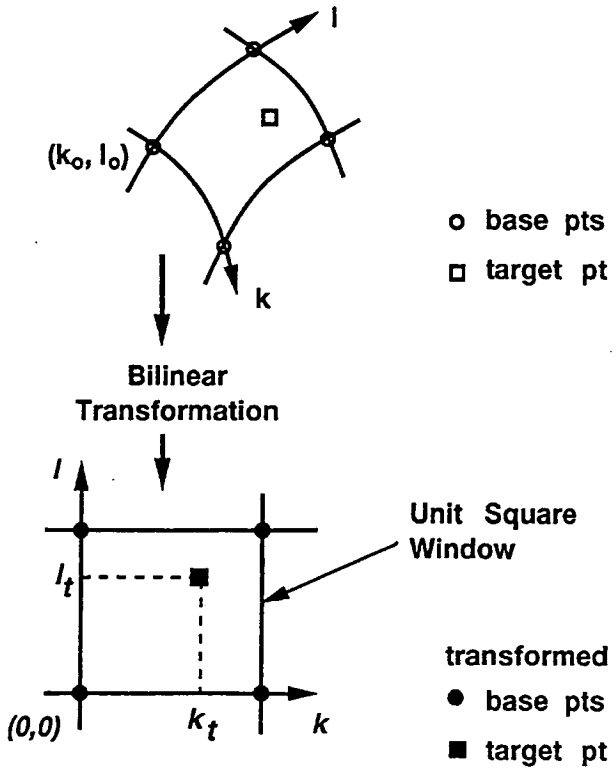


Fig. 9. The directed hunt procedure.

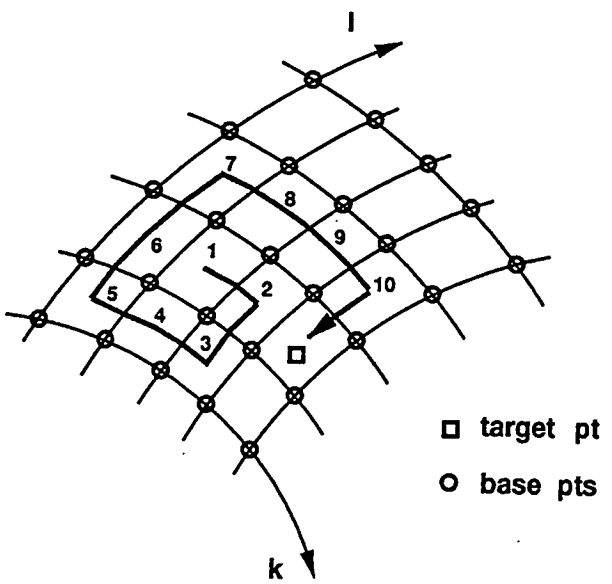


Fig. 8. The sequence of steps required for the shell search procedure.

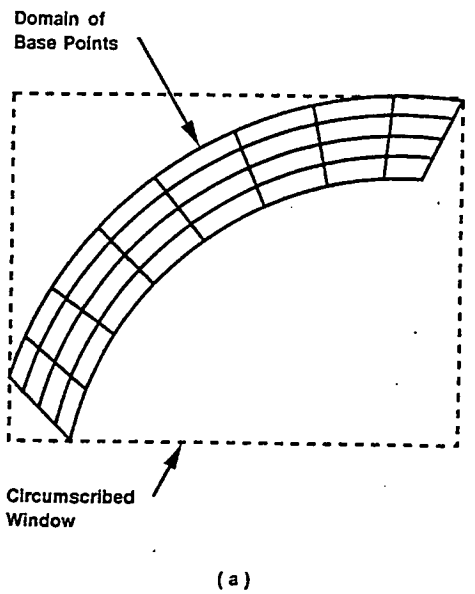


Fig. 10. The range limiter procedure.

