Winter 1999

Efficient Dynamic Unstructured Methods and Applications for Transonic Flows and Hypersonic Stage Separation

Xiaobing Luo
Old Dominion University

EFFICIENT DYNAMIC UNSTRUCTURED METHODS AND APPLICATIONS FOR TRANSONIC FLOWS AND HYPERSONIC STAGE SEPARATION

Xiaobing Luo

B.E., July 1987, Aircraft Design and Applied Mechanics Department
Beijing University of Aeronautics and Astronautics, P.R.China

M.S., March 1990, Aircraft Design and Applied Mechanics Department
Beijing University of Aeronautics and Astronautics, P.R.China

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

AEROSPACE ENGINEERING
OLD DOMINION UNIVERSITY

December 1999

Approved by

Oktay Baysal (Director)

Thomas E. Alberts

Osama A. Kandil

David E. Reubush (NASA)

ABSTRACT

EFFICIENT DYNAMIC UNSTRUCTURED METHODS AND APPLICATIONS FOR TRANSONIC FLOWS AND HYPERSONIC STAGE SEPARATION

December 1999

Xiaobing Luo

Director: Dr. Oktay Baysal

Relative-moving boundary problems have a wide variety of applications. They appear in staging during a launch process, store separation from a military aircraft, rotor-stator interaction in turbomachinery, and dynamic aeroelasticity.

The dynamic unstructured technology (DUT) is potentially a strong approach to simulate unsteady flows around relative-moving bodies, by solving time-dependent governing equations. The dual-time stepping scheme is implemented to improve its efficiency while not compromising the accuracy of solutions. The validation of the implicit scheme is performed on a pitching NACA0012 airfoil and a rectangular wing with low reduced frequencies in transonic flows. All the matured accelerating techniques, including the implicit residual smoothing, the local time stepping, and the Full-Approximate-Scheme (FAS) multigrid method, are resorted once a dynamic problem is transformed into a series of "static" problems. Even with rather coarse Euler-type meshes, one order of CPU time savings is achieved without losing the accuracy of solutions in comparison to the popular Runge-Kutta scheme. More orders of CPU time savings are expected in real engineering applications where highly stretched viscous-type meshes are needed.

The applicability of DUT is also extended from transonic/supersonic flows to hypersonic flows through special measures in spatial discretization to simulate the staging of a hypersonic vehicle.

First, the simulations in Mach 5 and Mach 10 flights are performed on the longitudinal symmetry plane. A network of strong shocks and expansion waves are captured. A prescribed two-degrees-of-freedom motion is imposed on the booster and the adapter to mimic the staging.

Then, a 3-D static Euler solver with an efficient edge-based data structure is modified for time-accurate flows. The overall history of aerodynamic interference during the staging in Mach 5 flight is obtained by an animation method, consisting of six static solutions along the assumed stage path. From the animation method, the following conclusions are made. After the booster and the adapter move away from the research vehicle by 60% vehicle length, their effects on the research vehicle are confined to the wake flow of the research vehicle. The aerodynamic forces on the research vehicle converge to the values in free flight when the booster is away from the research vehicle by 1.77 times vehicle length. The aerodynamic interference is a highly nonlinear function in terms of the distance between the vehicle, the booster, and the adapter.

Finally, two dynamic computations are performed when the booster and the adapter are extremely close to the research vehicle. It is observed from these 3-D dynamic computations that as the stage separation advances, the aerodynamic interference becomes less sensitive to further relative motions.

To my parents, wife, daughter and son

ACKNOWLEDGMENTS

I would like to take this opportunity to express my appreciation to my advisor and research director, Dr. Oktay Baysal, for his valuable guidance and support during my four years at Old Dominion University.

I also would like to thank my other committee members, Dr. Osama A. Kandil, Dr. Thomas E. Alberts, and Mr. David E. Reubush, for their valuable suggestions and comments on my dissertation.

Thanks are also due to Dr. S. Pirzadeh and Dr. K.P. Singh for providing some of the grids used in this study. I would like to extend my heart-felt thanks to my graduate colleagues for their long time concerns and valuable discussions.

I would like to express my deepest appreciation, love and gratitude to my parents and wife for their constant support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

a : speed of sound

a_t : contravariant face speed

A_k : face area

A_j^x, A_j^y, A_j^z : projection of area of cell i onto x, y, and z directions, respectively

A_c^* : idea area

CFD : computational fluid dynamics

CFL : Courant number

C_A : axial force coefficient

C_L : lift coefficient

C_N : normal force coefficient

C_p : pressure coefficient

CPU : central processing unit

DUT : dynamic unstructured technology

e_0 : total energy

f : frequency

F : cost function in area correction

FAS : full-approximation-storage scheme

$\tilde{F}$: flux vector in cartesian coordinates

$\hat{F}^{\pm}$: split flux terms in van Leer scheme

$F_n^{\pm}$: split flux terms in Hänel scheme

GCL : geometric conservative low

h_0 : total enthalpy

I_h^{2h} : restriction operator

I_{2h}^{h} : prolongation operator

k_{ij} : spring stiffness

k : reduced frequency

L : length of vehicle

m_r, m_p : highest degree of polynomials plus one on the construction of restriction and prolongation operators

M : mass matrix or ratio of CPU in implicit scheme to explicit scheme

M_∞ : freestream Mach number

$\hat{n}$: outward unit normal vector

N_j : shape function

N_i : connectivity at node i

p : pressure

$\vec{Q}$: vector of conservative variables

∇Q : gradient vector of variables

Q_n : values of variable at mesh node n

$\vec{q}$: vector of primitive variables

q_∞ : dynamic pressure

R_i : residual

R^* : pseudo residual

$R^\pm$: Riemann invariants

$\vec{r}$: position vector

Δr : radius vector

S : surface area

x_t, y_t, z_t : velocities of grid nodes

$\Delta x, \Delta y, \Delta z$: displacements of nodes in spring analogy

t : time

Δt : physical time step

Δt^* : pseudo time step

$\Delta t_i, \Delta t_e$: time step in implicit and explicit scheme

u, v, w : Cartesian velocities

U_n : contravirant velocity

V : volume

$\vec{V}$: velocity vector

v : error in nonlinear system

W : weighting function or an approximate unknown

Greek symbols

α : angle of attack in degrees

α_0 : amplitude

ρ : non-dimensional density

γ : specific heat ratio

ω : angular velocity

θ : angle between normal

κ : constant in van Albada limiter

ε : constant in van Albada limiter, and presmoothing of dynamic mesh

Ω : control volume in finite volume method

Γ : boundary surface of control volume

CHAPTER 1*

INTRODUCTION

1.1 Motivation

There is a strong need to develop robust, efficient, and accurate methods for unsteady flow simulations, since numerous engineering applications are unsteady in nature. At this moment, Computational Fluid Dynamics (CFD) does not have sufficient methods for unsteady flows. In response, Baysal and his students [1-3] developed a new dynamic unstructured technique (DUT) to simulate unsteady flows around single or multi-body objects in relative-motion. The baseline method employed the popular explicit multi-stage Runge-Kutta scheme to perform the integration in time domain. An explicit scheme may be appropriate for some unsteady applications with small time domain. However, when dealing with many low frequency phenomena or problems with large time domain, explicit schemes may require long CPU times, which can easily make this method almost impractical. It is necessary, therefore, to derive new algorithms capable of dealing with the above problems efficiently, while not compromising the accuracy of solutions.

There is also a necessity to widen the applicability of the previous DUT from subsonic, transonic, and supersonic flows to hypersonic flows. For example, in the NASA sponsored Hyper-X hypersonic research program, the research vehicle separates from its launch vehicle during the staging. A significant interference among the launch vehicle and the research vehicle is expected. This kind of interference has large effects on the

* The style of this dissertation is that of AIAA journal

safety of the staging and the finial altitude of the research vehicle. Using dynamic unstructured technique (DUT) to simulate the staging of a hypersonic vehicle is another impetus for the present research. However, the previous DUT mainly focuses on transonic flows, and its direct applications in hypersonic flows will face severe numerical stability problems.

1.2 Objectives of Present Work

The main objectives of the present research are to develop a robust and efficient computational fluid dynamics method to simulate transonic flows and to widen the applicability of the previous DUT from subsonic, transonic, and supersonic flows to hypersonic flows to simulate the hypersonic flows past objects in relative-motion. The specific steps aimed to achieve this goal are given below:

(1) Developing an efficient implicit scheme for DUT: the dual-time stepping scheme implemented for the above purpose will be validated by solving unsteady transonic flows around an airfoil and a wing, undergoing a prescribed pitching motion with a low reduced frequency.

(2) Developing accelerating techniques to improve the efficiency of the implicit scheme: local time stepping, residual smoothing, and multigrid method are implemented to accelerate the convergence rate of subiterations in the dual-time stepping scheme.

(3) Modifying the existing dynamic mesh module [1], based on the spring-analogy method, to improve the quality of dynamic meshes when large time steps are used in an implicit scheme.

(4) Widening the applicable range of an existing unsteady solver [1] from transonic flows to hypersonic flows, and simulating the staging of a hypersonic vehicle on the longitudinal symmetry plane, that is, two-dimensional simulation.

(5) Modifying a static 3-D solver, which is based on an efficient edge-based data structure [4], into a dynamic flow solver, and simulating the staging of a hypersonic vehicle in three dimensions.

1.3 Outline of Dissertation

The contents of this dissertation are organized as follows. Chapter 2 presents the literature survey in the area of dynamic CFD. The details of computational methodology with upwind and central difference schemes are presented in Chapter 3. These details include governing equations, spatial discretizations with high-order accuracy on unstructured meshes, temporal discretization by an explicit scheme and a dual-time stepping scheme, geometric conservation law for a deformed dynamic mesh, acceleration techniques for steady solutions, initial and boundary conditions, and the dynamic mesh using the spring analogy method.

Chapter 4 describes the validations of the dual-time stepping scheme in transonic flows through simulating unsteady flows around the pitching NACA0012 airfoil and the pitching ONERA M6 wing with a low reduced frequency. Results in the implicit scheme are compared with those in the explicit scheme in terms of the accuracy of solutions and CPU times. The accuracy of the solutions in the explicit scheme is verified by available experimental data.

In Chapter 5, the staging of a hypersonic research vehicle is simulated on the longitudinal symmetry plane using the techniques described in Chapter 3. A two-degrees-of-freedom motion is imposed on the adapter jaw and the booster in order to obtain the history of aerodynamic interference in the staging process. Full three-dimensional simulations of hypersonic vehicle staging are carried out and are also presented in Chapter 5.

Conclusions, based on the present research, are presented in Chapter 6. Some recommendations for future work are also included in this chapter.

CHAPTER 2

LITERATURE SURVEY

2.1 Unstructured Mesh Methods

Significant efforts have been expended to simulate unsteady flows around a complex geometry. The following types of grids have shown their capability to handle these kinds of problems: overlapped (Chimera) meshes; multiblock and zonal meshes [5]; adaptive Cartesian meshes [6]; and unstructured meshes [7-10].

For problems with a complex geometry, the structured methods require such sophisticated strategies as blocked, patched, overlapped or hybrid meshes. Therefore, the main issue of applying structured meshes in relative-motion problems is the conservative interpolation of fluxes among patches. An unstructured method can always guarantee a dramatically decreased time, which is required to create a grid over a complicated geometry. Flow features can be captured with good accuracy via the mesh refinement or the ideal distribution of mesh nodes. The Interpolation of fluxes is not necessary when unstructured methods are applied in relative-motion problems. The main reasons for the popular usage of unstructured meshes to discretize complex computational domains are given below:

1). Since triangles in 2-D and tetrahedra in 3-D are the simplest geometrical elements possessing an area or a volume, unstructured meshes have the capability to discretize irregularly shaped domains more efficiently.

2). Since unstructured meshes do not have a regular connectivity, they lend themselves

naturally to adaptation.

However, the solvers on unstructured meshes are generally less efficient than their structured counterparts [7]. On the basis of per mesh node, unstructured methods have typically a factor of 2 to 3 increases in running memory and CPU times.

The methods of generating an unstructured mesh can be roughly cast into three categories: 1) triangulation of structured meshes; 2) Delaunay triangulation; and 3) advancing-front method. Although the first method is simple, it does not completely exploit the advantages offered by unstructured meshes. In the Delaunay method, a series of nodes are provided, and these points are connected to form unstructured meshes, based on the principle of Delaunay. The meshes generated by this method are able to minimize the discretization errors in a finite element computation [11]. The problem is how to give an ideal initial distribution of mesh nodes. In the advancing-front method, boundary surfaces, such as inner bodies and far-field surfaces, are triangulated to construct initial fronts, then a grid is generated by forming cells, starting from these fronts and marching into the interior of the computational domain. Figure 1.1 presents the process of the mesh generation. By controlling the strength and distribution of sources on the background grid, a highly clustered unstructured mesh can be obtained. The mesh generator [12], based on this method, was used to create all the meshes in the present research.

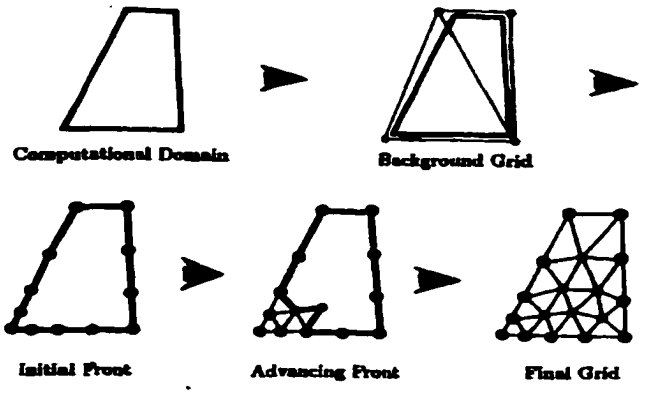


Figure 2.1 Advancing-front process for a simple 2D domain

2.2 Methods for Relative Moving Boundary Problems

In computing unsteady flows around a multi-component configuration with one or more components engaged in relative motions, there are four levels of assumptions that can be made for the incident flow and solid-surface interaction [13]. From the least to the most accurate, they are classified as follows:

1) All the moving components are instantaneously frozen, and at each instant, either a steady state or unsteady computation is performed. Luo and Baysal [14] used this method to compute the aerodynamic interference in the staging of a hypersonic vehicle. This method will be explained in Chapter 5.
2) All the moving components are assumed to engage in the same rigid-body motion, which also is assigned to the whole mesh.
3) Each moving component is assigned a known rigid-body motion [15-19]. The method will be explained in Chapter 4.

4) Beyond level 3, the trajectories of moving components are determined by instantaneous flow field and the principles of rigid-body dynamics, i.e., aerodynamically determined motion [1].

There are different methods that have successfully been developed to simulate the relative-moving boundary problems, which can be cast-based on the reference frame (Eulerian or Lagrangian) or the type of meshes (structured or unstructured).

Löhner et al. [20] successfully used Lagrangian methods to solve the moving boundary problems. In this approach, mesh nodes were fixed with flow particles, and each cell was associated with the same fluid element. This method has three advantages: 1) it is much easier to track material interface; 2) the enforcement of interfacial boundary conditions is simple; 3) the absence of numerical diffusion reduces the numerical errors associated with it. The main drawbacks of this approach are that the strong mesh distortion may appear, and the governing equations always face the problem of specifying the speeds of mesh nodes. Löhner et al. [21] found that a smooth distribution of mesh speeds, obtained by specifying the speeds of nodes with a certain distance from moving surfaces equal to zero, could minimize the mesh distortion.

In Eulerian methods, the computational mesh is treated as a frozen reference frame through which flow particles can move. In this approach, the coordinate system is stationary in a fixed reference frame or moves in a prescribed manner to account for the continuously changing shape of the computational domain. The main advantages of Eulerian methods over Lagrangian methods are that it can handle the flows with a large distortion, and the formulae are easily derived. Because of its merits, this method has

been widely used in moving boundary problems. Unfortunately, this method has three obvious flaws [22]: 1) fluid particles are free to cross the integral volume surfaces; 2) the numerical diffusion is solely associated with discretizing errors from the computation of convective terms; 3) a contact/shear layer may be smeared with the elapse of time and distance.

In spite of its inherent shortcomings, the Eulerian approach has obtained fidelity among CFD society [23-25]. The research present in this dissertation is based on this method.

To perform a computation, a body-fitted mesh has to be regenerated or deformed to account for the motion of bodies. The former option is obviously expensive. Several successful approaches to obtain deformed meshes have been reported. Goswam [26] obtained a valid dynamic mesh with superior quality by combining the smoothing and restructuring operation for bodies undergoing a huge translation and rotation. The Laplacian-type smoothing operation was used to move mesh nodes to new locations, and a restructuring operation was resorted to correct those cells violating the Delaunay property. Batina et al. [27] found another way to deal with a large-scale motion, wherein a coarse mesh was generated, either locally or globally, followed by the use of an adaptive H-refinement technology. The tension-spring analogy and potential flow analogy [2, 27-29] were also reported to obtain good results. In the spring analogy method, the distribution of spring stiffness is crucial. For such simple configurations as isolated pitching airfoils or wings, the spring analogy method has been proven to work well [27,29]. In the potential analogy method, the nodes and the mesh cells are

considered to represent an incompressible flow field, and the mesh nodes are forced around a body as the body moves through a field of mesh nodes.

Takayama et al. [6] used a dynamic adaptive method to adjust the existed unstructured mesh in simulating moving boundary problems. The flexible data structure of an unstructured mesh naturally supported his concept of dynamic adaptation, in which new nodes were inserted and mesh cells were subdivided around such singularities as shock waves and contact surfaces. Then, the earlier inserted nodes were deleted as singularities moved away. However, the repeated interpolations between successive meshes may also contribute to additional numerical errors.

2.3 Multigrid Methods on Unstructured Meshes

The multigrid method is a well-known approach to accelerate the convergence rate of a static solution. While its application on structured meshes is straightforward, additional issues, such as how to communicate the values of variables among meshes and the construction of additional coarse meshes from fine meshes, are involved when it is applied on unstructured meshes.

A Full-Approximation Scheme (FAS), in which it is assumed that the high frequency errors on the fine mesh have been annihilated, is widely employed to obtain the corrections on coarser meshes. Connel et al. [30] employed an adaptive multigrid scheme to solve Euler equations, where fine meshes were obtained from a coarse mesh by dividing cells. In this approach, a sequence of fully nested meshes is yielded and a particularly simple interpolating operator among meshes is promised. Easy automation is

another advantage of this method. The lack of flexibility in handling problems on a specified fine mesh is its first flaw. The second flaw is that this strategy places conflicting demands on the construction of coarse meshes.

Baysal and Luo [31] generated two levels of completely independent coarse meshes by the advancing-front method and employed linear interpolations to perform the restriction and the prolongation operation. This approach is more flexible than the nested one, and the only requirement is that all levels of meshes conform to the same computational domain. Therefore, the meshes can be optimized for the accuracy and the rate of convergence, respectively.

Guillard [32] proposed to automatically generate coarse meshes from the fine mesh by moving the selected fine mesh nodes and retriangulating the remaining mesh nodes. This technique results in a vertex-nested mesh in which the grid nodes of coarse meshes are a subset of the fine mesh nodes. The main merit of this approach is that the construction of the inter-grid transfer operators is simple, because of the vertex-nested property. How to specify the optimal coarsening strategy to guarantee that the critical boundary points are not to be moved is a challenge. For certain problems, a simple uniform coarsening strategy is far from optimal.

Venkatakrishnan et al. [33] used the agglomeration method to automatically construct coarse meshes from a fine mesh, by fusing the neighbored control volumes on the fine mesh and resulting in a smaller set of larger polygonal control volumes. The coarsening procedure can be repeated, and a series of agglomerated coarse mesh result. The fact that the fine mesh and coarse meshes always satisfy the same boundary is the

main merit of this approach. However, how to interpolate the corrections on large polygons accurately back to cells on the fine mesh is a challenge.

2.4 Time Integration

The time-accurate computations generally require more efforts than their steady-state counterparts. Current time-accurate methods include iterative and non-iterative procedures.

Explicit methods, in which the spatial derivatives are evaluated by using known values, are easy to implement. The implementation of boundary conditions is also easier. The multistage Runge-Kutta scheme is the most successful one of this type of schemes. Due to the above features, this method is widely used to simulate unsteady flows [1,34-36]. Its major disadvantage lies in the numerical stability denoted by the CFL limit. For problems with a large time domain, the restriction imposed by the numerical stability becomes particularly severe, and explicit schemes will lead to such large running CPU times that they almost can not be accepted as a practical approach.

The implicit approximate-factorization methods were used to compute unsteady flows by solving Euler and Navier Stakes equations [37]. Although these methods relieve the stringent restriction on the allowable time step, factorization errors are introduced, which, in practice, limit the maximum allowable time step.

The two above methods are typical representatives of non-iterative methods. Several iterative methods, in which the errors associated with factorization or

linearization are eliminated by inner iterations, along with a discussion of their characteristics, will be explained next.

Normally, there are two kinds of iterative implicit schemes: Approximate-Newton and dual-time stepping scheme. Rumsey et al. [37] employed the former to compute unsteady viscous flows around an airfoil on a structured mesh. Frink [39] solved a set of linear equations from the two-point backward-differencing scheme using a subiterative procedure, wherein tetrahedral cells were grouped into "colors" so that no two cells shared a common face. He used this "red-white" iterative method to simulate laminar flows around the OM6 wing. Venkatakrishnan et al. [40] used the Generalized Minimum Residual technique (GMRES) to solve the approximate systems by Newton linearization on unstructured meshes. Three different preconditioners, namely incomplete LU factorization (ILU), block diagonal factorization and symmetric successive over relaxation (SSOR), were investigated. However, the Jacobian matrix after linearization has a block structure, and each block consists of a 5 by 5 sub-matrix. Therefore, the implicit schemes, based on the linearization method, usually incur a memory requirement at least three to four times larger, as compared to their explicit counterparts.