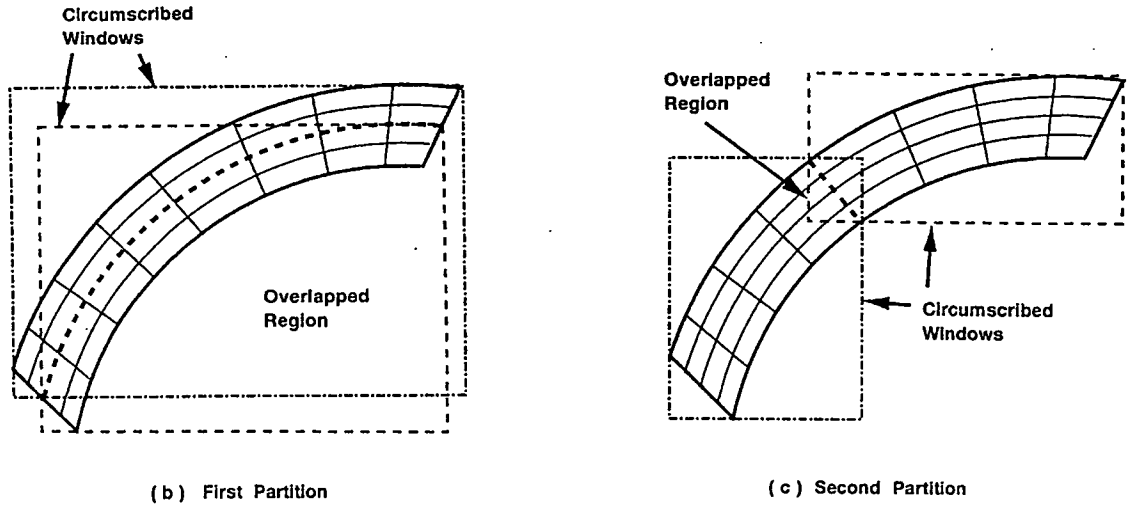


Fig. 10. The range limiter procedure.

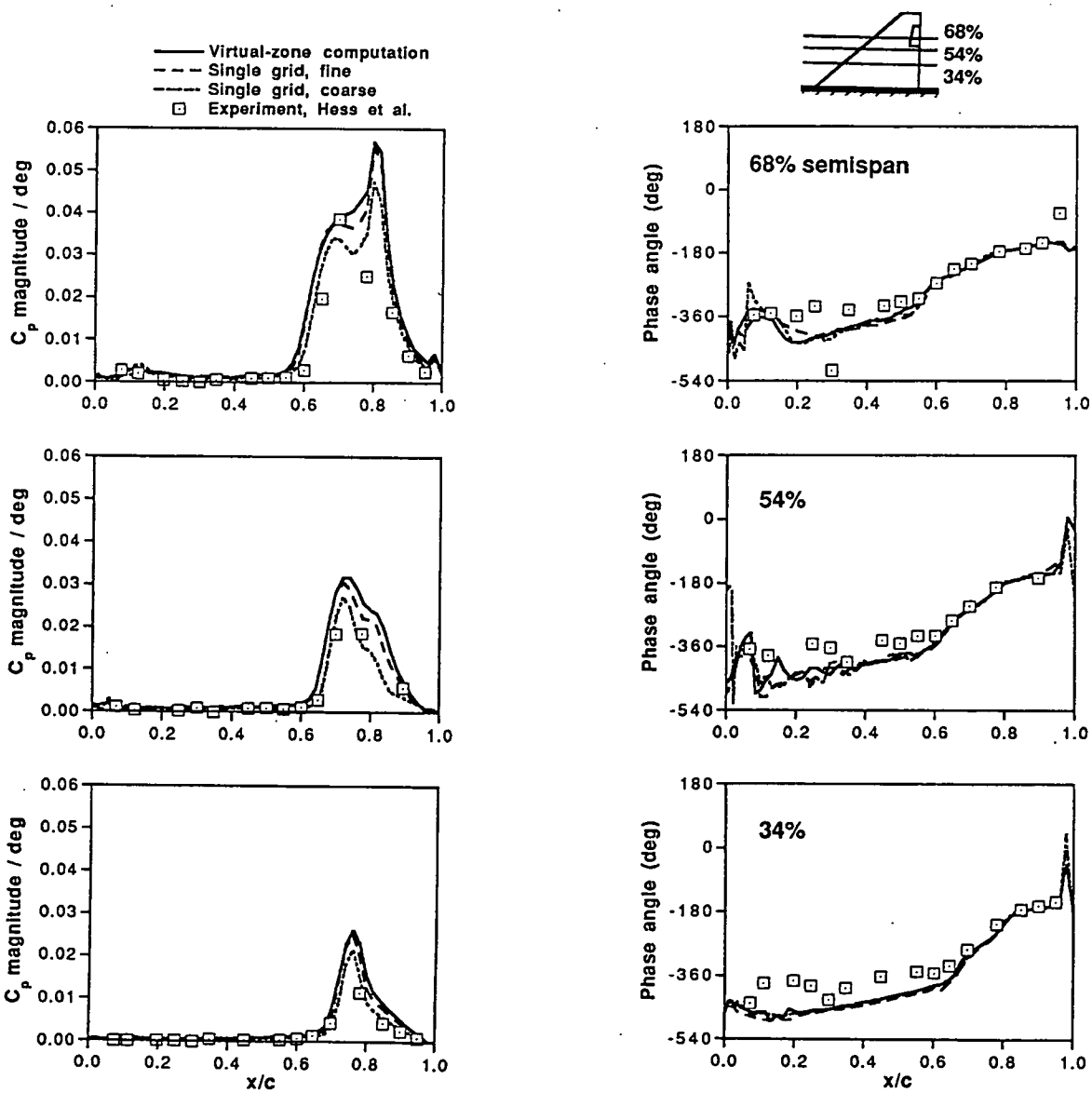


Fig. 11. Comparison of unsteady pressure coefficients between experiment (Ref. 13) and computations using virtual zones and single grids.