In the dual-time stepping scheme, an additional pseudo-time derivative is added to the derivative in the physical time domain. Then, subiterations are resorted to rule out the errors at each corresponding instant. Jameson [41] used the multigrid method as a driver to accelerate the convergence rate of subiterations. The main merits of this scheme are that it incurs no storage overheads, which plague traditional implicit schemes, based on the linearization, and the size of time steps is solely determined by the concerns of accuracy and is completely independent of meshes. Thus, the dual-time stepping scheme

is an attractive approach to solve unsteady problems with a large time domain on unstructured meshes. Recently, it has been successfully employed to simulate unsteady flows [31,42-44].

2.5 Computations of Hypersonic Flows

Since the early 1980's, computational fluid dynamics (CFD) has been used to simulate hypersonic flows. CFD has become an attractive analysis and design tool for hypersonic flows. One class of problems which takes advantage of CFD is in the airbreathing propulsion concepts for a hypersonic flight. NASA has sponsored several hypersonic programs, namely X-33, X-34, X-38 and Hyper-X [4, 45-47]. In hypersonic regions, CFD plays a much more important role than in transonic and subsonic flows, partly because it may be the only available design and analysis tool. In general, with the existence of strong shocks, hypersonic flows present a larger challenge to the development of computational capabilities, as compared to transonic/supersonic flows.

Bibb et al. [4] computed inviscid hypersonic flows around the X-33 model in Mach 5.25 and Mach 10.6 flows on an unstructured mesh. An efficient edge-based finite volume formulation was derived to discretize the computational domain. A flux splitting scheme, capable of representing a constant enthalpy, was used to calculate the convective flux. The results were compared to experimental data and those on a structured mesh by LAURA code. Due to its accuracy and flexibility, this methodology has successfully been applied in the preliminary design stage of X-33 and X-34 programs.

Campbell et al. [45] reported the computations of hypersonic flows around the X-38 Model, using commercial codes: IEC3D and GASP. Multi-block structured meshes and the van Leer flux-splitting scheme are utilized in these codes. Results in viscous and inviscid longitudinal flows at Mach 6 flight were compared to wind-tunnel data. Frendi [46] reported three-dimensional inviscid computations in the support of the Hyper-X vehicle aerodynamic database, using GASP code (version 3.16). The effects of Mach number, angle of attack and sideslip on the Hyper-X stability and control were investigated extensively.

This chapter provides an overview of state-of-start methods in the field of dynamic CFD. The next chapter will present the details of methodology used in this dissertation.

CHAPTER 3

METHODOLOGY

3.1 Governing Equations for Moving Boundary Problems

The time-dependent Euler equations for an ideal gas express the conservation of mass, momentum and energy in a compressible inviscid flow. Also, it is assumed that flows are non-conducting adiabatic in the absence of external forces.

Three-dimensional time-dependent Euler equations with moving boundaries are written below in the integral form for a bounded domain with a boundary $\partial\Omega$:

$$\frac{\partial}{\partial t}\iiint_{\Omega}\vec{Q}dV + \iint_{\partial\Omega}F(\vec{Q})\bullet\hat{n}dS = 0 \tag{3.1}$$

where

$$\vec{Q} = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_0 \end{Bmatrix} \tag{2.2}$$

$$\vec{F}(\vec{Q})\bullet\hat{n} = (\vec{V}\bullet\hat{n})\begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_0 + p \end{Bmatrix} + p\begin{Bmatrix} 0 \\ \hat{n}_x \\ \hat{n}_y \\ \hat{n}_z \\ a_t \end{Bmatrix} \tag{3.3}$$

The above equations are nondimensionalized with respect to the reference density ρ_∞ and sound speed a_∞. The velocity vector, $\vec{V}$, is written in the form including the grid motion:

$$\vec{V} = \{(u - x_t), (v - y_t), (w - z_t)\} \tag{3.4}$$

The contravariant grid velocity, a_t, is calculated by averaging the speeds of grid nodes, and is defined as:

$$a_t = x_t \hat{n}_x + y_t \hat{n}_y + z_t \hat{n}_z \tag{3.5}$$

where $x_t, y_t,$ and z_t are the grid node's speeds in x, y, and z directions, respectively.

For an ideal gas, the pressure and total enthalpy can be defined as:

$$p = (\gamma - 1)[e_0 - \tfrac{1}{2}\rho(u^2 + v^2 + w^2)] \tag{3.6}$$

$$h_0 = \tfrac{\gamma}{\gamma - 1}\tfrac{p}{\rho} + \tfrac{1}{2}(u^2 + v^2 + w^2) \tag{3.7}$$

3.2 Methodology with Upwind Finite Volume Scheme

3.2.1 Spatial discretization

The finite volume method makes it possible to ensure the conservation laws directly in the physical space. The governing equation (3.1) can be written in a semi-discrete form:

$$\frac{\partial}{\partial t}(M\vec{Q}V)+\sum_{k}(\vec{F}\bullet n^{k})A_{k}=0 \tag{3.8}$$

Here, V is the volume of an integral domain, and the sum is taken over the k^{th} face with unit vector n^{k} and area A^{k}.

The mass matrix, M, comes from the assumption that the update of conservative variables, induced by net fluxes across the surrounding surfaces of a control volume, should be averaged. For simplicity, the mass matrix is lumped.

In cell-centered schemes, mass lumping does not compromise the accuracy, since the variable values at centroids match the averaged values up to second-order accuracy. However, in cell-vertex schemes, the mass lumping induces locally first-order spatial errors because the centers of control volumes are not the vertices in a non-uniform mesh [44]. On unstructured meshes, a cell-centered scheme may incur two times CPUs and memory requirement in comparison to a cell-vertex scheme [49]. However, the quality of a solution in a cell-centered scheme is superior to that in a cell-vertex scheme [50], partly due to the fact that the control volumes in a cell-centered scheme are smaller. For simple time-dependent problems, such as pitching airfoils or wings, the cell-centered method is recommended to avoid the errors related to mass lumping and to obtain better solutions at acceptable costs in terms of CPU time and running memory. However, for complex time-dependent problems, such as the analysis of 3-D aerodynamic interference among multi bodies in relative motions, the cell-vertex method is recommended to discretize the computational domain to save computer resources.

Upwind schemes introduce the physical properties of flow equations into spatial discretization to aim at preventing the occurrence of unwanted oscillations near nonlinear regions. In upwind schemes, the artificial numerical dissipation is automatically added. The flux vector-splitting scheme is one of two ways to implement the upwind scheme, and it is used in the present study. The Godunov-type schemes are the other class of upwind schemes.

In a flux vector-splitting scheme, the first level of physical information is introduced by splitting the interface fluxes on the sign of eigenvalues. However, fluxes after splitting are not continuously differentiable, since they have different slopes at sonic velocity. Van Leer imposed four conditions on the splitting terms to overcome this shortcoming. With those imposed conditions, the fluxes in supersonic flows can be evaluated as:

$$\hat{F}^{+} = (\vec{F}(\vec{Q}) \bullet \hat{n})^{+}, \quad \hat{F}^{-} = (\vec{F}(\vec{Q}) \bullet \hat{n})^{-} = 0 \quad \text{if} \quad M_n \geq 1 \tag{3.9}$$

$$\hat{F}^{-} = (\vec{F}(\vec{Q}) \bullet \hat{n})^{-}, \quad \hat{F}^{+} = (\vec{F}(\vec{Q}) \bullet \hat{n})^{+} = 0 \quad \text{if} \quad M_n \leq -1 \tag{3.10}$$

In subsonic flows, the fluxes are split into the form given as:

$$\vec{F}(\vec{Q}) = \hat{F}^{+}(Q^{-}) + \hat{F}^{-}(Q^{+}) \tag{3.11}$$

where

$$\hat{F}^{\pm} = \begin{Bmatrix} f^{\pm}_{mass} \\ f^{\pm}_{mass}[u + \hat{n}_x(-U_n \pm 2a)/\gamma] \\ f^{\pm}_{mass}[v + \hat{n}_y(-U_n \pm 2a)/\gamma] \\ f^{\pm}_{mass}[w + \hat{n}_z(-U_n \pm 2a)/\gamma] \\ f^{\pm}_{energy} \end{Bmatrix} \tag{3.12}$$

with

$$f^{\pm}_{mass} = \pm\frac{a\rho}{4}(M_n \pm 1)^2 \tag{3.13}$$

$$f^{\pm}_{energy} = f^{\pm}_{mass}\left[\frac{(1-\gamma)U_n^2 \pm 2(\gamma-1)U_n a + 2a^2}{(\gamma^2-1)} \right.$$

$$\left. \frac{a_t(-U_n \pm 2a)}{\gamma}\right] \tag{3.14}$$

Here, a_t is the grid speed; U_n is the contravariant velocity and is defined as:

$$U_n = \vec{V} \bullet \hat{n} = (u - x_t)\hat{n}_x + (v - y_t)\hat{n}_y + (w - z_t)\hat{n}_z \tag{3.15}$$

The split fluxes in the equations (3.11)-(3.15) are used for moving boundary problems by modifying the standard van Leer scheme to include grid speeds in the contravariant form and adjusting the convective velocity.

When the above method is extended directly to hypersonic flows, it becomes less robust and needs more dissipation to maintain numerical stability due to the strength of shocks. In flows where strong normal shocks appeared, the "carbuncle phenomenon" was reported, and the Hänel flux vector splitting scheme was suggested [4]. The main attractive feature of this scheme is that the converged solution always has a constant

enthalpy whenever the enthalpy in an incoming flow is a constant. The scheme in two dimensions can be written as:

$$F_n(U) = F_n^+(U^-) + F_n^-(U^+) \tag{3.16}$$

where

$$F_n^{\pm}(U) = \rho u_n^{\pm} \begin{pmatrix} 1 \\ u \\ v \\ H \end{pmatrix} + p^{\pm} \begin{pmatrix} 0 \\ \widehat{S}_1^e \\ \widehat{S}_2^e \\ 0 \end{pmatrix} \tag{3.17}$$

The definition of $u_n^{\pm}$ and $p^{\pm}$ is given by:

$$u_n^{\pm} = \begin{cases} \pm(u_n \pm c)^2 / 4c, & \text{if } |u_n| \le c \\ \frac{1}{2}(u_n \pm |u_n|), & \text{otherwise} \end{cases} \tag{3.18}$$

and

$$p^{\pm} = \begin{cases} \frac{1}{4} p(u_n / c \pm 1)^2 (2 \mp u_n / c), & \text{if } |u_n| \le c \\ p(u_n \pm |u_n|) / 2u_n, & \text{otherwise} \end{cases} \tag{3.19}$$

where $u_n = u_k \widehat{S}_k^e$; the speed of sound, c, is given by $c^2 = \frac{p\gamma}{\rho}$.

In an upwind scheme, discontinuous states are assumed to exist on both sides of a mesh interface. In a first-order cell-centered scheme, the values of conservative variables are assumed to be constants within a control volume. Although a first-order scheme provides sufficient dissipation to make the computation stable and has a capability to capture shocks just over two or three cells, it obtains a poor solution in smooth regions

and requires unnecessarily fine meshes. Normally, a reconstruction method, based on the Taylor series expansion theory, is resorted for a higher-order spatial discretization. At an interface, having:

$$Q(x,y,z) = Q(X_c, Y_c, Z_c) + \nabla Q_c \bullet \Delta\bar{r} + \vartheta(\Delta r^2) \tag{3.20}$$

where ∇Q_c is the gradient of variables, usually obtained by Green's theorem:

$$\nabla Q_c = \frac{1}{V_\Omega} \oiint_{\partial\Omega} Q\hat{n} dS \tag{3.21}$$

For a cell-centered scheme, taking advantage of the invariant characteristics of a tetrahedron, the second term in equation (3.20) is evaluated as:

$$\nabla Q_c \bullet \Delta\bar{r} = \frac{\partial Q}{\partial r}\Delta r \approx \left[\frac{1/3(Q_{n1}+Q_{n2}+Q_{n3})-Q_{n4}}{4\Delta r}\right]\Delta r \tag{3.22}$$

where Q_{n1}, Q_{n2}, Q_{n3} denote the values of primitive variables at the nodes of the interface; Δr denotes the distance from the tetrahedron center to the interface center; Q_4 represents the values of primitive variables at the fourth node of the tetrahedron.

The above approximation reduces CPU time by approximately two orders. Since the variables are stored at cell centers, nodal quantities in the equation (3.22) are

determined by averaging the values in surrounding cells, taking the reverse of the distance between that node to the centers of surrounding cells as weighting functions,

$$Q_n = \left(\sum_{i=1}^{N} \frac{Q_{c,i}}{r_i}\right) \Big/ \left(\sum_{i=1}^{N} \frac{1}{r_i}\right) \tag{3.23}$$

where N is the number of cells to which the node n belongs; the weighting function, r_i, is defined as:

$$r_i = [(x_{c,i} - x_n)^2 + (y_{c,i} - y_n)^2 + (z_{c,i} - z_n)^2]^{1/2} \tag{3.24}$$

An unexpected advantage of the above procedure is that in transonic flows, shocks can be captured with virtually no oscillation without the use of limiters. This is credited to the above averaging procedure, which promises that Q_n is bounded by values in neighboring cells. Therefore, there is no new extrema introduced. However, it is proven that the accuracy of spatial discretization is degraded slightly from second-order to about 1.85 order [39].

Unfortunately, the dissipation added in the above averaging process is not sufficient to guarantee stable computations in a hypersonic flow with a network of strong shocks. Therefore, Van Albada limiter [18] is resorted. For two-dimensional problems, considering an interface of two cells with the center and the other vertex denoted as i, j, $i+1$ and $j+1$, respectively, Q^- and Q^+ components at this interface are given as:

$$Q^{-} = Q_i + \frac{s}{6}[(1-\kappa s)\Delta_{-} + (1+\kappa s)\Delta_{+}]_i \tag{3.24}$$

$$Q^{+} = Q_{i+1} - \frac{s}{6}[(1+\kappa s)\Delta_{-} + (1-\kappa s)\Delta_{+}]_{i+1} \tag{3.25}$$

with the following definitions:

$$\Delta_{-i} = Q_i - Q_j,\ \Delta_{+i} = Q_{i+1} - Q_i,\ \Delta_{-i+1} = Q_{i+1} - Q_i,\ \Delta_{+i+1} = Q_j - Q_{i+1} \tag{3.26}$$

A continuously differential flux limiter, known as the van Albada limiter, is obtained by defining:

$$s = \frac{2\Delta_{-}\Delta_{+} + \varepsilon}{\Delta_{-}^2 + \Delta_{+}^2 + \varepsilon} \tag{3.27}$$

where the parameter, ε , is a very small number, which is defined to be zero in smooth regions, to avoid singularity.

Despite the use of the van Albada limiter, an occasional instability was encountered, and it had to be treated in the following approaches:

1) $p_{cell} = \max(p_{cell}, p_{tol})$, where P_{tol} is the user prescribed minimal value of pressure.
2) For cells once P_{tol} was used, the spatial discretization was dropped to first-order in subsequent iterations.

The practice has shown that those measures offered enough local numerical diffusion to stabilize and ensure the convergence of solutions. With the directive application of the above spatial discretization, numerical difficulties occur in the blunt-base region, where a wake-type flow would be developed in a real viscous flow.

Therefore, the inviscid assumption yields a singularity, and the computation attempts to negotiate an expansion corner, as shown in figure 3.1.

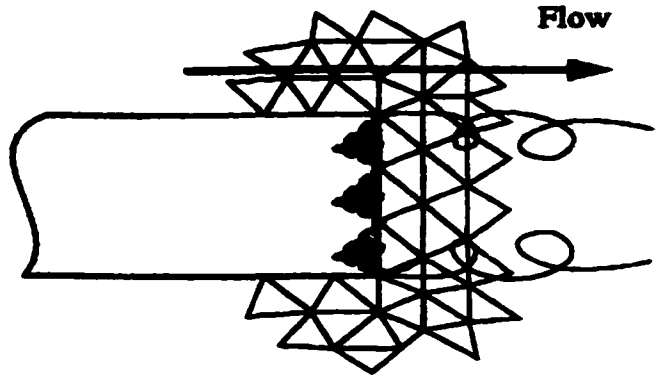


Figure 3.1 Numerical flow in blunt-based region.

A special boundary condition has been implemented to mimic the relieving effects of a blunt-based wake in the surrounding inviscid flows. The reconstructed quantity (ρ, u, v, p) from the nodes of blunt-based boundary cells are averaged and assigned to the boundary faces and ghost cells, which introduces a transpiration boundary condition to the faces and simulates the existence of a wake.

3.2.2 Time integration

There are two different approaches to advance a solution in time domain. The explicit scheme, used in previous dynamic unstructured technique (DUT), is explained briefly. Then, the details of a novel implicit scheme are described.

3.2.2.1 Explicit scheme

The semi-discrete form of governing equations after spatial discretization is written as:

$$V_i \frac{\partial Q_i}{\partial t} + R_i = 0, \quad i = 1,2,3 \cdots N \tag{3.28}$$

$$R_i = \sum_{j=1}^{3} F_{i,j} \, \Delta S_{ij} \tag{3.29}$$

The residual, R_i, is accrued by summing over the fluxes on the four faces of a tetrahedral cell. M-stage Rung-Kutta scheme is utilized to advance in time domain:

$$\begin{aligned} Q_i^{(0)} &= Q_i^n \\ Q_i^{(1)} &= Q_i^{(0)} - \alpha_1 \frac{\Delta t}{V_i} R_i^{(0)} \\ Q_i^{(m-1)} &= Q_i^{(0)} - \alpha_{m-1} \frac{\Delta t}{V_i} R_i^{(m-2)} \\ Q_i^{(m)} &= Q_i^{(0)} - \alpha_m \frac{\Delta t}{V_i} R_i^{(m-1)} \\ Q_i^{(n+1)} &= Q_i^{(m)} \end{aligned} \tag{3.30}$$

where n denotes the time level; m denotes the stage number in the Runge-Kutta scheme; the weighting factor, α_i, is defined as:

$$\alpha_i = \frac{1}{m-i+1}, \qquad i = 1,2,...m \tag{3.31}$$

For linear problems, the above equation will give m^{th} order accuracy in time domain.

3.2.2.2 Implicit scheme

In this dissertation, an implicit scheme has been implemented to replace the explicit scheme. The details of the implicit scheme are explained below.

Taking three-point backward difference for the derivative in the equation (3.28) yields:

$$\frac{3}{2\Delta t}V^{n+1}U^{n+1} - \frac{2}{\Delta t}V^{n}U^{n} + \frac{1}{2\Delta t}V^{n-1}W^{n-1} + R(U^{n+1}) = 0 \tag{3.32}$$

The above equation is A-stable when applied for a linear differential equation. It can be treated as an equation in a static flow at a corresponding instant. Hence, a derivative in pseudo time domain is added, and the equation (3.32) is rewritten as:

$$\frac{\partial VW}{\partial t^*} + R^*(W) = 0 \tag{3.33}$$

where W is the approximation of U^{n+1} .

The unsteady residual, $R^*(W)$, and source term, S, are defined as:

$$R^*(W) = \frac{3}{2\Delta t} VW + R(W) - S(V^n U^n, V^{n-1} U^{n-1}) \tag{3.34}$$

$$S(V^n U^n, V^{n-1} U^{n-1}) = \frac{2}{\Delta t} V^n U^n - \frac{1}{2\Delta t} V^{n-1} U^{n-1} \tag{3.35}$$

The low-storage, m-stage Runge-Kutta scheme is used to advance the equation (3.33) in pseudo time domain:

$$\begin{aligned} &Q_0 = W^n \\ &V^{n+1} Q_k = V^{n+1} Q_0 - \alpha_k \Delta t^* R^*(Q_{k-1}), \quad k = 1, \ldots \text{m} \\ &W^{n+1} = Q_m \end{aligned} \tag{3.36}$$

Starting with $W^L = U^n$, and after W^L, $L = 1,2,\ldots$, W finally converges to U^{n+1}. Therefore, there are two different time steps in the equation (3.36): physical time step Δt and pseudo time step Δt^*. The physical time step is used to advance solutions in physical time domain and it comes solely from the concerns of accuracy in physical time domain. The subiterations in pseudo time domain play a role of smoother, and the pseudo time step is determined from the numerical stability analysis.

It was found that the equation (3.36) was unstable for small Δt, because the term $\frac{3}{2\Delta t} VW$ was explicitly included in the unsteady residual, $R^*(W)$. It was observed that the stable region would increase as Δt was decreased by treating this term implicitly:

$$
\begin{aligned}
&Q_0 = W^n \\
&[I + \tfrac{3}{2\Delta t}\alpha_k \Delta t^*] V^{n+1} Q_k = V^{n+1} Q_0 - \alpha_k \Delta t^* [R(Q_{k-1}) - S], k = 1, m \\
&W^{n+1} = Q_m
\end{aligned}
\tag{3.37}
$$

The dual-time-stepping scheme has three obvious advantages. First, it uncouples the dynamic mesh module and the flow solver. In an explicit scheme, they are inherently coupled by using the same time step from the numerical stability analysis. Actually, the allowable time step for the spring analogy method is much larger than that for a solver, especially in the cases using fine meshes. Therefore, CPUs are wasted in the dynamic mesh module. Once these two modules are uncoupled, the dynamic mesh module will be invoked only when necessary by using as large a time step as allowable in the spring analogy method. Second, all the matured accelerating techniques for static problems, such as local time stepping, residual smoothing and multigrid method, can be explored for unsteady problems. Third, it incurs no storage overheads, which typically plague implicit schemes, based on linearization method.

Next, the efficiency of the dual-time stepping scheme with respect to an explicit scheme is explained in a transonic pitching airfoil problem. With the minimal size of mesh cells equal to Δx, the allowable time step with CFL number around one in an explicit scheme can be estimated as:

$$\Delta t_e \approx (\Delta x / U_\infty) \tag{3.38}$$

For an A-stable implicit scheme to resolve the unsteady flows with a reduced

frequency equal to k, assuming that N steps per period are enough for a second-order scheme yields:

$$T = 2\pi / \omega = c\pi / kU_\infty \tag{3.39}$$

$$\Delta t_i = (\pi / N)(c / kU_\infty) \tag{3.40}$$

If CPUs spent to advance one step in the implicit scheme are M times those in the explicit scheme, then the ratio of CPUs in implicit and explicit schemes to advance the same size in time domain is given as:

$$\frac{CPU_{emplicit}}{CPU_{implicit}} = \frac{\Delta t_i}{M\Delta t_e} \approx \frac{\pi C}{MNk\Delta x} \tag{3.41}$$

Here, Δt_i and Δt_e represent the time step in the implicit scheme and the explicit scheme, respectively.

From the equation (3.41), provided that the parameter, M, is not too large, a great benefit in the implicit scheme is observed when the reduced frequency or the minimal size of a mesh is small. M represents the efficiency of subiterations in pseudo time domain, and all the matured accelerating techniques for static problems are powerful tools to decrease it.

3.2.3 Geometric conservation law

For moving boundary problems, meshes need to be adjusted for the relative motions of bodies. Since the Eulerian method allows flows to pass through control

volumes, a uniform steady-state flow in a moving mesh requires the geometric conservation law to avoid the violation of the conservation laws of mass, momentum, and energy. The geometric conservation law is in the same integral form as the mass conservation law and can be explained in the following way.

For a time interval $t_2 - t_1$, the Euler equations over a time-dependent control volume $\Omega(t)$ can be written below:

$$\int_{\Omega(t_2)} QdV - \int_{\Omega(t_1)} QdV + \int_{t_1}^{t2} \oint_{\partial\Omega(t)} \vec{F}(t) \bullet \hat{n} dSdt = 0 \qquad (3.42)$$

where V is the time-dependent volume of an integral subdomain, and $\hat{n}$ is the unit vector on the boundary surface of a control volume.

For time-dependent problems, the convective flux vector, $\vec{F}$, can be split into static and dynamic parts:

$$\vec{F}(t) = (\vec{u} - \vec{x}_t)Q = \vec{F}_{static} - \vec{x}_t Q \qquad (3.43)$$

where $\vec{u}$ and $\vec{x}_t$ are the velocity vectors of fluid particles and the faces of a control volume, respectively.

The geometric conservation law can be obtained by combining the two above equations and considering a uniform free-stream flow in which the vector $\vec{Q}$ is a constant.

$$[V(t_2) - V(t_1)]Q_\infty + \vec{F}_{static} \bullet \int_{t_1}^{t2} \oint_{\partial\Omega(t)} \vec{n} dSdt - Q_\infty \int_{t_1}^{t2} \oint_{\partial\Omega(t)} \vec{n} \bullet \vec{v} dSdt = 0 \qquad (3.44)$$

where the second term is equal to zero because of the geometric property.

For a uniform free-stream flow, the following time-dependent equation should be satisfied [53]:

$$[V(t_2)-V(t_1)]=\int_{t_1}^{t_2}\oint_{\partial\Omega(t)}\vec{n}\bullet\vec{v}dSdt \tag{3.45}$$

The equation (3.45) should accompany the conservation laws of mass, momentum, and energy in moving boundary problems in order to avoid errors induced by a moving mesh. Therefore, the volume V in the equation (3.30) and (3.36) should be updated by the equation (3.44) in each time step.

3.2.4 Acceleration techniques

For static problems, the multistage Runge-Kutta scheme is exploited to implement the integration in pseudo time domain. However, it experiences severe stability problems when the CFL number is close to or larger than $2\sqrt{2}$, which is a very serious restriction on fine meshes. The well-known accelerating techniques, such as local time stepping, residual smoothing, and the multigrid method, are usually resorted to accelerate the convergence rate of static solutions.

3.2.4.1 Local time stepping

In pseudo time domain, each cell is advanced at its allowable time step, which is obtained from the local stability analysis. Taking advantage of local time stepping, the

error waves can be transported through the computational domain at a higher speed. However, for time-accurate problems, the fixed global time step is the only choice. For any cell k with projected areas A_k^x, A_k^y, A_k^z in x, y, and z direction, respectively, the local time step can be given as:

$$\Delta t_k = \frac{(CFL)V_k}{(|u_k|+a_k)A_k^x + (|v_k|+a_k)A_k^y + (|w_k|+a_k)A_k^z} \tag{3.46}$$

where V_k is the volume of cell and a_k is the local speed of sound.

3.2.4.2 Implicit residual smoothing

The maximum CFL number for an explicit Runge-Kutta scheme in uniform meshes can be increased up to 4 by using the implicit average of cell residuals in neighboring cells [54]. In fact, residuals are filtered through a smoothing operator, and a set of equations is solved by the Jacobian method. For cell i, having:

$$R_i^{new} = (R_i^{old} + \varepsilon \sum_{j=1}^{\kappa(i)} R_j^{old}) \Big/ (1 + \varepsilon \sum_{j=1}^{\kappa(i)} 1) \tag{3.47}$$

where $\kappa(i)$ is the local connectivity of cell i. By practice, two or three iterations are enough.

3.2.4.3 Full-Approximation-Storage scheme

In this approach, an entire nonlinear problem is transferred from a fine mesh to a set of coarser meshes. After using a three-point second-order backward difference

formulation and denoting the vector of conserved variables by Q, the integral form of the governing equations can be written as:

$$L[(Q)^{n+1}] = A \tag{3.48}$$

Here, L and A denote the operator and source term, respectively, which are defined as:

$$L[(Q)^{(n+1)}] = \frac{3}{2\Delta t}(Q)^{n+1} + R(Q^{n+1}) \tag{3.49}$$

$$A = \frac{2}{\Delta t}(VQ)^{n} - \frac{1}{2\Delta t}(VQ)^{n-1} \tag{3.50}$$

where the residual, R, is constructed by summing over fluxes through the surfaces of a control volume:

$$R_i = \sum_{j=k(i)} F_{ij} \ \Delta S_{ij} \tag{3.51}$$

Defining the errors in the above nonlinear system on the fine mesh as v:

$$L^h(Q^h + v^h) - L^h(Q^h) = A^h - L^h(Q^h) = R^h \tag{3.52}$$

then, on the coarse mesh, having:

$$L^{2h}(I_h^{2h}Q^h + v^{2h}) - L^{2h}(I_h^{2h}Q^h) = I_h^{2h}(A^h - L^hQ^h) \tag{3.53}$$

By rearranging terms and defining a forcing function on coarse meshes as:

$$f^{2h} = I_h^{2h}(A^h - L^h Q^h) + L^{2h}(I_h^{2h} Q^h) \tag{3.54}$$

On the coarse mesh, having:

$$L^{2h}(Q^{2h}) = f^{2h} \tag{3.55}$$

To ensure that the corrections computed on coarser meshes lead to physically correct solutions, the source term, A, on coarse meshes must be defined as the equation (3.54), so that the evolution on coarse meshes is driven by residuals on the finest mesh.

The computations of residuals R^h and R^{2h} must not necessarily be carried out in the same way. The accuracy of solutions is mainly determined by the computations of residual R^h on the fine mesh, and the efficiency of solutions is highly dependent on the convergence of solutions on coarser meshes. Therefore, it is allowable to choose the computing method for R^{2h} solely to improve the convergent rate. A first-order spatial discretization is an ideal choice for R^{2h}, since it has an excellent damping property.

Once Q^{2h} is obtained on a coarser mesh, the solution, Q^h, on a finer mesh is updated:

$$(Q^h)_{new} = (Q^h)_{old} + I_{2h}^h[Q^{2h} - I_h^{2h}(Q^h)_{old}] \tag{3.56}$$

The accuracy of the restriction and the prolongation operators, I_h^{2h} and I_{2h}^h, must be sufficient to avoid inducing excessive errors into a solution, which would in turn have detrimental effects on the efficiency of the solution. A fundamental rule for the accuracy of these inter-grid transfer operators is given by [51]:

$$m_r + m_p > m \tag{3.57}$$

where m_r and m_p represent the highest degree of polynomials plus one on the construction of the restriction and the prolongation operators, respectively, and m denotes the order of partial differential equations being solved.

The equation (3.57) is seldom violated for convection-diffusion equations if both the restriction operator and the prolongation operator utilize bilinear interpolations. Although the process described below is for a restriction operator, the construction for a prolongation operator is in the same way.

In order to transfer data from a finer mesh to a coarser mesh, a bilinear distribution of unknowns is assumed within a control volume. For simplicity, the equations are written in a two dimensional problem. For any cell on a fine mesh, the following linear equation for unknowns can be written as:

$$Ax + By + C = q \tag{3.58}$$

After knowing the values of unknowns at three vertices, the coefficients, A, B, and C, are determined by assembling a system of equations using the values at these three vertices as follows:

$$\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{3.59}$$

After solving the above equations, the following expressions are obtained:

$$A = \frac{q_1(y_2-y_3)+q_2(y_3-y_1)+q_3(y_1-y_2)}{x_1(y_2-y_3)+x_2(y_3-y_1)+x_3(y_1-y_3)} \tag{3.60}$$

$$B = \frac{q_1(x_3-x_2)+q_2(x_1-x_3)+q_3(x_2-x_1)}{x_1(y_2-y_3)+x_2(y_3-y_1)+x_3(y_1-y_3)} \tag{3.61}$$

$$C = \frac{q_1(x_2y_3-x_3y_2)+q_2(x_3y_1-x_1y_3)+q_3(x_1y_2-x_2y_1)}{x_1(y_2-y_3)+x_2(y_3-y_1)+x_3(y_1-y_3)} \tag{3.62}$$

For each node n in a coarser mesh, once the enclosing cell in a finer mesh is known, it is apparent that the unknowns at node n can be written as the summation of geometric weights multiplied by the values at three vertices of the enclosing finer cell:

$$Q_n = W_1Q_1 + W_2Q_2 + W_3Q_3 \tag{3.63}$$

Given the coordinates of grid nodes n, x_n, y_n, the weighting functions W_1, W_2, W_3 are calculated as:

$$W_1 = \frac{x_n(y_2-y_3)+y_n(x_3-x_2)+(x_2y_3-x_3y_2)}{x_1(y_2-y_3)+x_2(y_3-y_1)+x_3(y_1-y_2)} \tag{3.64}$$

$$W_2 = \frac{x_n(y_3-y_1)+y_n(x_1-x_3)+(x_3y_1-x_1y_3)}{x_1(y_2-y_3)+x_2(y_3-y_1)+x_3(y_1-y_2)} \tag{3.65}$$

$$W_3 = \frac{x_n(y_1-y_2)+y_n(x_2-x_1)+(x_1y_2-x_2y_1)}{x_1(y_2-y_3)+x_2(y_3-y_1)+x_3(y_1-y_2)} \tag{3.66}$$

It is easy to verify that the summation of the above weights is equal to unity. Through the above approach, the values of variables at grid nodes or centers of cells in different meshes can be interpolated linearly.

However, if the restriction operator for transferring residuals from a finer mesh to a coarser mesh in the equation (3.54) is constructed by the above approach, an extreme case may appear, as shown in figure 3.2. It is clear that a nonzero residual at a fine grid node P is not utilized on the coarse mesh because none of the fine-grid cells sharing node P enclose any coarse mesh nodes (A, B and C).

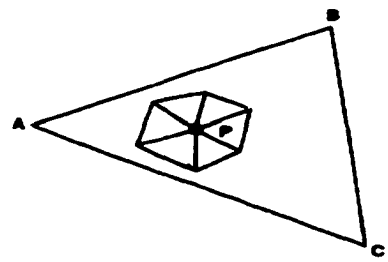


Figure 3.2 Example of a fine-grid node P.

Since residuals are directly related to the averaged rate of change of the conserved variables within a control volume, their restriction operators should be constructed to promise the conservative interpolation. Otherwise, another problem is solved on coarser meshes. In the present dissertation, the distribution method is resorted to construct the restriction operators for residuals.

For each cell center on a finer mesh, its enclosing cell on a coarser mesh is determined, and residuals at the cell center are distributed to the vertices of the surrounding cell. The weights used in the distribution are determined by the equation

(3.64)-(3.66), the same as those in constructing the prolongation operators. After distributing the residuals at each cell center on a finer mesh to coarse mesh nodes, all the contributions from the fine mesh are summed at each coarse mesh node. By the above approach, the residuals on the fine mesh are transferred to the coarse mesh conservatively, since the weights at any given cell center on a finer mesh add up to unity.

3.2.5 Initial and boundary conditions

The approach to impose boundary conditions plays an important role in setting up a well-posed problem for hyperbolic propagation-dominated systems. On the other hand, initial conditions are required in any time-marching numerical computations. In steady flows, a uniform flow field is chosen as the initial condition, while the converged static solutions are specified as initial conditions in unsteady flows.

In steady flows, explicit boundary conditions are used to obtain the velocity and the speed of sound at far-field surfaces with the aids of two locally one-dimensional Riemann invariants, which are given as:

$$R^{\pm} = U \pm \frac{2a}{\gamma - 1} \tag{3.67}$$

Based on the properties of hyperbolic systems, these two invariants are taken as constants along their characteristic lines, so the velocity and the speed of sound in the normal direction of a boundary surface can be acquired. In supersonic flows, R^- and R^+ are specified from the free-stream conditions and the interior of computational domain, respectively. Hence, they can be evaluated as:

$$U_{b,n} = \frac{1}{2}(R^+ + R^-) \tag{3.68}$$

$$a_b = \frac{\gamma - 1}{4}(R^+ + R^-) \tag{3.69}$$

The density on the far-field boundary is calculated from the relationship between the entropy and the speed of sound, while the entropy comes from outside or inside computational domain, depending on the signs of eigenvalues:

$$\rho_b = \left(\frac{a_b}{S\gamma_b}\right)^{\frac{1}{\gamma - 1}} \tag{3.70}$$

The pressure on the far-field surfaces can be obtained from the state equation, once ρ and a are known. On the solid walls, the normal velocity is specified as zero, since flows are not allowed to penetrate walls. The velocity and pressure are extrapolated from the interior of computational domain by a zeroth-order accurate method:

$$\begin{aligned} u_{wall} &= u_{interior} - n_x U \\ v_{wall} &= v_{interior} - n_y U \\ w_{wall} &= w_{interior} - n_z U \end{aligned} \tag{3.71}$$

and

$$\begin{aligned} \rho_w &= \rho_{interior} \\ p_w &= p_{interior} \end{aligned} \tag{3.72}$$

where the subscript indicates a value from computational domain.

The contravariant velocity, U, is defined as:

$$U = u_{\text{interior}}\hat{n}_x + v_{\text{interior}}\hat{n}_y + w_{\text{interior}}\hat{n}_z \tag{3.73}$$

For unsteady problems, the above equations need some modifications to include the speeds of grid nodes. The pressure gradient exists if a body moves with acceleration; hence, pressure can not be obtained by the zeroth-order extrapolation. Therefore, the momentum equation on the moving walls is written as:

$$\frac{\partial p}{\partial n} = -\rho \hat{n} \bullet \vec{a} \tag{3.74}$$

3.2.6 Dynamic mesh

In moving boundary problems, the existing mesh needs to adjust to the motions of boundary surfaces. In the present research, the spring analogy method is used.

3.2.6.1 Spring analogy

The spring analogy method is utilized to move the interior nodes to new locations as body surfaces move, while no nodes are inserted or removed. In the spring analogy method, each edge in a mesh is assumed as a spring. After the displacements of nodes on moving boundaries are specified, a well-posed equilibrium problem is set up whose resulting linear equations are solved by the Jacobian iterative method. The practice has shown that several iterations are usually sufficient, and the key issue for desirable meshes

is to specify suitable coefficients of spring stiffness. Normally, they are assumed inversely proportional to the distance between two nodes in a power form, thus the spring stiffness coefficient of the edge connecting nodes i, j can be written as:

$$k_{i,j} = 1.0 \Big/ \left[(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2\right]^{p/2} \tag{3.75}$$

The parameter p is used to control the coefficient in different regions within the computational domain. In clustering regions, p should be increased, but it should be decreased in smooth regions. For each node, the fictitious external forces from all of the connecting springs are summed after resolving into their Cartesian components, and the resulting algebraic systems are solved to obtain the displacements of that node in x, y, z directions by a point Jaobian method:

$$\Delta x_i^{n+1} = \frac{\sum_{j=1}^{m(i)} k_{ij} \Delta x_i^n}{\sum_{j=1}^{m(i)} k_{ij}} \tag{3.76}$$

$$\Delta y_i^{n+1} = \frac{\sum_{j=1}^{m(i)} k_{ij} \Delta y_i^n}{\sum_{j=1}^{m(i)} k_{ij}} \tag{3.77}$$

$$\Delta z_i^{n+1} = \frac{\sum_{j=1}^{m(i)} k_{ij} \Delta z_i^n}{\sum_{j=1}^{m(i)} k_{ij}} \tag{3.78}$$

where n represents the time level.

The efficiency of the above dynamic mesh method can be improved by specifying a window, since only a small portion of mesh nodes which are near moving bodies need to be adapted [52]. A window is created by either specifying a normal distance from moving bodies or choosing such a basis shape as a circle, an ellipse or a rectangle in 2-D problems. After the buildup of a window, those nodes within the window are flagged as window points, which will respond to the motions of boundary surfaces.

3.2.6.2 Presmoothing

Since the spring analogy method is nothing but solving linear equations and obtaining new coordinates by keeping the grid connectivity at any costs, there is no way to promise an ideal distribution of grid nodes in a deformed mesh [53]. In some cases, the spring analogy can only reach equilibrium when some of the grid cells become overlapped. The following examples are two efforts to improve the quality of dynamic meshes and weaken the possibility of the violation of Delaunay property.

The presmoothing procedure smoothes the distribution of grid nodes by using a Jacobi method:

$$(I + \varepsilon N_i) x_i^{new} = x_i^{old} + \varepsilon \sum_{j=1}^{N_i} x_j^{new} \tag{3.79}$$

Here, N_i is the degree of node i ; ε is a constant, usually taken as 0.25.

With the aid of presmoothing, the abrupt change of area or volume in adjacent cells throughout grids is avoided.

3.2.6.3 Grid optimization

A corrective approach [53] adjusts the overlapped cells by the spring analogy method and maintains the original smoothness in the rest of a grid as much as possible. The basic idea of this corrective approach is to minimize a cost function, which is defined as the deviation of the Jacobian in an actual mesh from that in an associated "ideal" mesh. For a two-dimensional grid, the cost function, F, is given as:

$$F = \frac{1}{3}\sum_{n=1}^{N}\sum_{c=1}^{C_n}[A_c - A_c^*)/A_c^*]^2 \qquad (3.80)$$

Here, A_c is the actual area of a grid cell, sharing node n; A_c^* is the ideal area for that cell.

Minimizing the cost function yields a set of equations in terms of the coordinates of grid nodes, and the new coordinates of a corrective mesh are obtained by solving these equations.

3.3 Methodology with Central-Difference Finite Element Scheme

3.3.1 Introduction

Even though there have been significant developments in using upwind methods in the past two decades, central schemes are still widely used in the simulations of practical compressible inviscid flows. When the upwind schemes described in the previous section were used in the three-dimensional simulations of hypersonic vehicle staging, severe stability problems were encountered. Although the Hänel flux splitting scheme with the van Albada limiter and transpiration/relaxation boundary conditions worked well in the computations on the longitudinal symmetry plane, it failed in the full three-dimensional simulations. The static computations diverged after several hundred iterations, even when the first-order spatial discretization was used. It was concluded that the above upwind scheme did not offer sufficient damping to promise the solution stable. The reasons may be as follows.

Although the strength of shocks was weaker in 3-D cases than in the 2-D cases, the shape of the shock system was more complex. A series of expansion waves appeared in spanwise direction, which did not appear in the 2-D computations.

Therefore, a central differencing scheme, plus artificial viscosity, was used to replace the upwind scheme by choosing the minimally allowable artificial viscosity to make the solutions stable. The artificial dissipation is constructed in the form of a monotonicty function, and the bulk viscosity is also added to mimic the mass expansion in hypersonic flows.

In order to save computer resources in full three-dimensional computations, the finite element method with an efficient edge-based data structure was also used to replace the finite volume method with a cell-center data structure. By practice, almost one order of run-time memory savings was achieved by the new data structure.

The spring analogy method described in the previous section was also used to account for the moving boundary surfaces, and the multi stage Runge-Kutta explicit scheme was employed to advance the governing equations in time domain.

3.3.2 A finite element formulation

A suitable weak formulation for the governing equation (3.1) is discretized as finding U(x, t) for any suitable weighting function W (x) and all t > 0, which satisfies the following equation:

$$\int_{\Omega} \frac{\partial U}{\partial t} W d\Omega = \sum_{j=1}^{3} F^{j} \frac{\partial W}{\partial x_j} d\Omega - \sum_{j=1}^{3} F^{j} n^{j} W d\Gamma \qquad (3.81)$$

Here, n^j denotes the j^{th} component of the normal vector on the boundary Γ.

With Ω represented by an unstructured assembly of four-noded tetrahedral elements, a piecewise linear approximate solution, U^*, is sought in the following form:

$$U^*(x,t) = \sum_{J} U_J(t) N_J(x) \qquad (3.82)$$

Here, the summation extends over each node J $(1 \le J \le n_p)$ in the control volume;

$U_J(t)$ denotes the values of the approximation U^* at node J at time t; $N_J(x)$ is the standard linear finite element shape function associated with node J.

Using the equation (3.81), a Galerkin approximate solution is constructed as a function of U^* for each N_I $(1 \le I \le n_p)$ and for all $t > 0$:

$$\int_{\Omega} \frac{\partial U^*}{\partial t} N_I d\Omega = \sum_{j=1}^{3} \int_{\Omega} F^j(U^*) \frac{\partial N_I}{\partial x_j} d\Omega - \sum_{j=1}^{3} F^j(U^*) N_I d\Gamma \qquad (3.83)$$

The integrals appearing in the equation (3.83) can be evaluated by summing the individual element contributions. The compact support of the shape functions means that the equation can be written as:

$$\sum_{e \in I} \int_{\Omega_e} \frac{\partial U^*}{\partial t} N_I d\Omega = \sum_{j=1}^{3} \sum_{e \in I} \int_{\Omega_e} F^j(U^*) \frac{\partial N_I}{\partial x_j} d\Omega - \sum_{j=1}^{3} \sum_{e \in I} \int_{\Gamma_e} F^j(U^*) n^j N_I d\Gamma \qquad (3.84)$$

Here, the summation extends only over those elements containing grid node I.