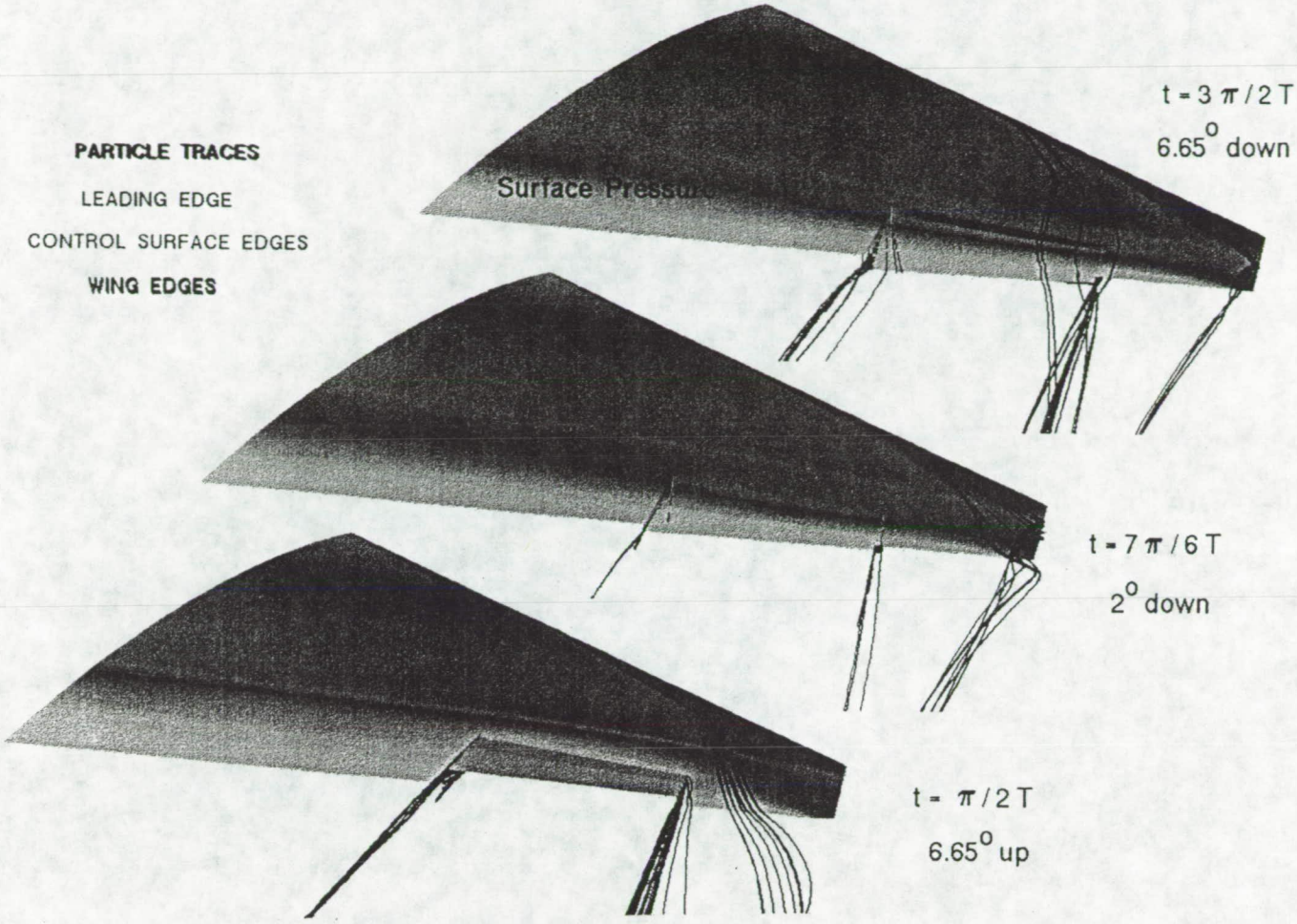


Fig. 12. Upper surface pressures and instantaneous streamlines emanating from the leading edge and the edges of the control surface.