After inserting the assumed form for U^* into this equation, the left-hand integrals can be evaluated as:

$$\sum_{e \in I} \int_{\Omega_e} \frac{\partial U^*}{\partial t} N_I d\Omega = \sum_{e \in I} \left\{ \int_{\Omega_e} N_I N_J d\Omega \right\} \frac{dU_J}{dt} \equiv M_{IJ} \frac{dU_J}{dt} \qquad (3.85)$$

Here, M denotes the finite element consistent mass matrix, which is usually lumped.

The integrals appearing on the right-hand side of the equation (3.84) are evaluated approximately by using a Lobatto-typed quadrature over each tetrahedral element. For an element e with nodes, $I, J, K,$ and L, the integral can be approximated as:

$$\int_{\Omega_e} F^j \frac{\partial N_I}{\partial x_j} d\Omega \approx \frac{\Omega_e}{4} \left. \frac{\partial N_I}{\partial x_j} \right|_e \left\{ F_I^j + F_J^j + F_K^j + F_L^j \right\} \tag{3.86}$$

However, the equation (3.84) can be reinterpreted and evaluated in a different way. Assuming that an interior node I is directly connected to n_i nodes, $J_1, J_2, \ldots, J_{n_i}$, through mesh edges, and using the results of the equation (3.85) and (3.86), the equation (3.84) can be written as:

$$[M_L]_I \Delta U_I = \Delta t \sum_{j=1}^{3} \sum_{s=1}^{n_i} C_{IJ_s}^j (F_I^j + F_{J_s}^j) \tag{3.87}$$

For the nodes on the computational boundary surfaces, the above equations need suitable modifications.

The coefficients in the equation (3.87) are computed as:

$$C_{IJ_s}^j \equiv \sum_{e \in IJ_s} \left. \frac{\Omega_e}{4} \frac{\partial N_I}{\partial x_j} \right|_e \tag{3.88}$$

Here, the summation extends only over those elements containing edge IJ_s.

It can easily be verified that these coefficients possess the following properties:

$$\sum_{s=1}^{n_i} C_{IJ_s}^{j} = 0 \quad \text{for all j} \tag{3.89}$$

$$C_{IJ_s}^{j} = -C_{J_s I}^{j} \quad \text{for all j and } J_s \tag{3.90}$$

For convenience, C_{IJ_s} is defined as:

$$C_{IJ_s} \equiv (C_{IJ_s}^{1}, C_{IJ_s}^{2}, C_{IJ_s}^{3}) \tag{3.91}$$

After representing the modulus and the unit vector of C_{IJ_s} by Ψ_{IJ_s} and S_{IJ_s}, respectively, the equation (3.87) can rewritten as:

$$[M_L]_I \Delta U_I = \Delta t \sum_{s=1}^{n_i} \Psi_{IJ_s} (f_I + f_{J_s}) \tag{3.92}$$

and $$f_I \equiv \left\{F_I^1, F_I^2, F_I^3\right\} \cdot S_{IJ_s}, \; f_{J_s} \equiv \left\{F_{J_s}^1, F_{J_s}^2, F_{J_s}^3\right\} \cdot S_{IJ_s} \tag{3.93}$$

3.3.3 Artificial dissipation

The stabilization can be accomplished by explicitly adding the artificial viscosity to the right-hand side of the equation (3.92). The artificial viscosity should not deteriorate the overall accuracy of the algorithm and should serve two purposes: avoiding the appearance of checker boarding modes in smooth regions and providing sufficient

damping in the neighborhood of flow discontinuities, such as shocks. An effective way to implement the artificial viscosity is to split the flux into convective and pressure terms as following:

$$f_I = f_I^c + U_I^p \tag{3.94}$$

where

$$f_I^c = (u_I . S_{IJ_s}) \begin{bmatrix} \rho \\ \rho u_i \\ \rho H \end{bmatrix} = (u_I . S_{IJ_s}) U_I^* \tag{3.95}$$

$$U_I^p = \begin{bmatrix} 0 \\ p S_{IJ_s} \\ 0 \end{bmatrix} \tag{3.96}$$

The equation (3.92) can be finally written as:

$$[M_L]_I \Delta U_I = \Delta t \sum_{s=1}^{n_i} \Psi_{IJ_s} \{ f_I + f_{J_s} - k_1(M_{IJ_s}) a_{IJ_s} (U_{J_s}^* - U_I^*) - k_2(M_{IJ_s})(U_{J_s}^p - U_I^p) \} \tag{3.97}$$

where M_{IJ_s} denotes the local Mach number and is calculated as:

$$M_{IJ_s} = \frac{u_{IJ_s} \cdot S_{IJ_s}}{a_{IJ_s}} \tag{3.98}$$

The functions k_1 and k_2 in the equation (3.97) are given as:

$$k_1(M) \equiv \begin{cases} \frac{1+M^2}{2} & |M \le 1| \\ |M| & |M| > 1 \end{cases}; \; k_2(M) \equiv \begin{cases} \frac{M(3-M^2)}{2} & |M| \le 1 \\ sign(M) & |M > 1| \end{cases} \tag{3.99}$$

The role of these blending functions is to ensure that a suitable domain of dependence is adopted for subsonic flows, while a full upwind scheme is recast for supersonic flows. The equation (3.97) has the property of preserving a constant enthalpy in steady inviscid flows. The higher-order spatial accuracy can be constructed by employing the gradient information. By independently reconstructing all the components of U^* and U^p, density, velocity, pressure, and enthalpy, a higher-order scheme can be written as:

$$[M_L]_I \Delta U_I = \Delta t \sum_{s=1}^{n_i} \Psi_{IJ_s} \left\{ f_1 + f_2 - d_v D_{IJ_s} \right\} \tag{3.100}$$

where d_v is a user-defined constant and,

$$D_{IJ_s} = k_1(M_{IJ_s}) a_{IJ_s} \left[\left\{ U^*_{J_s} - (\frac{\partial U^*_{J_s}}{\partial x_j})_{J_s} \frac{\Delta\sigma}{2} \right\} - \left\{ U^*_{J_s} + (\frac{\partial U^*_{J_s}}{\partial x_j})_I \frac{\Delta\sigma}{2} \right\} \right]$$

$$+ k_2(M_{IJ_s}) \left[\left\{ U^p_{J_s} - (\frac{\partial U^p_{J_s}}{\partial x_j})_{J_s} \frac{\Delta\sigma}{2} \right\} - \left\{ U^p_{J_s} + (\frac{\partial U^p_{J_s}}{\partial x_j})_I \frac{\Delta\sigma}{2} \right\} \right] \tag{3.101}$$

This chapter describes the details of methodology used in the present research. The dual-time stepping scheme was validated by simulating transonic flows around a pitching airfoil and a pitching rectangular wing, present in Chapter 4. The Hänel upwind scheme was employed to simulate the staging of a hypersonic vehicle on the longitudinal symmetry plane, present in the second section of Chapter 5. The three dimensional simulation of a hypersonic vehicle staging was carried out using the central differencing scheme plus artificial viscosity, present in the third section of Chapter 5.

CHAPTER 4

VALIDATION OF DUAL-TIME STEPPING SCHEME

4.1 Introduction

The computation of flows around a sinusoidally oscillating airfoil or wing is a useful approach to analyzing the nonlinear interaction between the flow and the structure response. This chapter will validate the efficiency of the dual-time stepping scheme, as compared to an explicit scheme, by simulating unsteady flows around the NACA0012 airfoil and the ONERA OM6 wing, undergoing a pitching motion with a low-reduced frequency. The subiterations in the implicit scheme were accelerated by a multigrid method when the convergence rate of subiterations was slow. The results from the implicit scheme were compared to those from the multistage Runge-Kutta explicit scheme, which were validated by available experimental data [51,54].

The accuracy of a computational solution is normally related to the following factors: the size and the quality of the mesh, the physical models and the numerical algorithms. This chapter mainly focuses on validating the efficiency of the implicit scheme, not on the accuracy of solutions. Therefore, it is assumed that the baseline solver on the specific meshes has acquired accurate results. Improving the efficiency of computations while not compromising the accuracy of solutions, as compared to the explicit scheme, is one of the main objectives of the present research.

All the accelerating techniques do not improve the accuracy of solutions for static problems. But, the accelerated subiterations in the dual-time stepping scheme have been proven to play a crucial role in acquiring better solutions, especially when the number of

subiterations is fixed as a small number in order to save CPU time as much as possible. Among all the accelerating techniques, the multigrid method is the most powerful.

4.2 Pitching Airfoil with Low-Reduced Frequency

4.2.1 Explicit scheme solution

The loading loop on the pitching NACA0012 airfoil was obtained by the explicit scheme first and was taken as a benchmark for the implicit scheme. The mesh was generated by the advancing front method, and the whole computational domain was discretized into 6,275 triangles and 3,214 nodes. A mesh window was built up around the airfoil in the shape of an ellipses, enclosing 2,816 nodes and 5,451 triangles. A mesh-independent study was carried out [3]. Figures 4.1 and 4.2 show the original mesh and Mach and pressure contours in a steady-state solution, respectively.

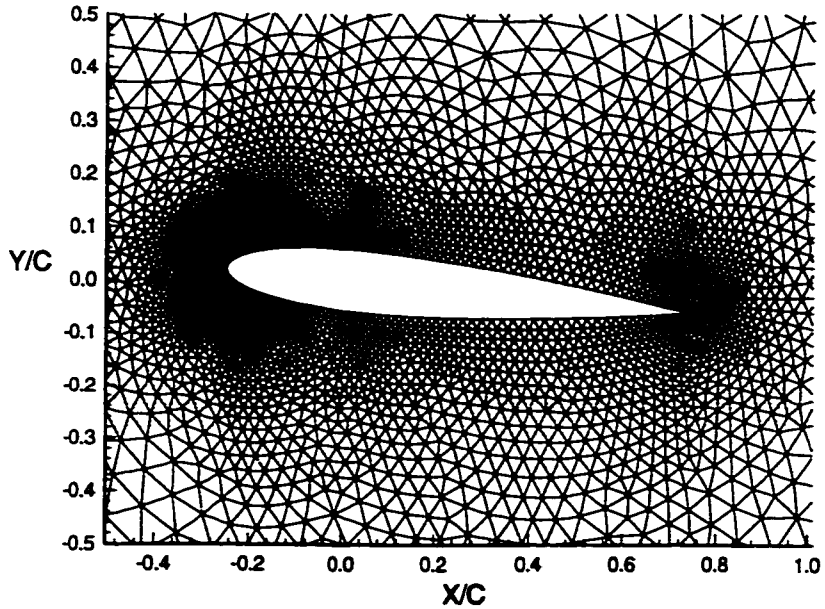


Figure 4.1 Static mesh of Naca0012 airfoil.

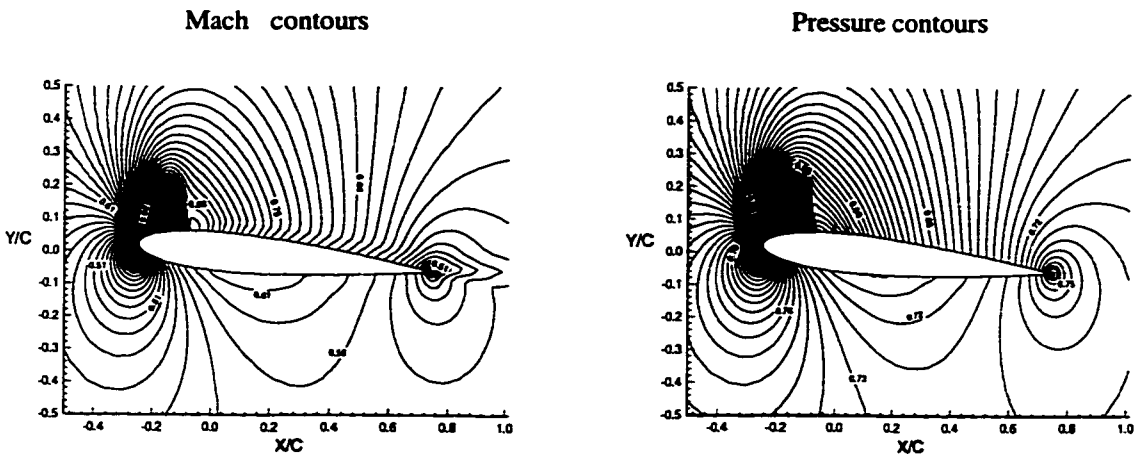


Figure 4.2 Off-surface mach and pressure contours at steady state.

The airfoil was oscillated about its quarter chord with the mean incidence and amplitude equal to 4.86° and 2.44°, respectively. The pitching motion is written in the form of angle of attack as:

$$\alpha(t) = \alpha_0 + \alpha_a \sin(M_\infty kt) \tag{4.1}$$

In the present research, the free-stream Mach number was equal to 0.6. The non-dimensional reduced frequency, k, was equal to 0.081, which is the measure of the unsteadiness of flows, and is defined as:

$$k = \frac{fc}{2U_\infty} \tag{4.2}$$

where f is the circular frequency; c is the chord length; U_∞ is the free-stream velocity.

In the explicit scheme, the minimal value of all the allowable time steps from the local numerical stability analysis was taken as the global time step to advance the governing equations in time domain. Even the mesh used was rather coarse, shown in figure 4.1, the final non-dimensional global time step was just 0.002. With the non-dimensional period equal to 129.284, the above time step resulted in 64,642 iterations per cycle. It took 80,944 seconds of CPU on a SUN-SPARC-10 workstation, which was unacceptable from the point-of-view of engineering applications. Therefore, it is highly desirable to develop a new robust method to handle the problems with large time domain, in which the time steps uniquely come from the concerns of accuracy, not the numerical stability. Figure 4.3 shows the instantaneous pressure distributions along the airfoil, as compared to experimental data [54].

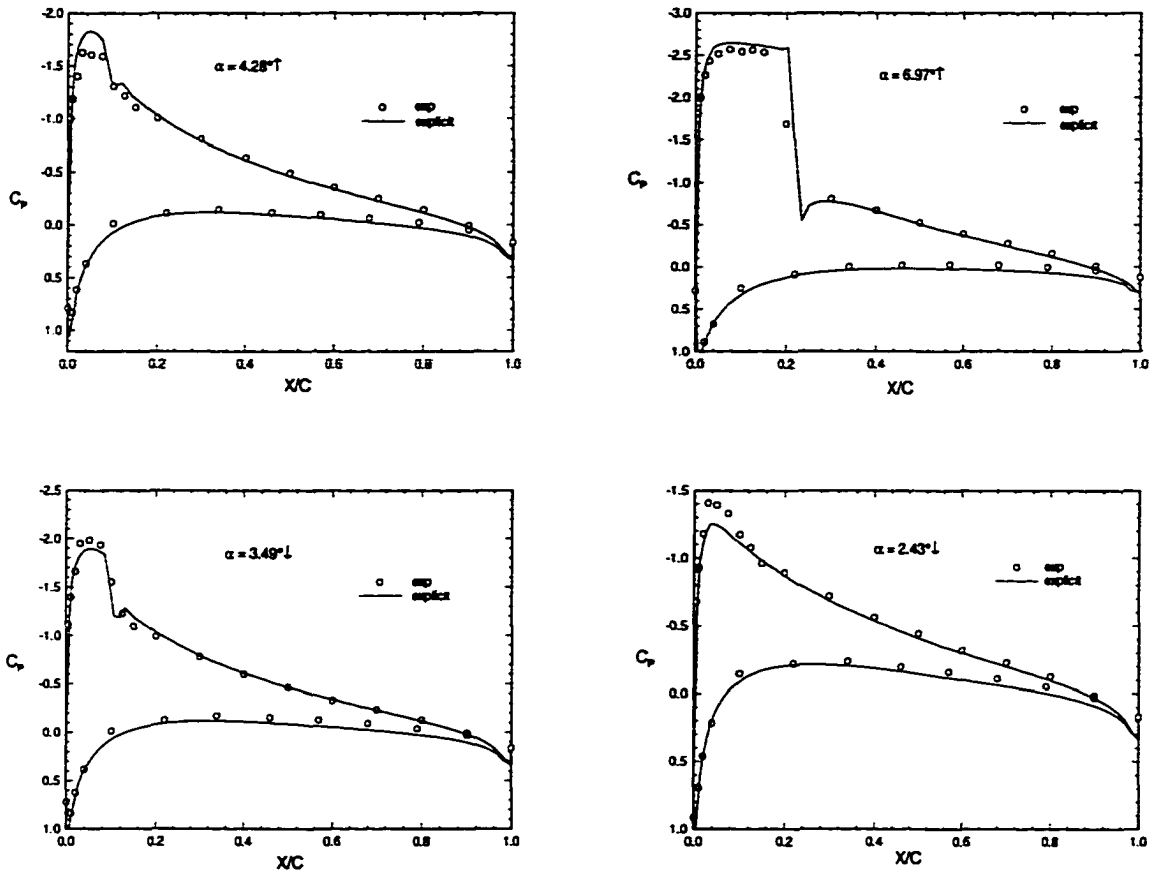


Figure 4.3 Comparison of instantaneous pressure coefficient with experiment [54].

In figure 4.3, a good agreement is observed between the computational and the experimental data, except for small discrepancies over the leading edge. This is inherently credited to the viscous terms omitted in the present Euler computations and the quality of mesh which was rather coarse but a good chance to show the efficiency of the dual-time stepping scheme.

4.2.2 Implicit scheme solution

The time step for the dynamic mesh module in an implicit scheme can be much larger than that in an explicit scheme. However, a small time step is desirable to make the process of modifying an existing mesh nonlinear in the spring analogy method. Therefore, it is necessary to improve the quality of the dynamic mesh module when a large time step is employed to take the advantages offered by an implicit scheme. Usually, an ideal node distribution, and whether or not the Delaunay property is violated in a deformed mesh, are two concerns in the spring analogy method. With the help of a Laplacian-typed smoothing, the node distribution without the sharp variation of area in the neighboring mesh cells is promised. One approach to retrieving the violation of the Delaunay property is the area correction, which is based on minimizing a cost function. Figure 4.4 presents an intentionally perturbed grid and its corresponding "ideal" grid. Figure 4.5 shows the resulting mesh after applying the area correction to the perturbed grid.

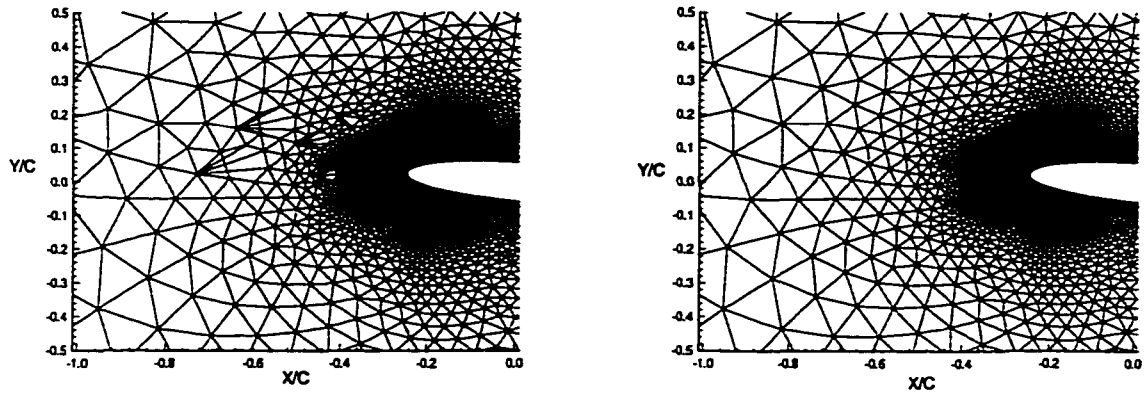


Figure 4.4 Perturbed and ideal meshes.

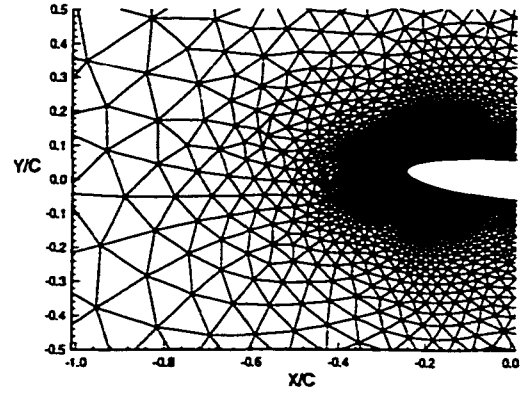


Figure 4.5 Corrected mesh.

To verify the implicit scheme and demonstrate its efficiency, two computations with the physical time steps equal to one thousand and five thousand times that of the explicit scheme were performed. In the first computation, each period was divided into 65 steps, while in the second one, only 13 steps existed. From the equation (2.41), the

parameter M plays an important role in evaluating the efficiency of the implicit scheme and is mainly determined by the efficiency of subiterations. In order to save as many CPUs as possible, the number of subiterations was fixed arbitrarily as 300. It should be stressed that the higher-order truncation errors are the only sources of errors after the subiterations are converged in the dual-time stepping scheme. Fortunately, the truncation errors are still bound in pitching problems with low-reduced frequency, even when large time steps are employed.

Figures 4.6-4.9 present the history of aerodynamic forces and the instantaneous pressure distributions along the airfoil in the first implicit computation, along with the results in the explicit scheme. Figure 4.10 shows the convergence characteristics of subiterations at different instants within a period. It is observed that in this computation, the CPUs spent per period is 0.37 times that of the explicit scheme. The difference between the results in the implicit scheme and the explicit scheme is minor, which is credited to the fast convergence of subiterations, as shown in figure 4.10. It should be mentioned here, that the residuals can only be dropped to 10^{-5} in a static solution by the baseline solver on the fine mesh due to the approach to implement the boundary conditions and the interpolation between a cell-center and a cell vertex, described in Chapter 3.

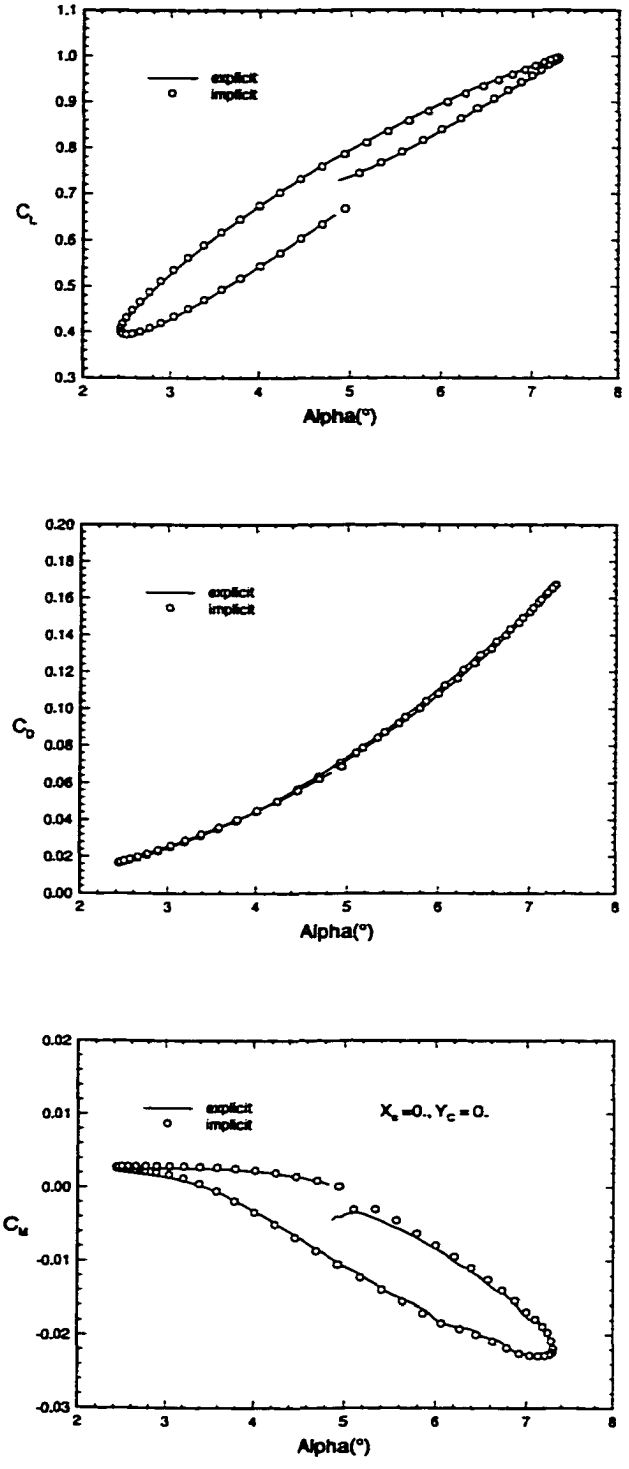
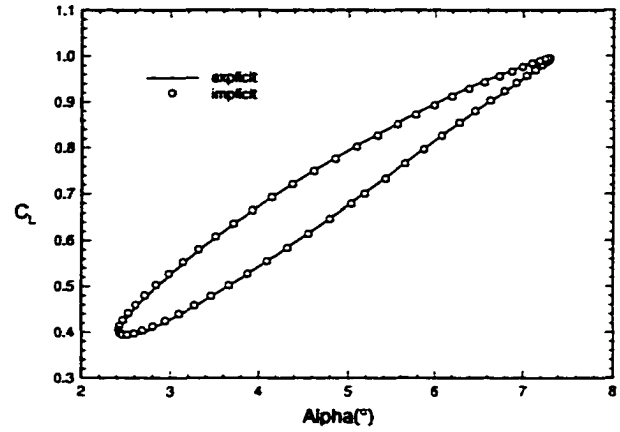
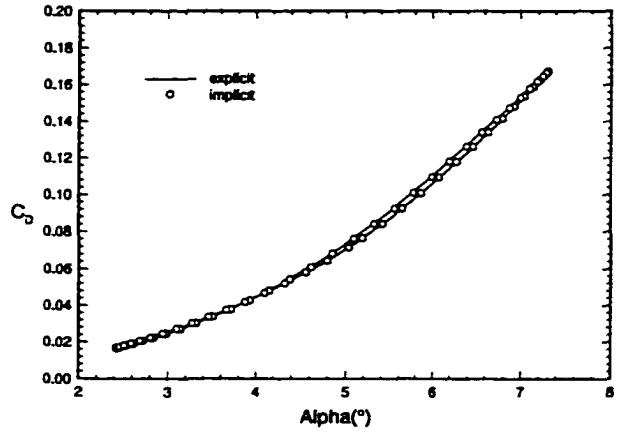


Figure 4.6 Aerodynamic forces in first cycle: $\Delta t_i = 1000\,\Delta t_e$.





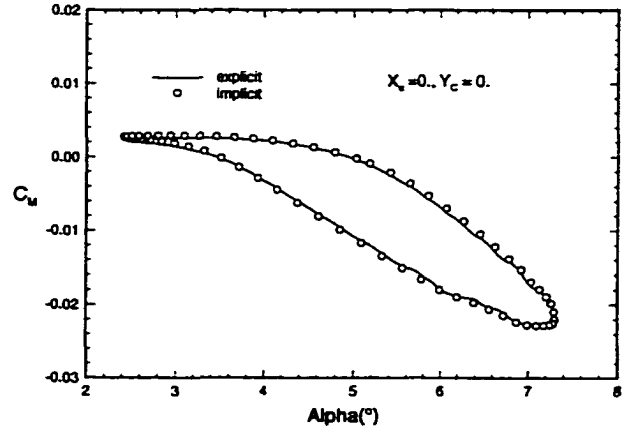
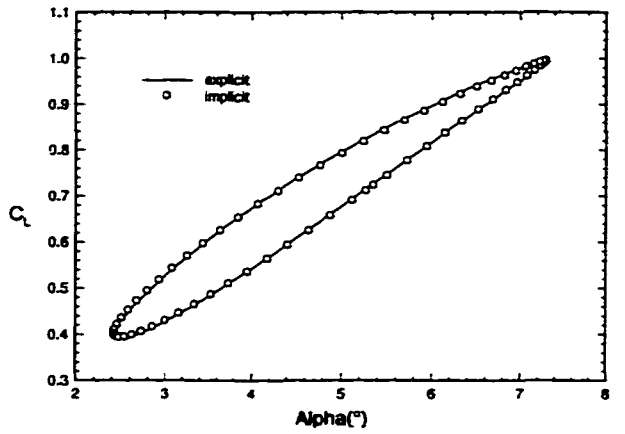
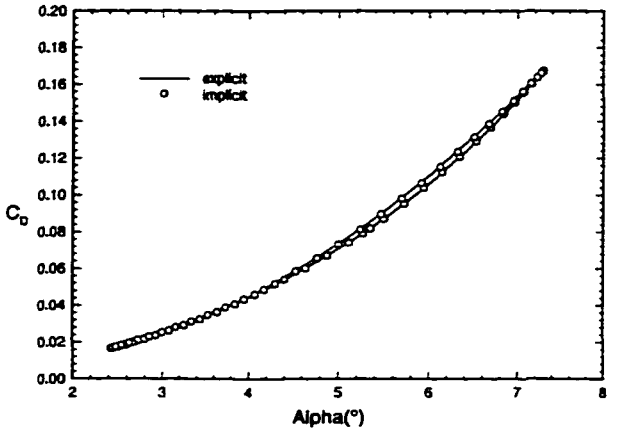


Figure 4.7 Aerodynamic forces in second cycle: $\Delta t_i = 1000\,\Delta t_e$.





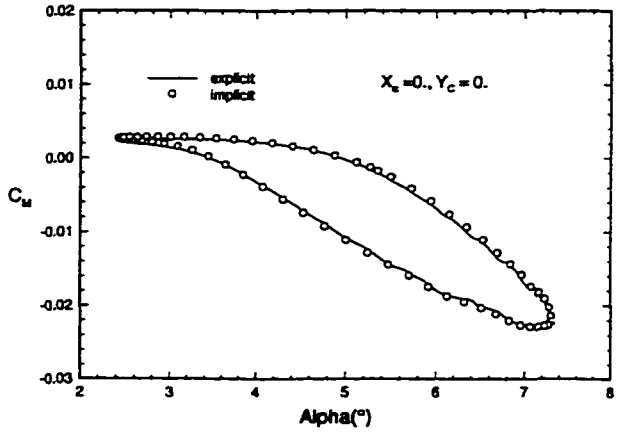


Figure 4.8 Aerodynamic forces in limit cycle: $\Delta t_i = 1000\,\Delta t_e$.

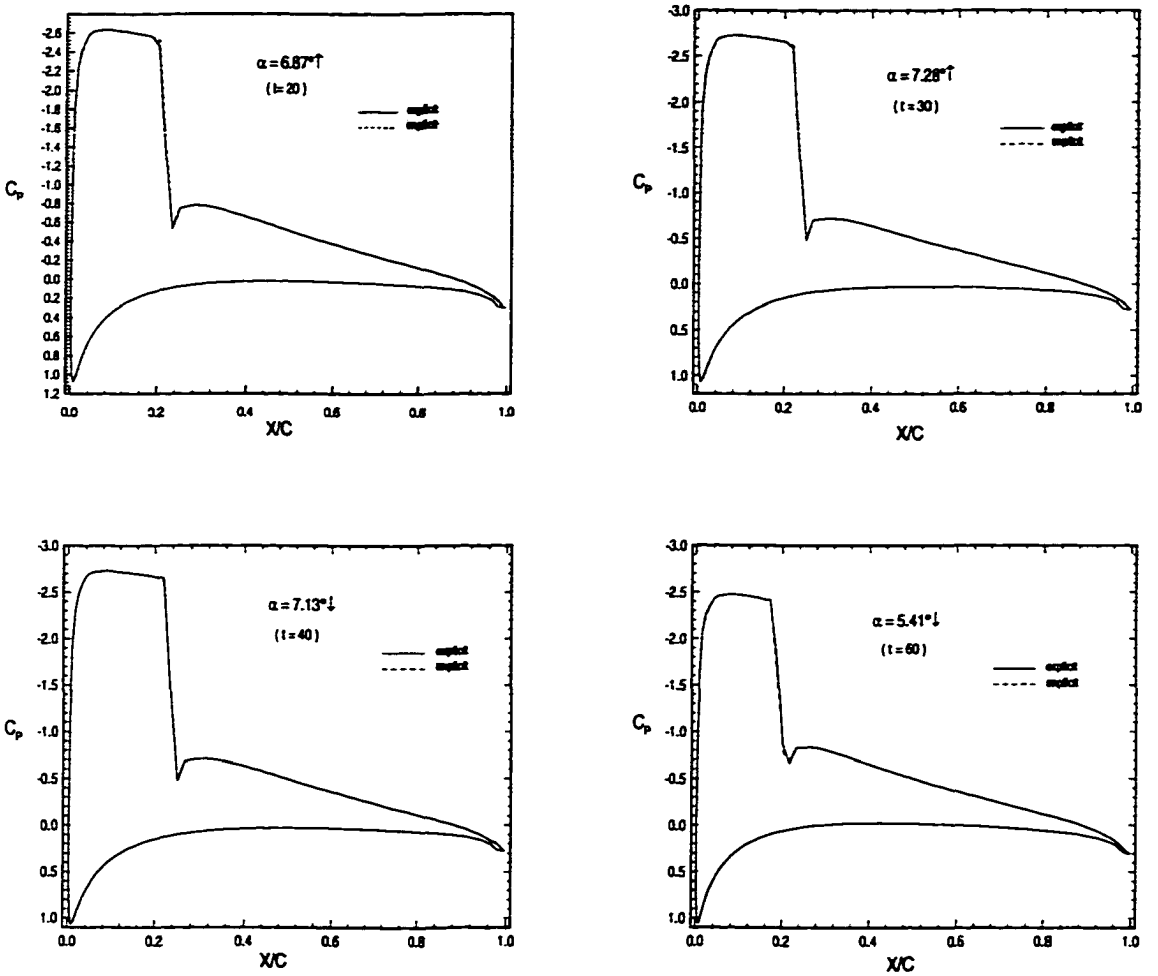


Figure 4.9 Instantaneous pressure distributions: $\Delta t_i = 1000\,\Delta t_e$.

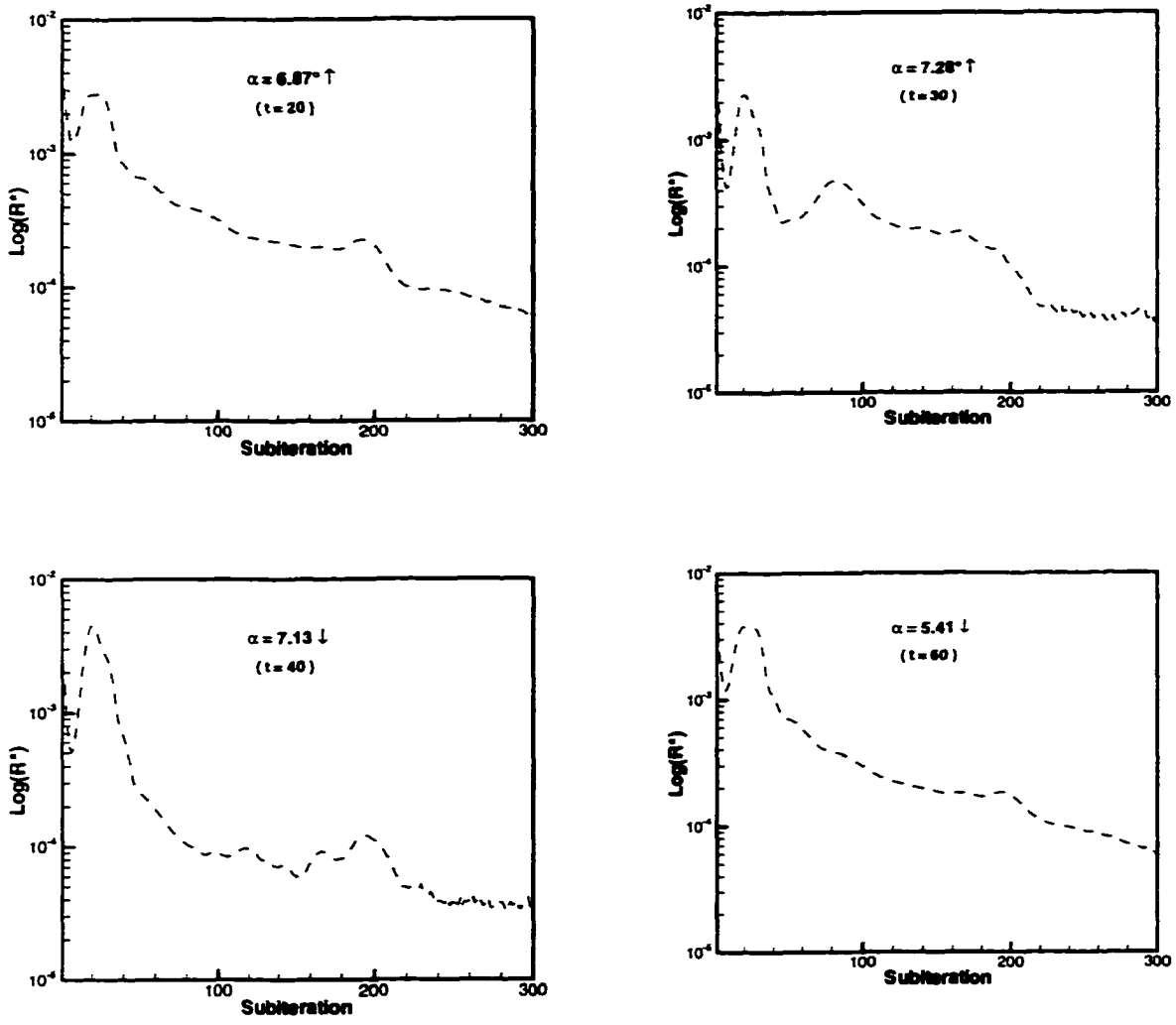


Figure 4.10 Convergence history of subiteration: $\Delta t_i = 1000\,\Delta t_e$.

As observed from the above figures, this computation did not take advantage of the dual-time stepping scheme completely. Approximately three times of CPU savings was achieved. Therefore, another computation was performed, in which the physical time step was equal to five thousand times that of the explicit scheme. Its results are present in the next paragraph.

Figures 4.11-4.14 display the history of aerodynamic forces and the instantaneous pressure distribution along the airfoil, respectively, along with those from the explicit scheme. Figure 4.15 shows the convergence characteristics of subiterations. In the above figures, the time step for the spring analogy method was five times that of the explicit scheme. Although the second implicit computation was about 14 times faster than the explicit scheme, the difference between the solutions in the implicit scheme and the explicit scheme was no longer minor, due to the slow convergence of subiterations, as shown in figure 4.15. It was concluded that without any aids of accelerating techniques, possibly more CPUs were needed in order to make the implicit solutions match with the explicit solutions, and the convergence of subiterations was highly related to the size of physical time step.

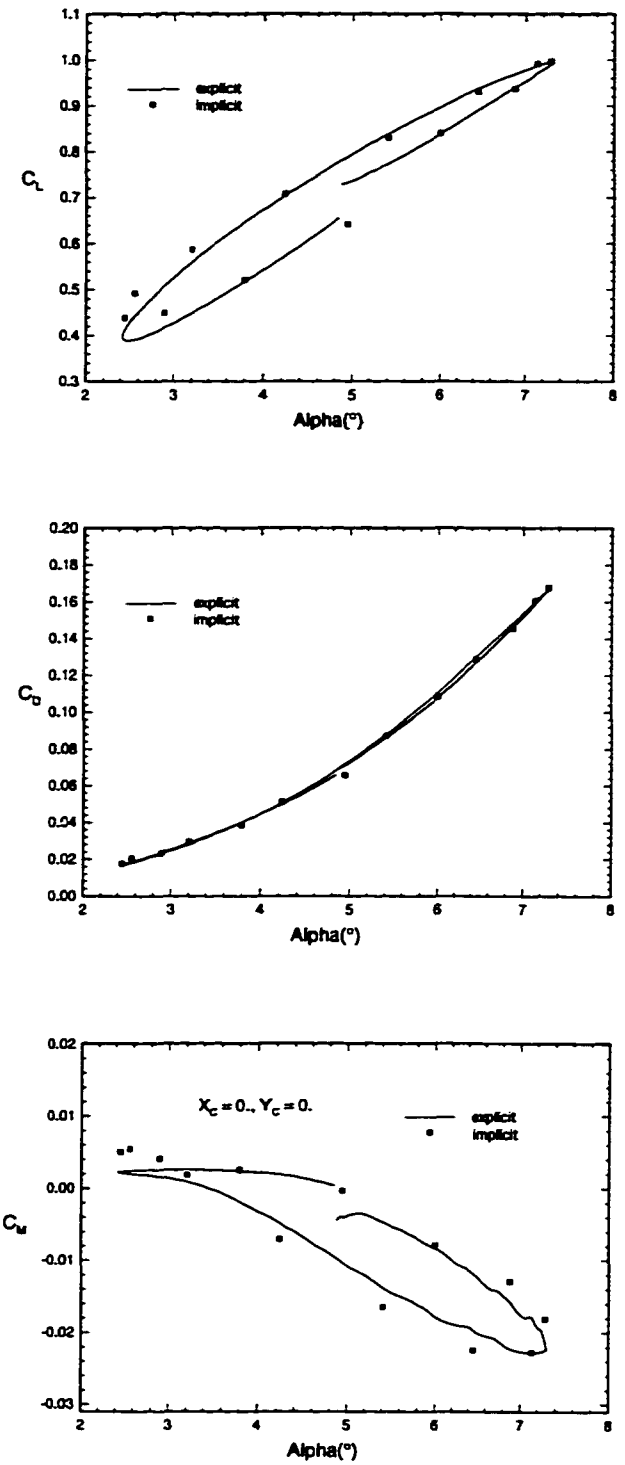


Figure 4.11 Aerodynamic forces in first cycle: $\Delta t_i = 5000\,\Delta t_e$.

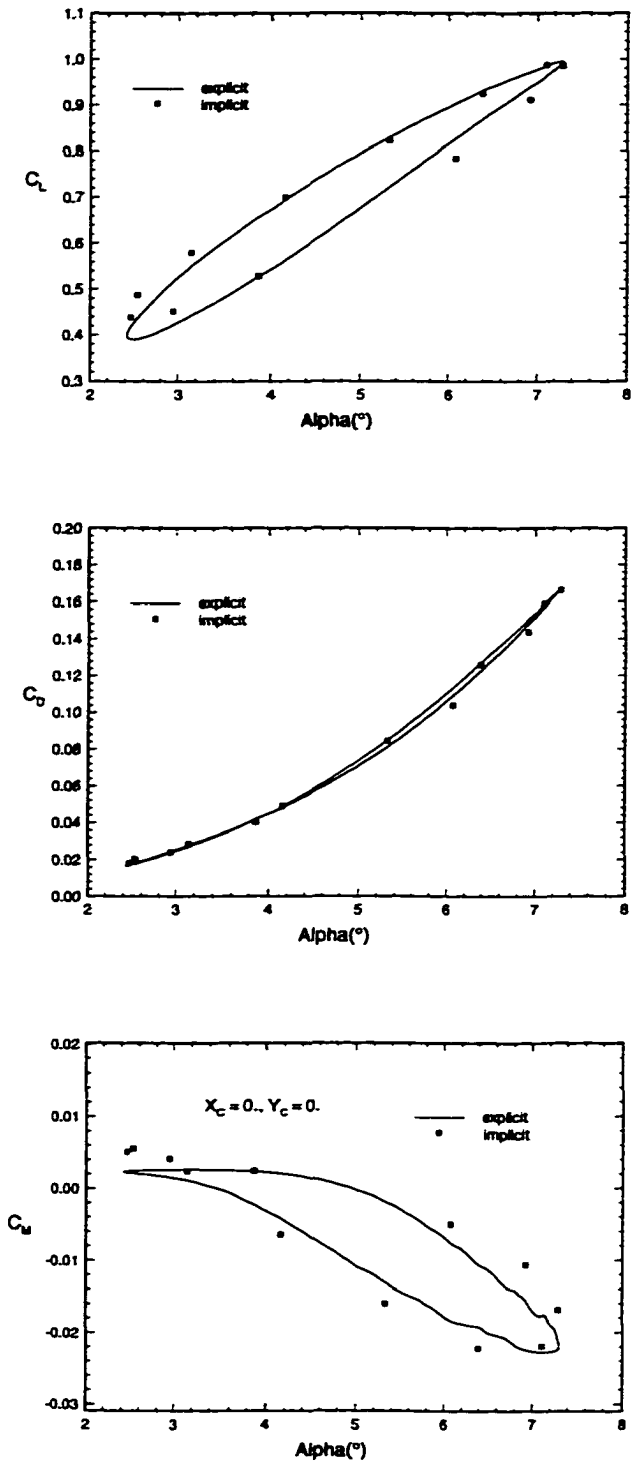


Figure 4.12 Aerodynamic forces in second cycle: $\Delta t_i = 5000\,\Delta t_e$.