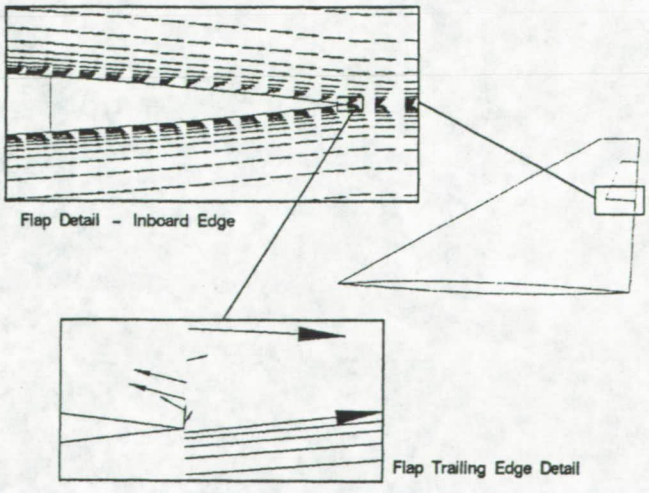


Fig. 13. Velocity vector plots near the inboard edge of the control surface at the instant of zero deflection and maximum flap velocity.

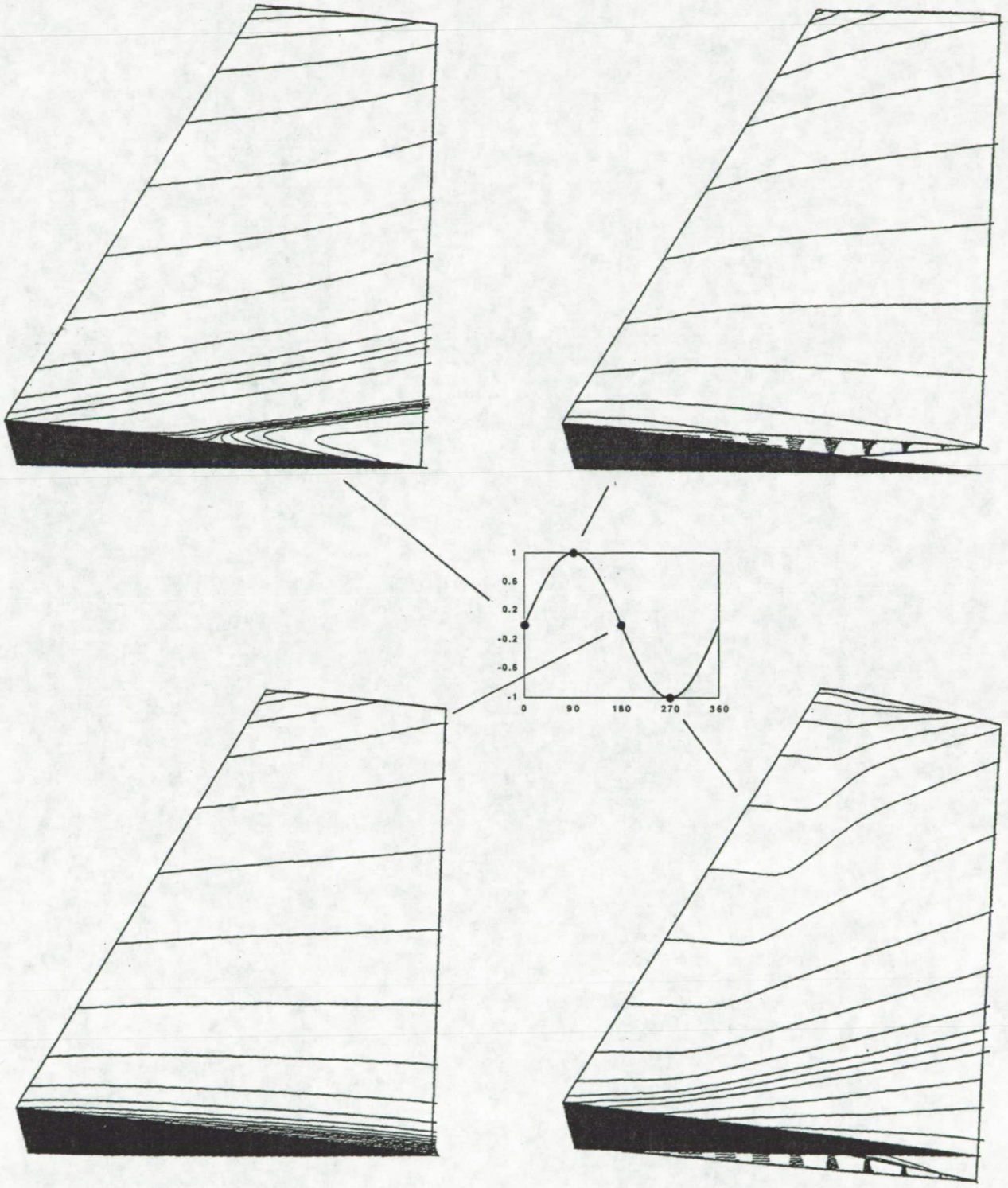


Fig. 14. Surface particles traces on the upper surface and inboard edge of the control surface at four instances of the control surface cycle.