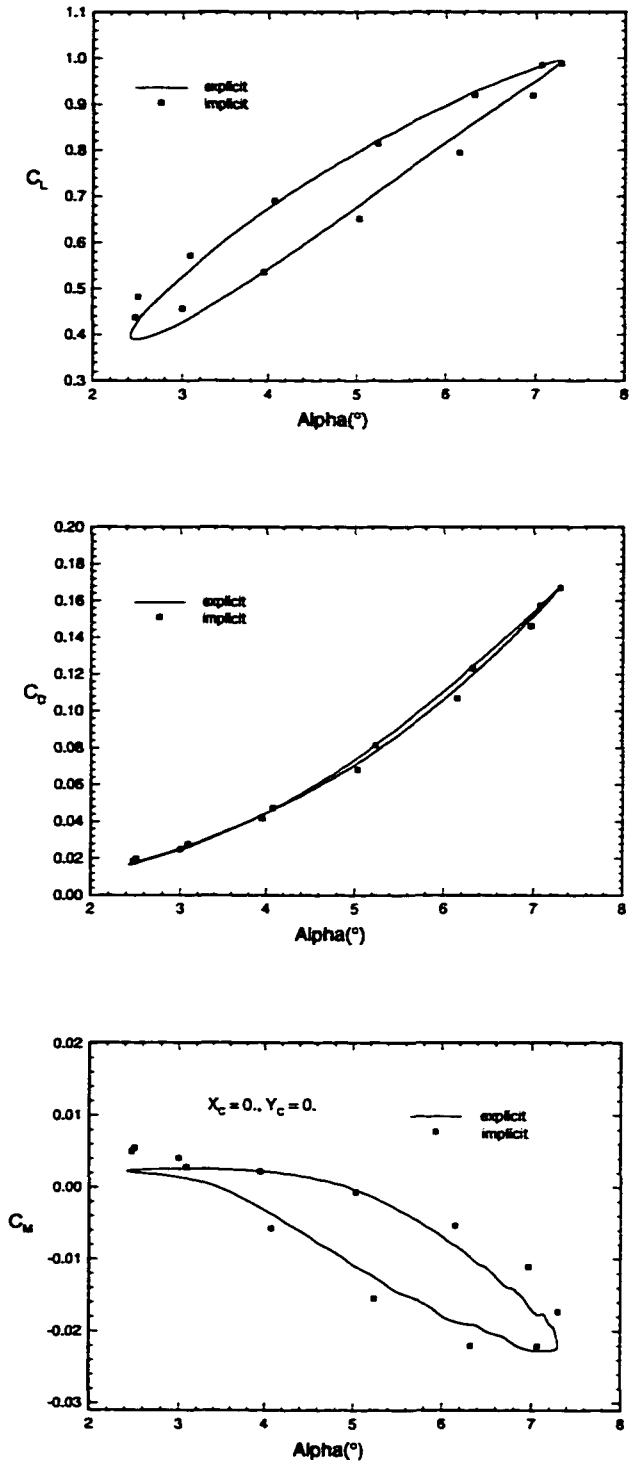


Figure 4.13 Aerodynamic forces in limit cycle: $\Delta t_i = 5000\,\Delta t_e$.

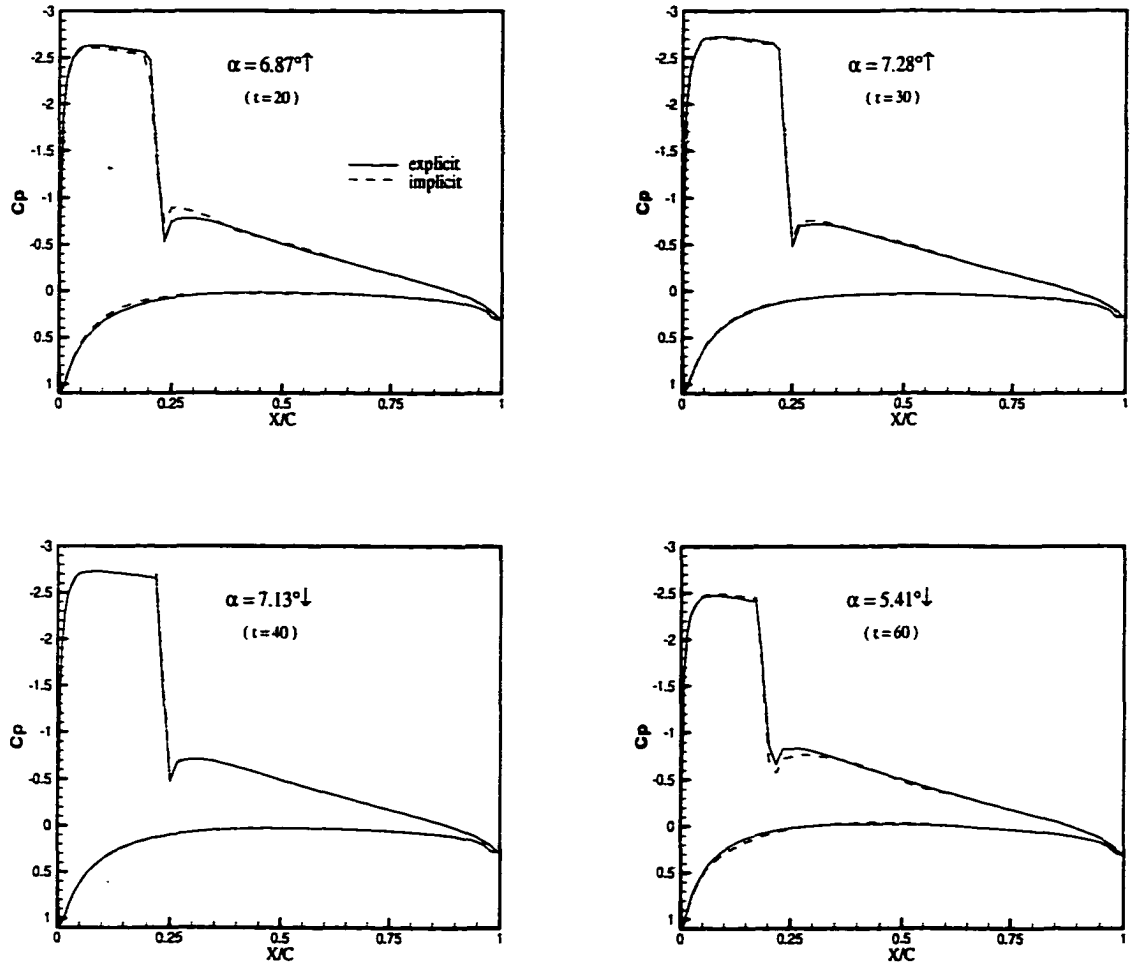


Figure 4.14 Instantaneous pressure distributions: $\Delta t_i = 5000\,\Delta t_e$.

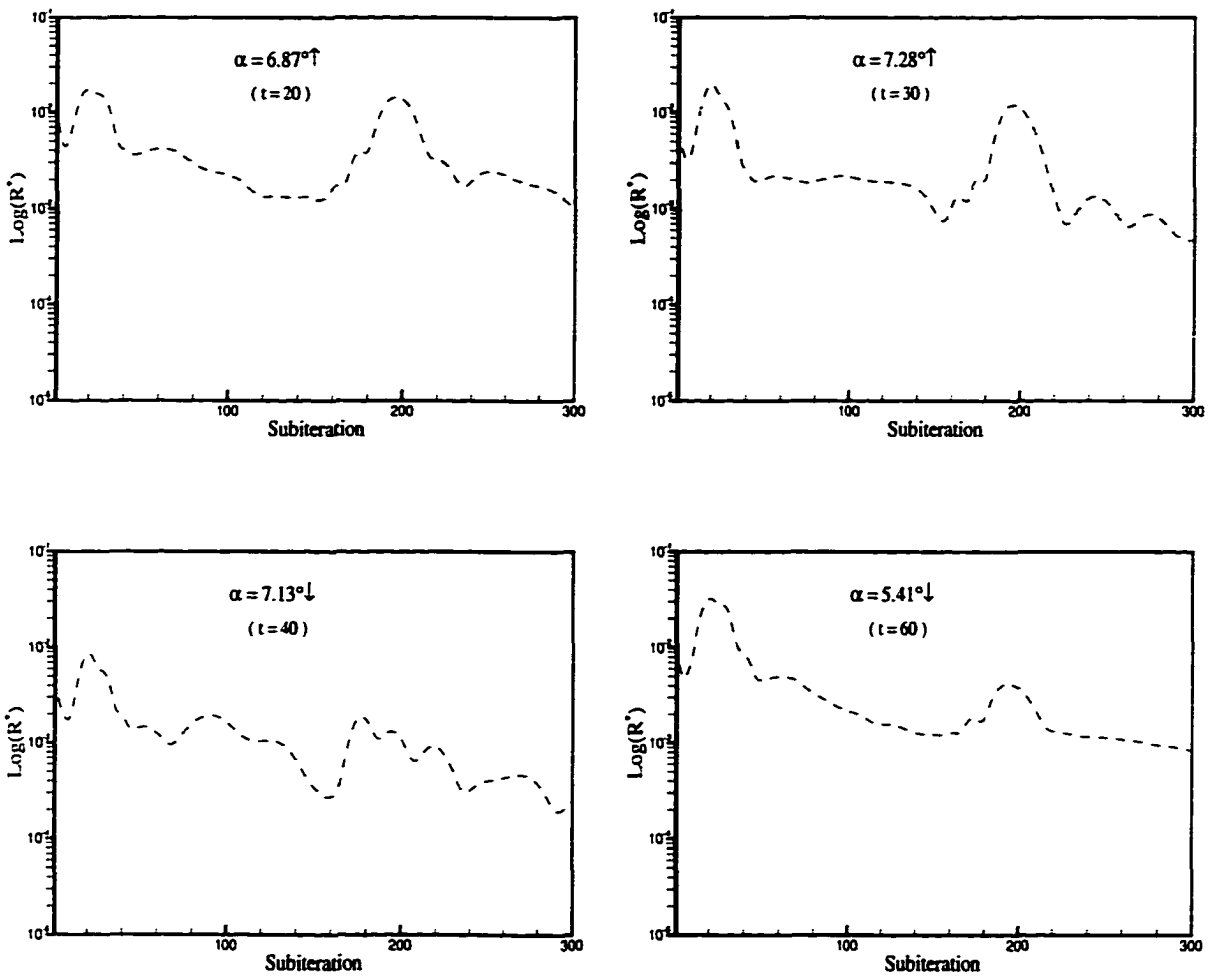


Figure 4.15 Convergence history of subiteraion: $\Delta t_i = 5000\,\Delta t_e$.

4.2.3 Multigrid accelerated implicit scheme solution

In the present research, two levels of overset-type coarse grids were generated by the advancing front method. The coarseness ratio was approximately 4.53 and 6.24. The coarse grids were generated independently of the fine grid, resulting in that the fine and coarse grids contained no common points. After constructing coarse grids, adaptive windows were built up in the coarse grids, and the coarse meshes were adjusted to the motion of the airfoil in the same way as the fine mesh. A simple V-type cycle strategy was employed in the communication of unknowns among these three meshes.

Figure 4.16 shows these meshes at different instants during a pitching cycle. The accelerated subiterations provided much better solutions within the fixed number of iterations. As shown in figures 4.17-4.19, the aerodynamic forces in the implicit scheme matched very well with those in the explicit scheme at corresponding instants. After spending some CPUs on additional coarse meshes and interpolations among meshes, the total CPUs spent per cycle was approximately 0.12 times that of the explicit scheme. Therefore, almost one order of CPU savings was achieved on a rather coarse mesh while not compromising the accuracy of solutions. More orders of CPU savings should be expected in the engineering applications having tens or hundreds of thousands of cells from the equation (2.41), where the numerical stability imposes a really serious limitation on the size of time steps. The instantaneous pressure distributions along the airfoil and the convergence characteristics of subiterations are shown in figures 4.20 and 4.21, respectively, while the instantaneous off-surface Mach and pressure contours are displayed in figures 4.22 and 4.23.

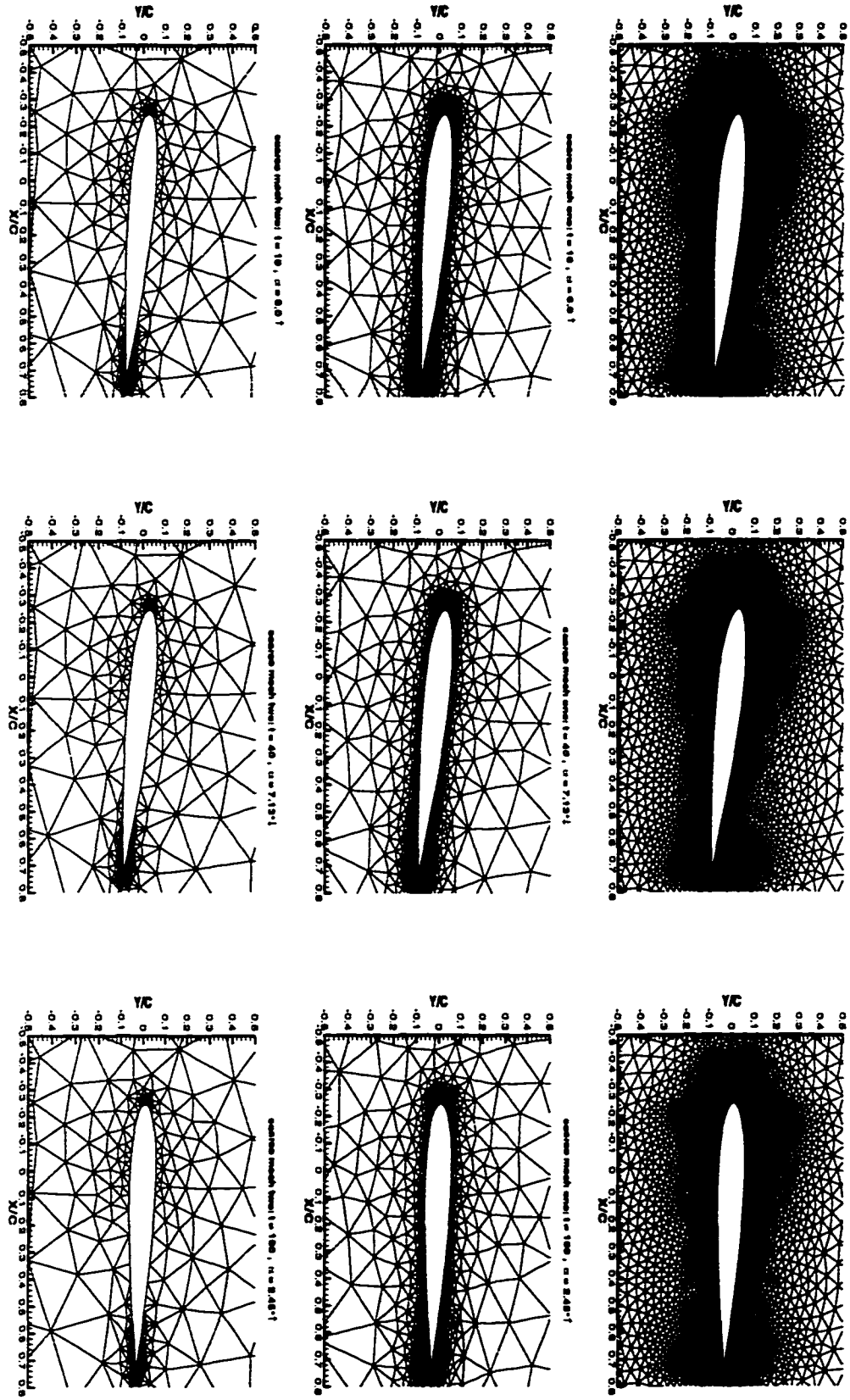
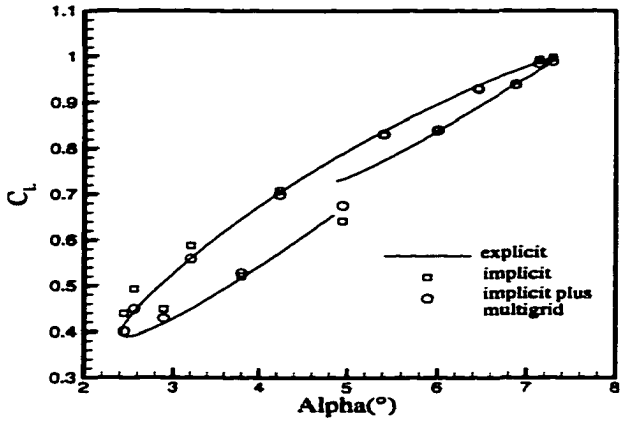
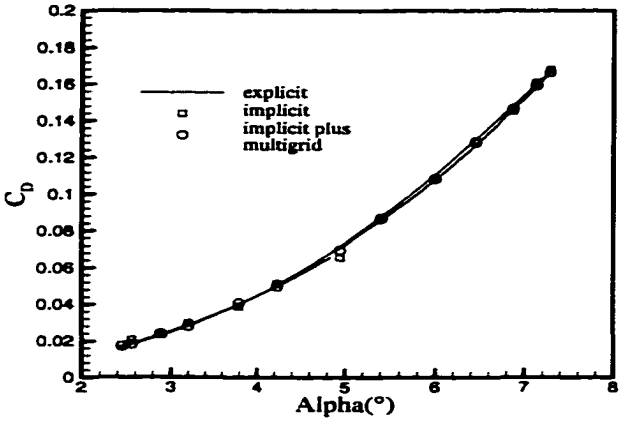


Figure 3.16 Three levels of meshes at three instants.





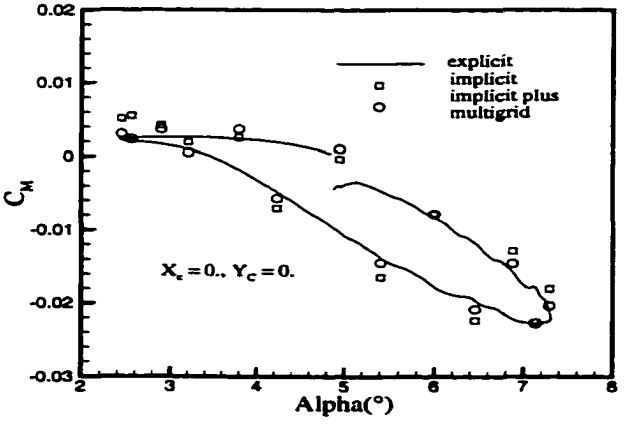
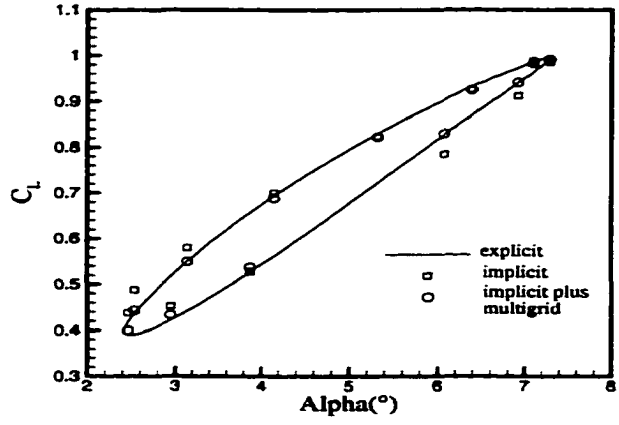
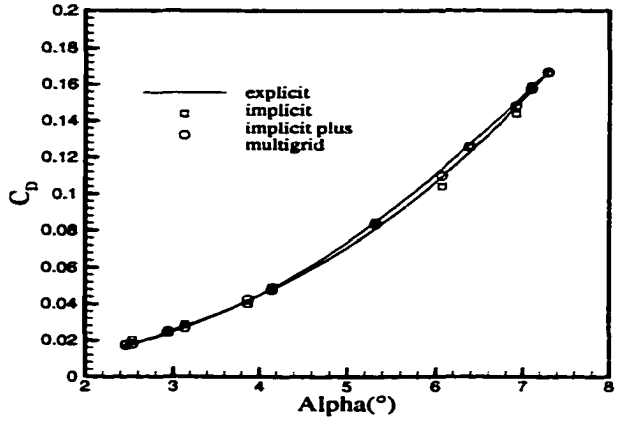


Figure 4.17 Aerodynamic force comparison in first cycle: $\Delta t_i = 5000\,\Delta t_e$.





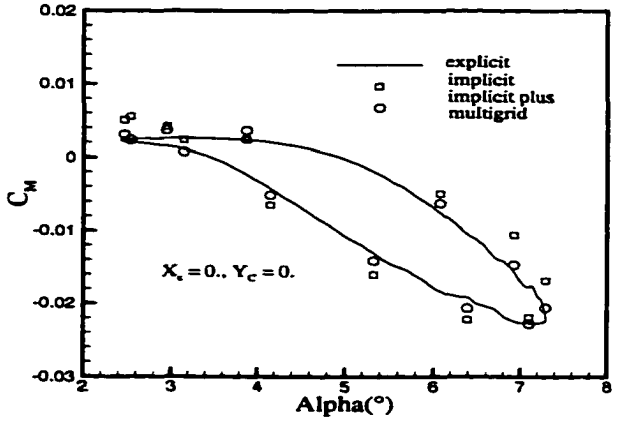


Figure 4.18 Aerodynamic force comparison in second cycle: $\Delta t_i = 5000\,\Delta t_e$.

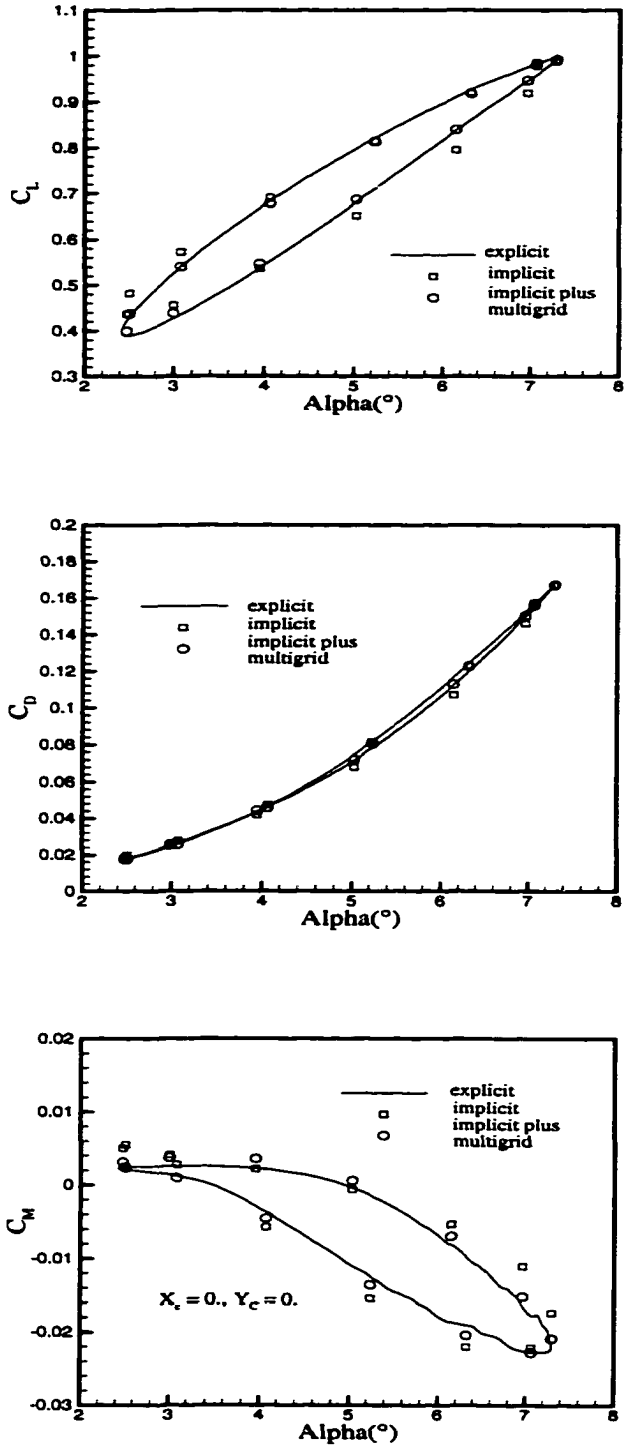


Figure 4.19 Aerodynamic force comparison in limit cycle: $\Delta t_i = 5000\,\Delta t_e$.

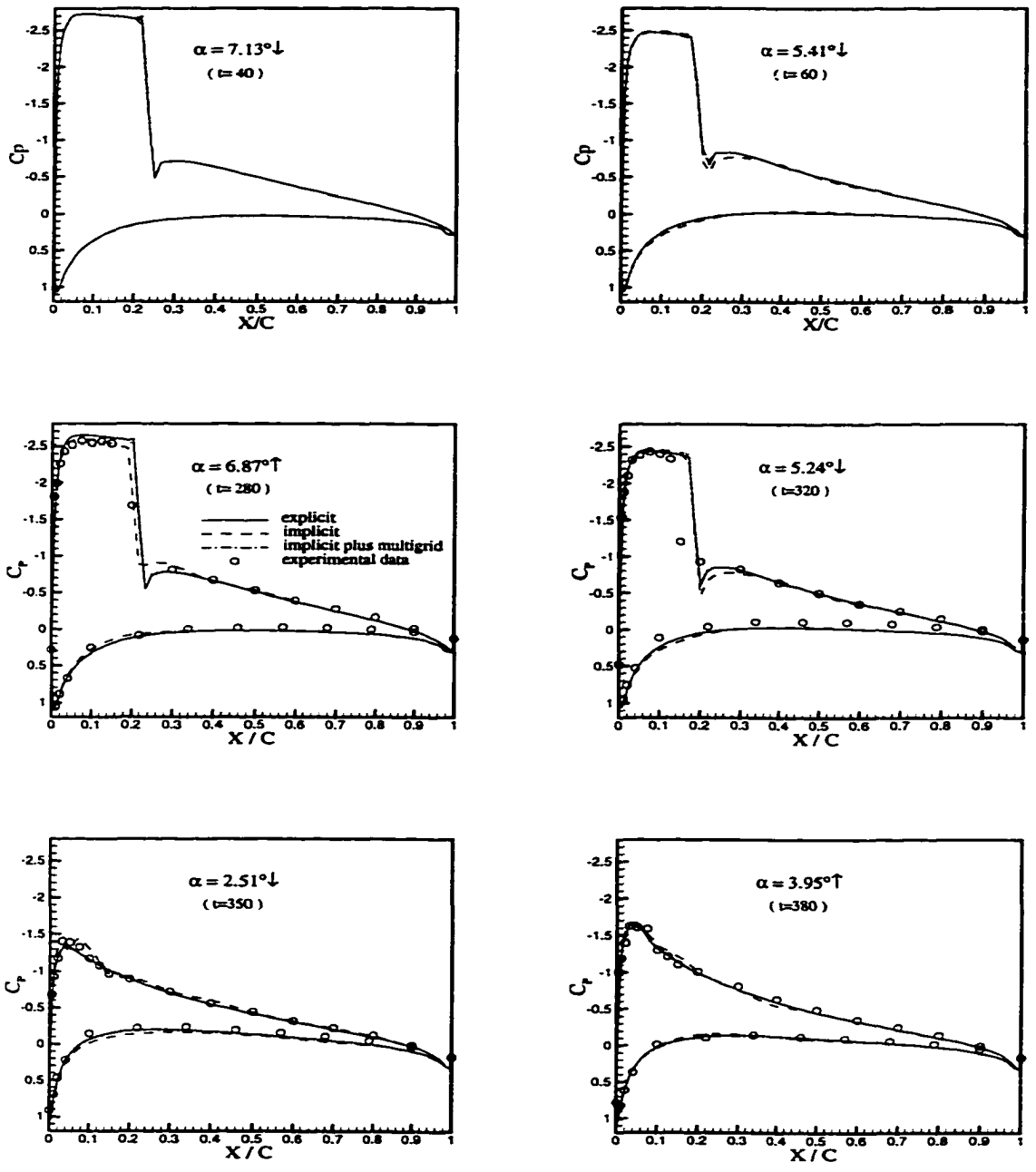


Figure 4.20 Instantaneous pressure distribution comparisons.

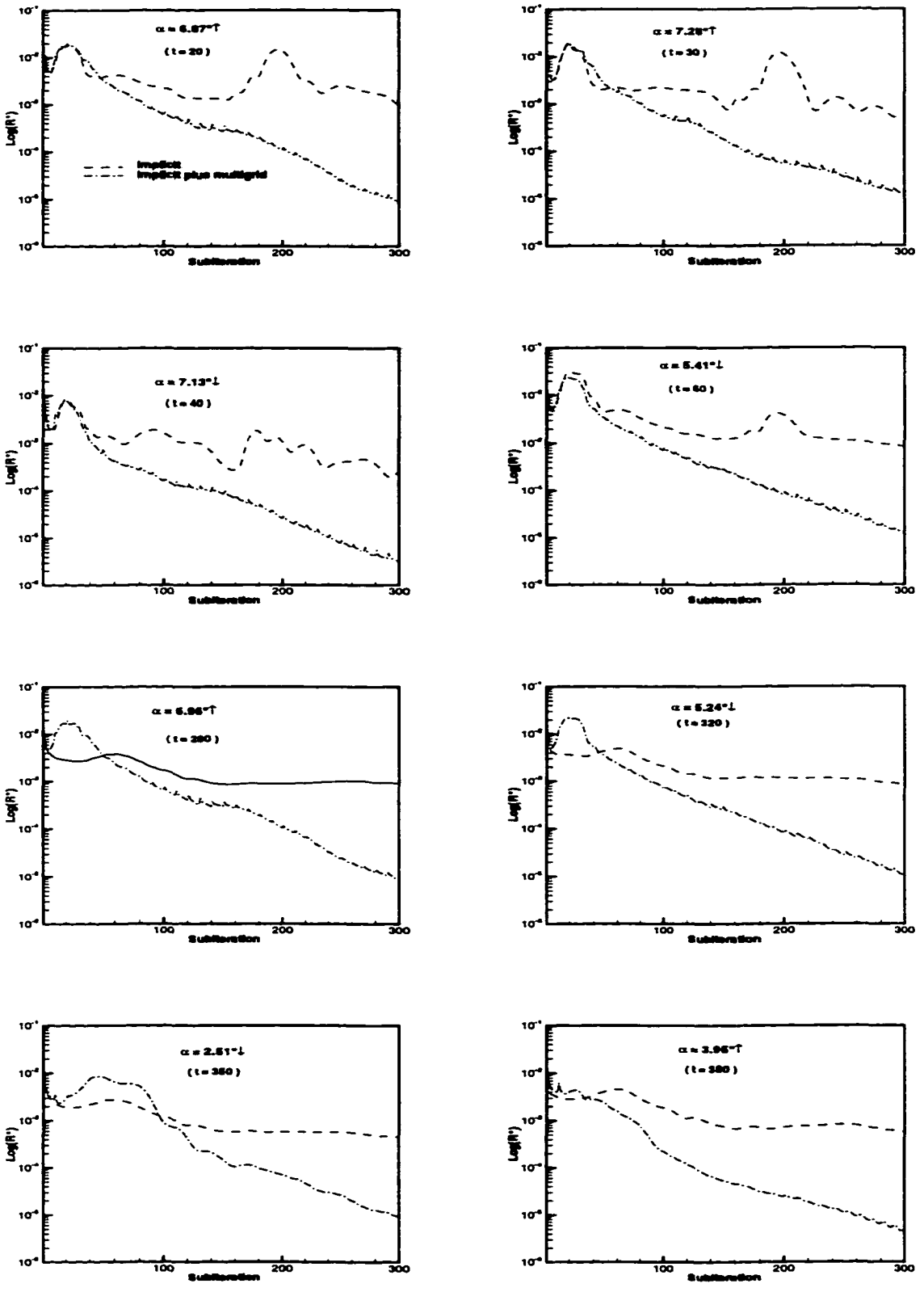


Figure 4.21 Convergence rate of subiterations at instants.

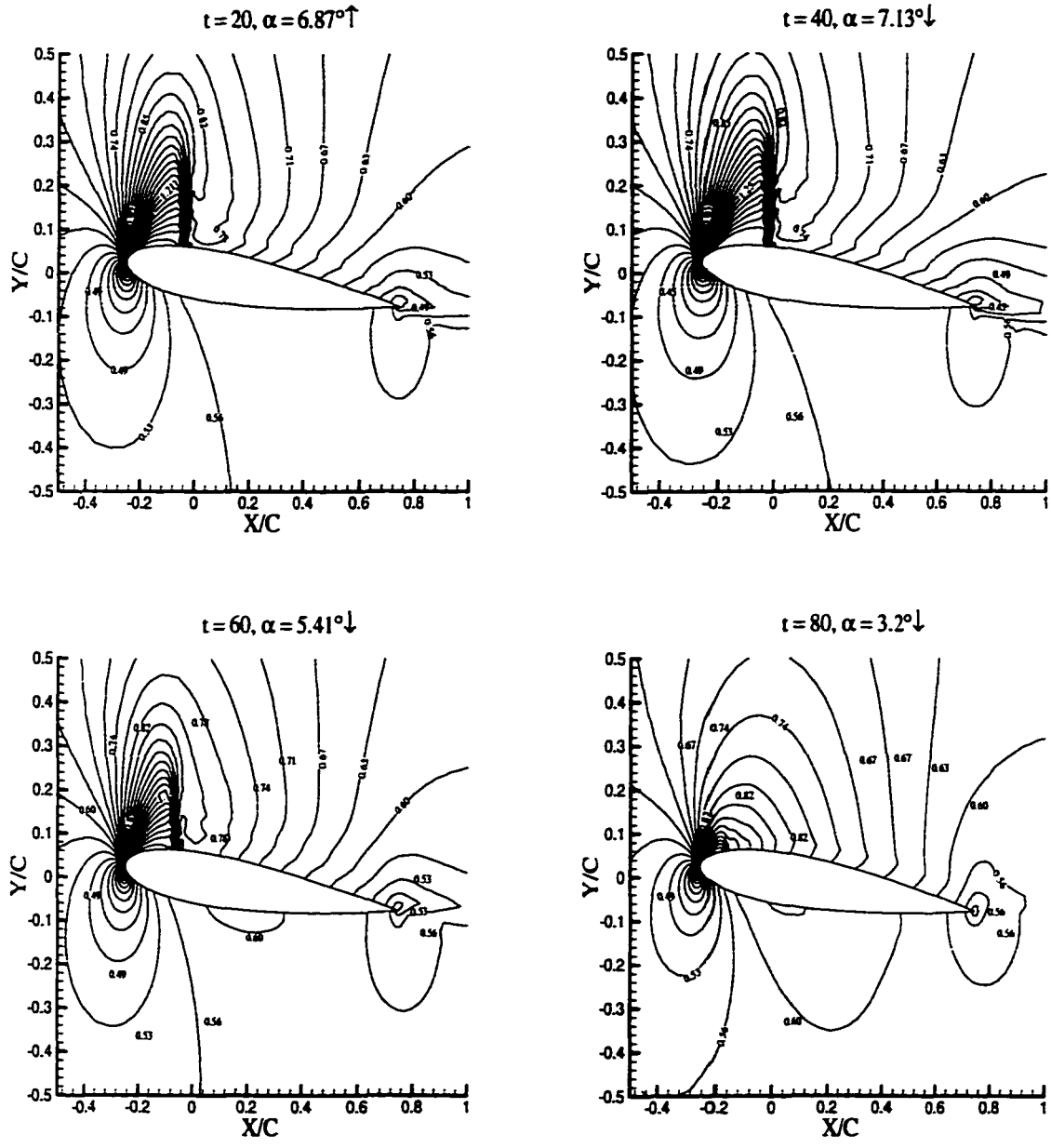


Figure 4.22 Instantaneous Mach contours with acceleration of mutligrid method.

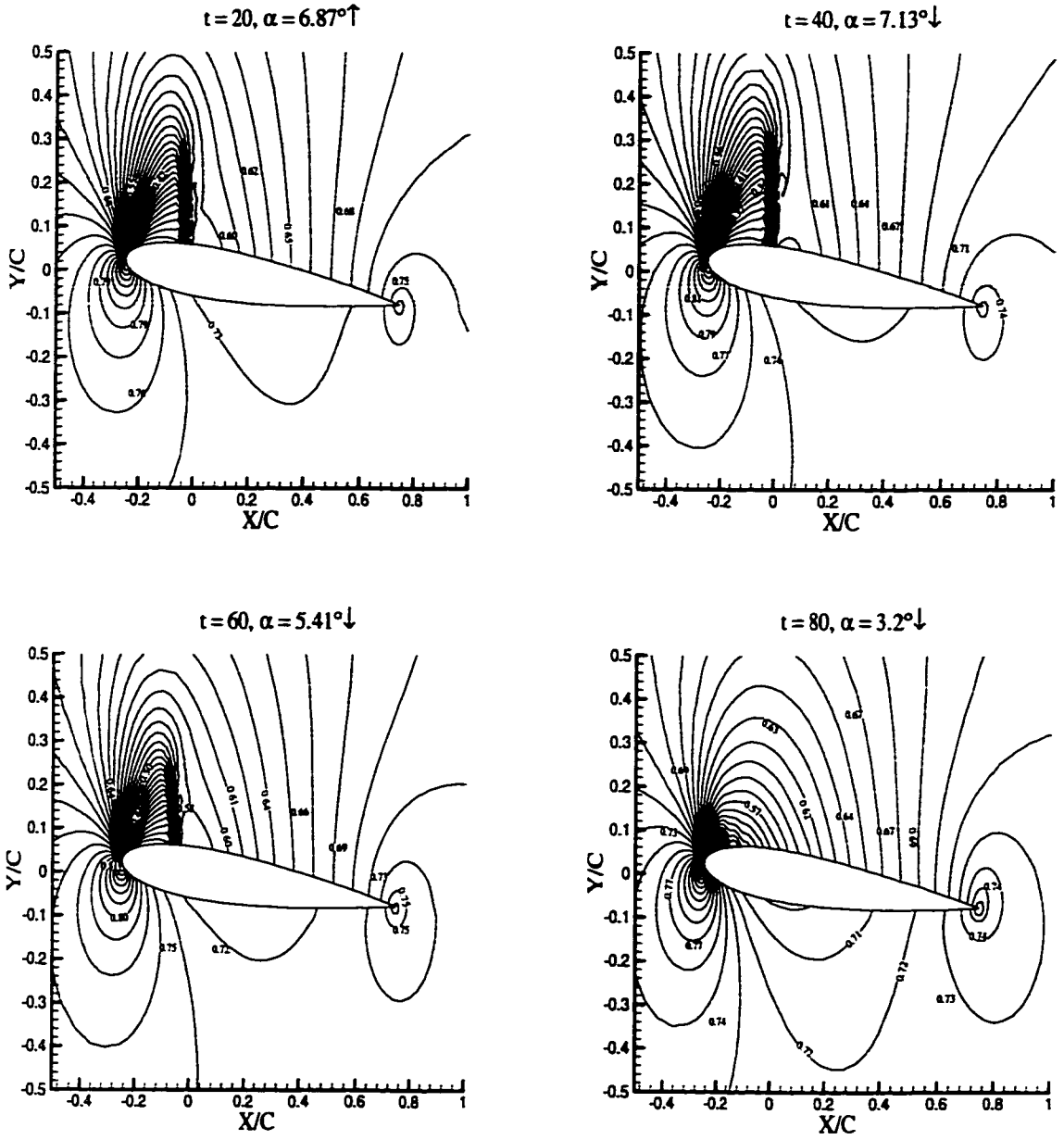


Figure 4.23 Instantaneous pressure contours with acceleration of multigrid method.

4.3 Pitching Wing with Moderate Reduced Frequency

4.3.1 Explicit scheme solution

The unsteady flow around a sinusoidally pitching rectangular wing was simulated by the explicit scheme described in Chapter 3. The rectangular wing has NACA 64A010 airfoil sections, and the aspect ratio is equal to 4. It was placed in a Mach 0.8 flow, and the angle of attack varied periodically according to the following expression:

$$\alpha(t) = -\alpha_0 \sin(M_\infty kt) \tag{4.3}$$

where α_0 is the amplitude, and the reduced frequency , k , is equal to 0.27.

The computational domain consisted of 12 chords in the normal and chordwise directions and 4 chords in the spanwise direction, respectively. The whole domain was tessellated into 40,533 tetrahedral cells and 7,775 nodes, of which 3,620 triangles and 1,048 nodes were placed on boundary surfaces. Figure 4.24 shows the inviscid-type unstructured mesh on the wing surface and the symmetry plane, while Mach contours and pressure contours on these surfaces and plane in the static zero incidence flow are shown in figures 4.25 and 4.26, respectively. The pressure coefficients are compared to available experimental data [51] at 50%, 77% and 94% semispan locations in figure 4.27 [3]. From these figures, the computational results matched with the experimental data very well.

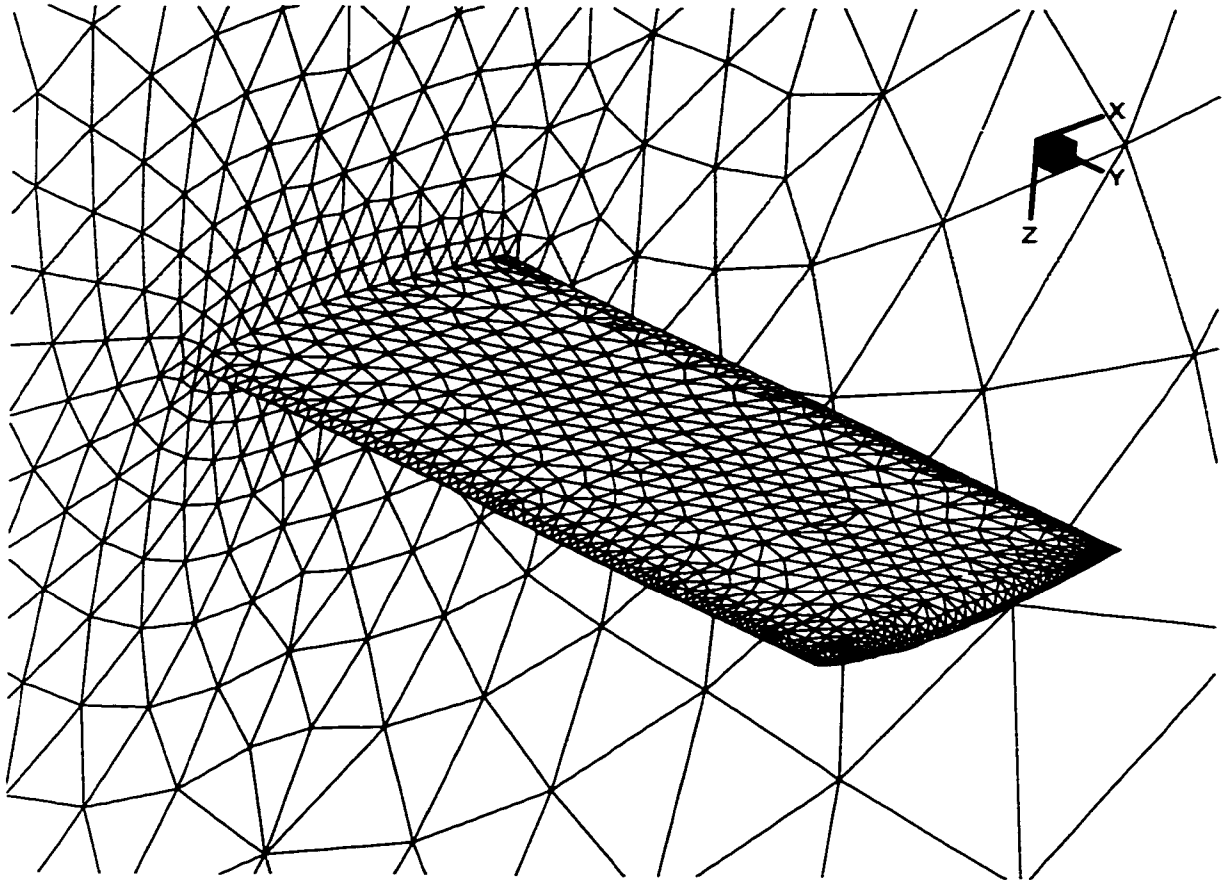


Figure 4.24 Unstructured mesh on surfaces.

Figure 4.25 Mach contours in static flow: $M_\infty = 0.8$.

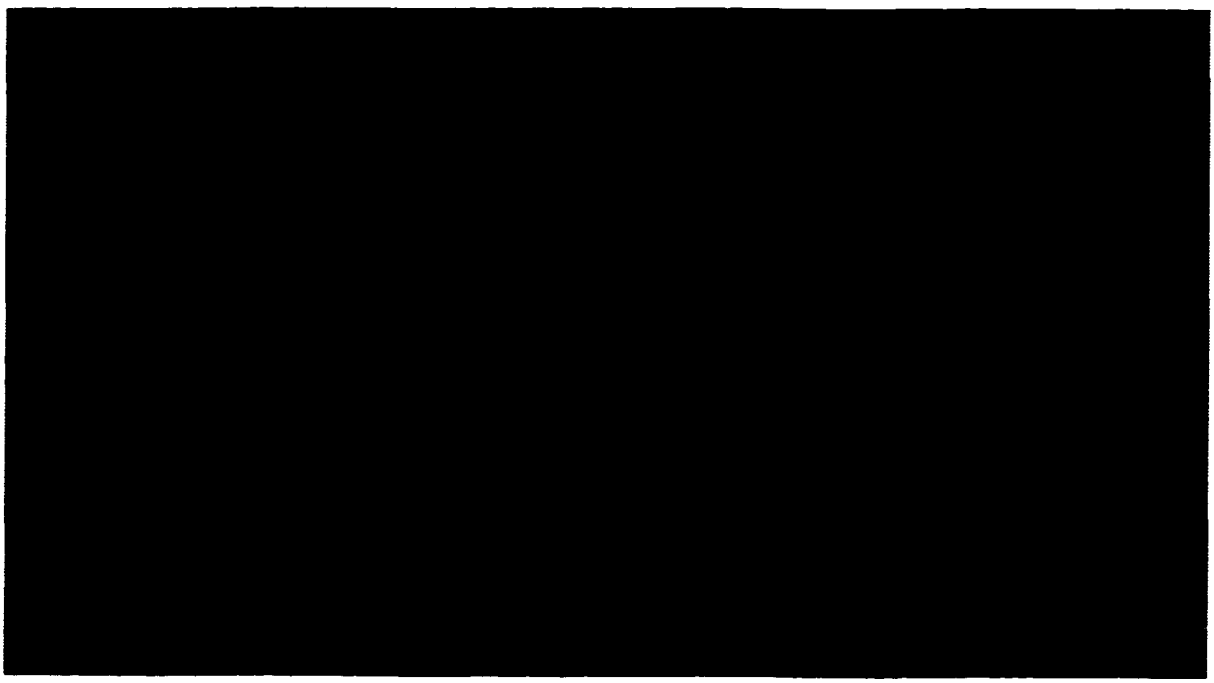


Fig 4.26 Pressure contours in static flow: $M_\infty = 0.8$.

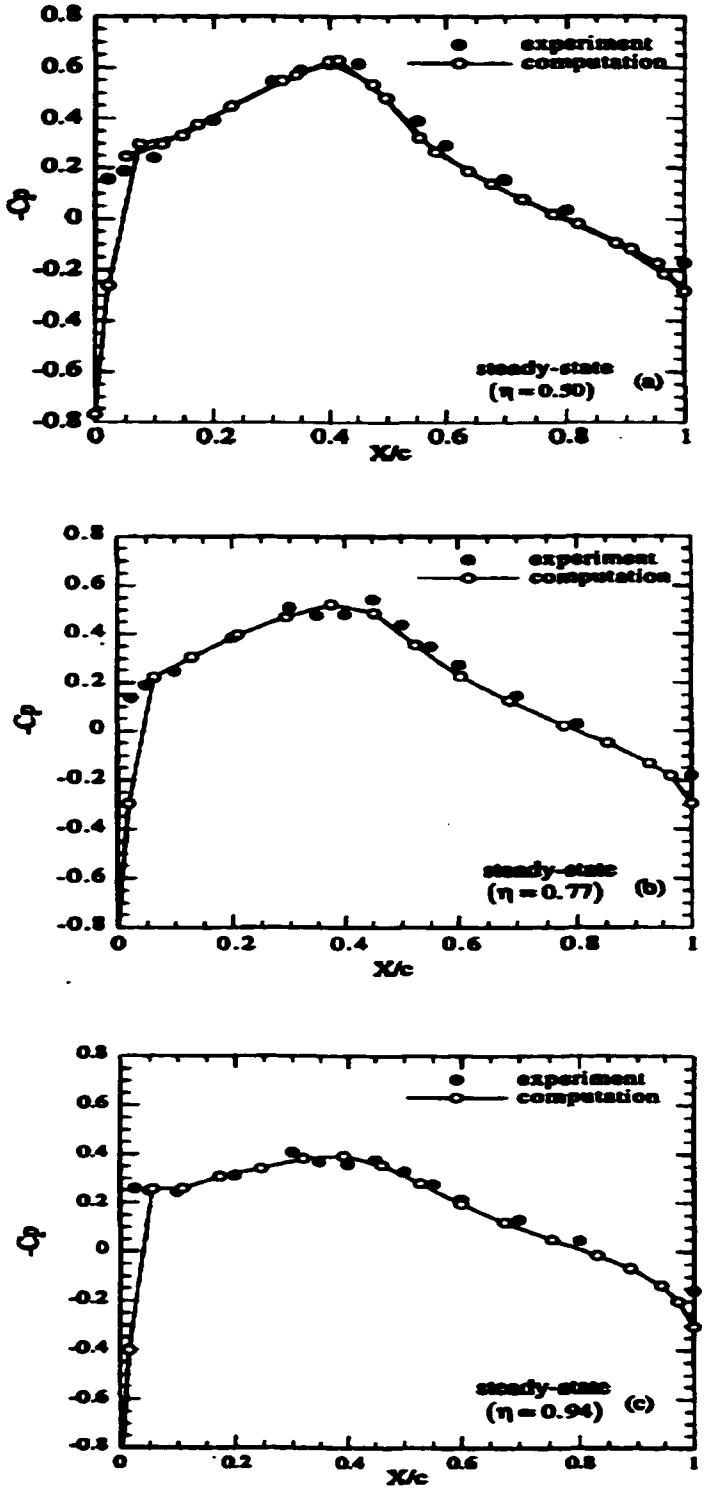


Figure 4.27 Comparison of static pressure coefficients at three semispan locations [3].

After obtaining the static solution, an adaptive window, enclosing 36,781 tetrehadra and 7,135 nodes, was built up. The time-accurate computation for unsteady flows was carried out with the amplitude, α_0, and the global time steps equal to 1.0° and 0.0018, respectively. This time step resulted in 17,600 iterations per period, which took approximately 113,000 CPUs on a SGI R10000 workstation.

For a harmonic motion, the complex number notation is a way to describe the variation of quantities, assuming that the response of fluid particles to a sinusoidal external excitation is also sinusoidal. Thence, the real part of pressure perturbation is in phase with the moving body while the imaginary counterpart is out of phase. The Fourier decomposition of the computed time-dependent pressure coefficient can be written as:

$$\text{Real}\{\ \overline{C}_p\ \} = \frac{2}{\alpha_1(t_2-t_1)} \cdot \int_{t_1}^{t_2} [C_p(\tau) \cdot \{\sin(M_\infty k\tau\}] d\tau \qquad (4.4)$$

$$\text{Imaginary}\{\ \overline{C}_p\ \} = \frac{2}{\alpha_1(t_2-t_1)} \cdot \int_{t_1}^{t_2} [C_p(\tau) \cdot \{\cos(M_\infty k\tau\}] d\tau \qquad (4.5)$$

where α_1 represents the pitching magnitude and $t_2 - t_1$ is the cycle time.

Figures 4.28-4.30 present the comparisons of the computed real and imaginary pressure coefficients, defined in the equation (4.4) and (4.35), with the experimental data [48] at 50%, 77%, and 94% semispan locations. Considering the fact that the used mesh was rather coarse and all the viscous terms were omitted, the results matched well.

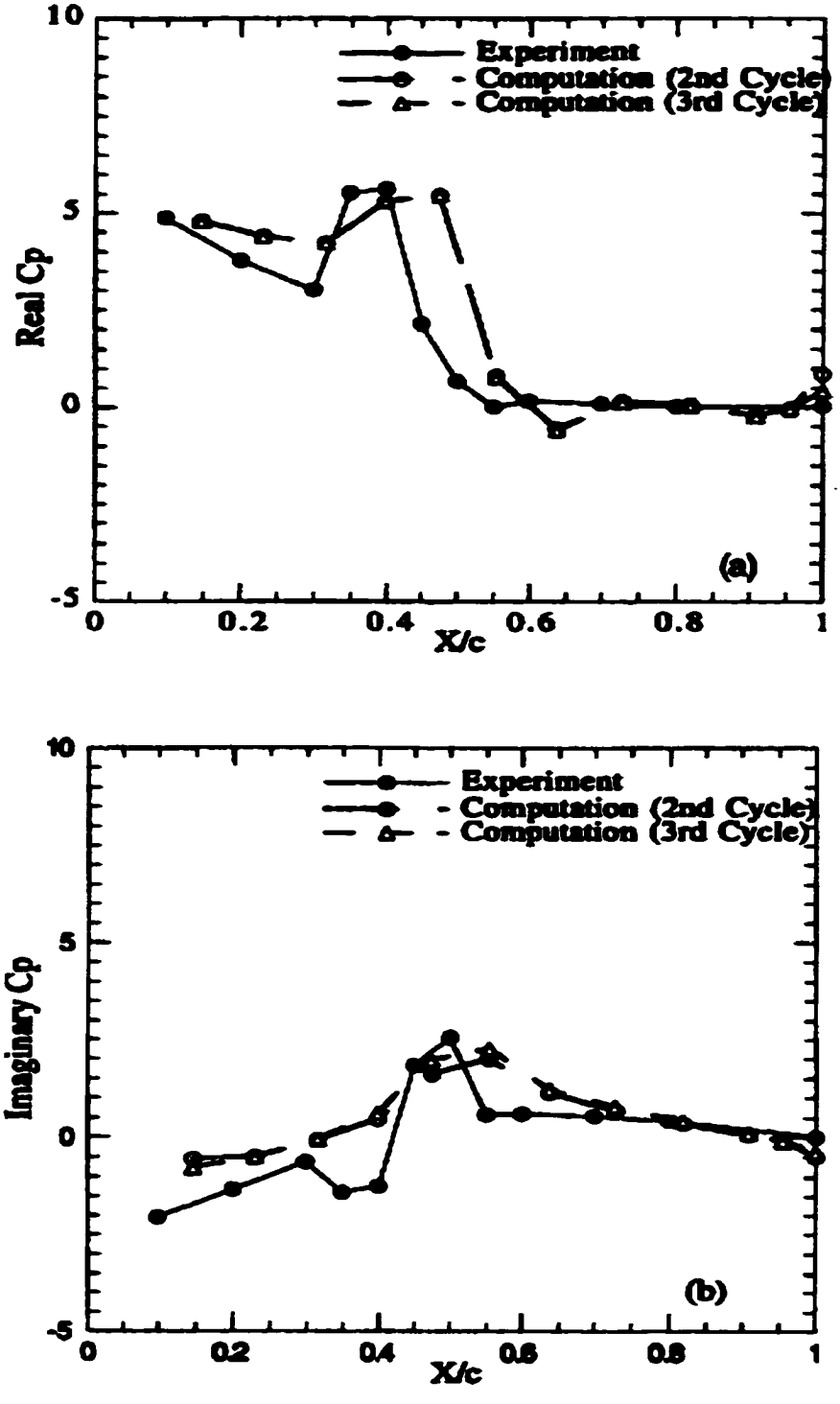


Figure 4.28 Comparison of pressure coefficient at 50% semispan.

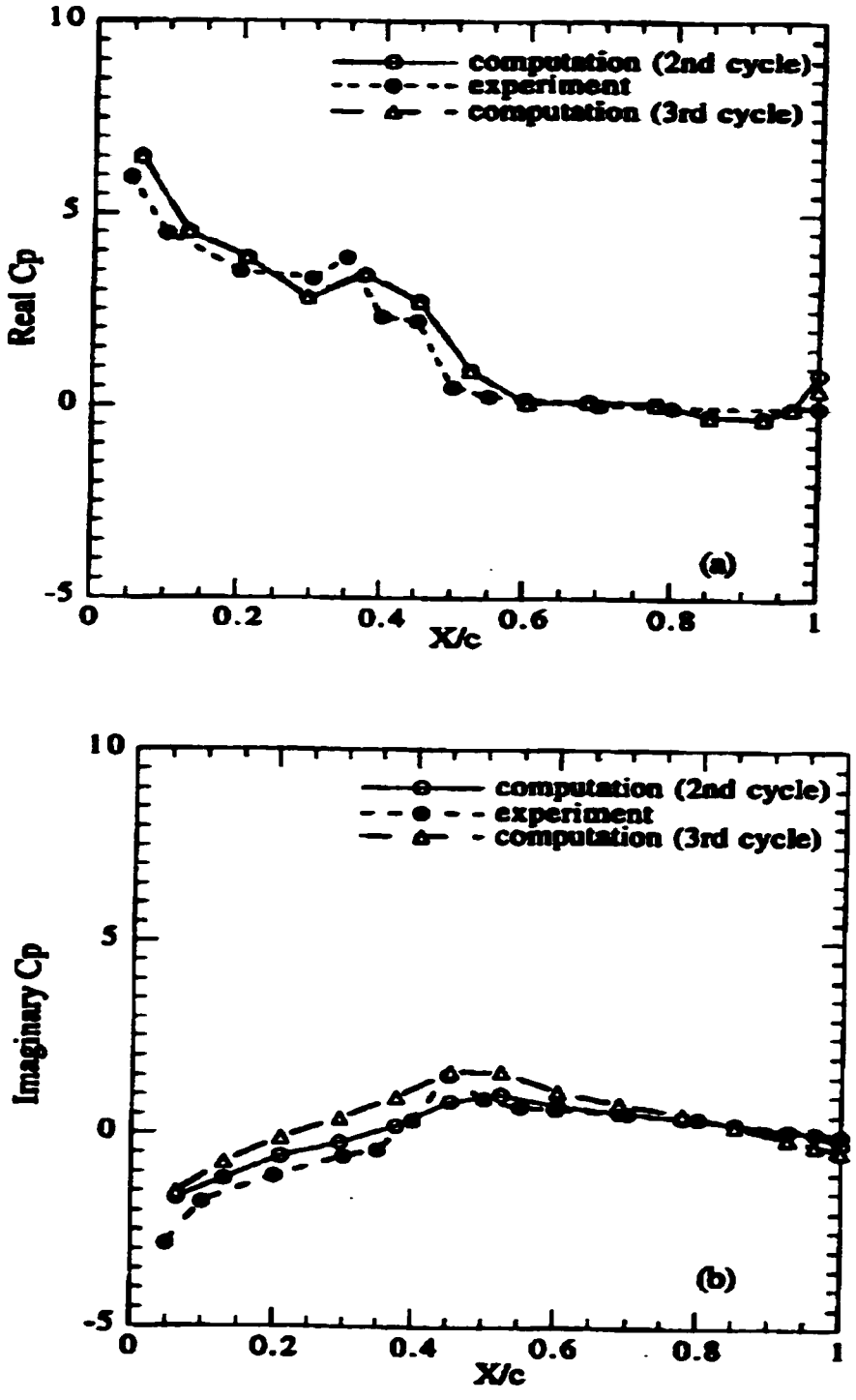


Figure 4.29 Comparison of pressure coefficient at 77% semispan.

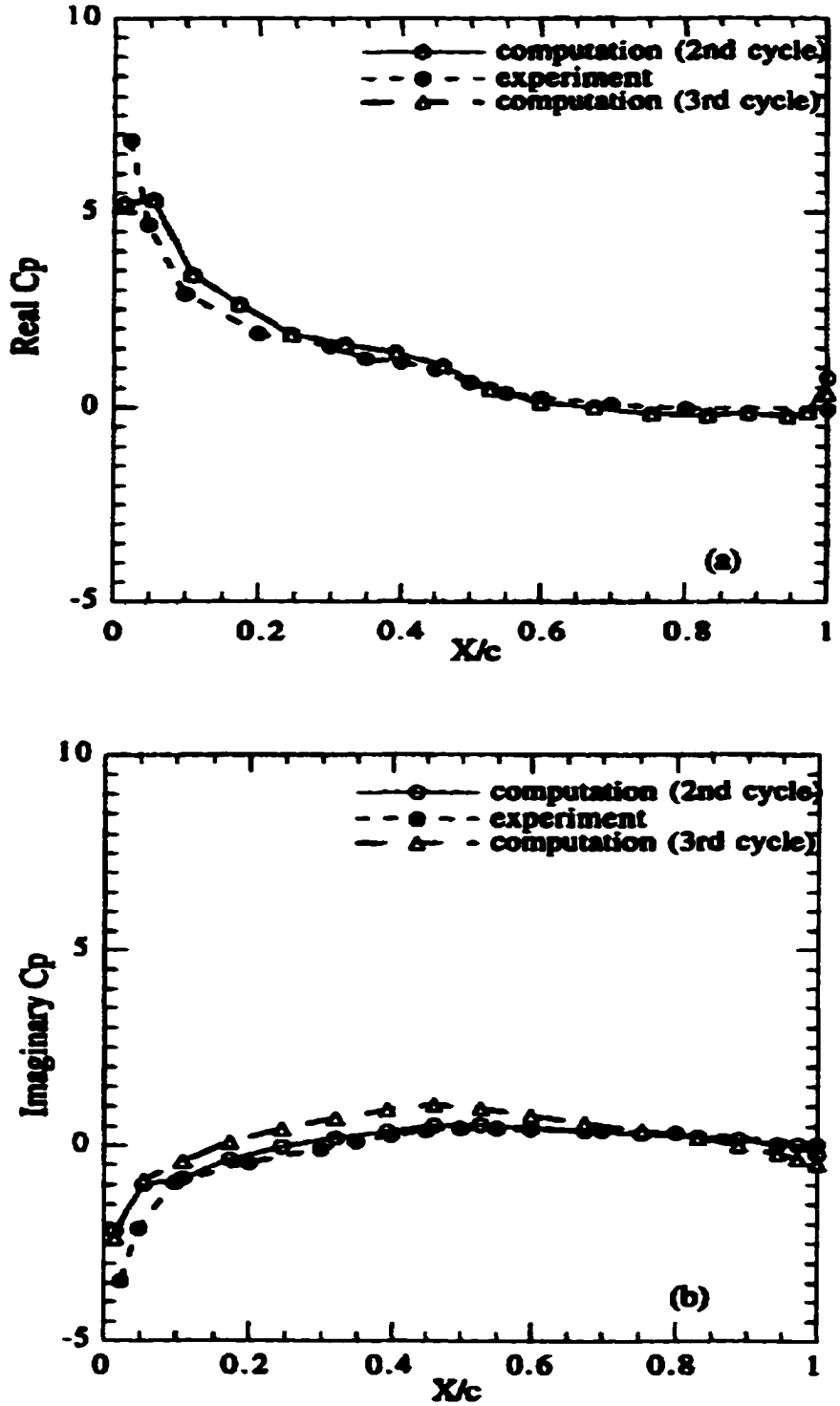
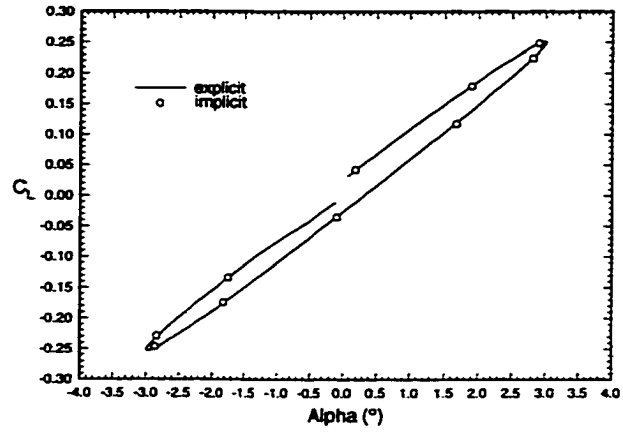
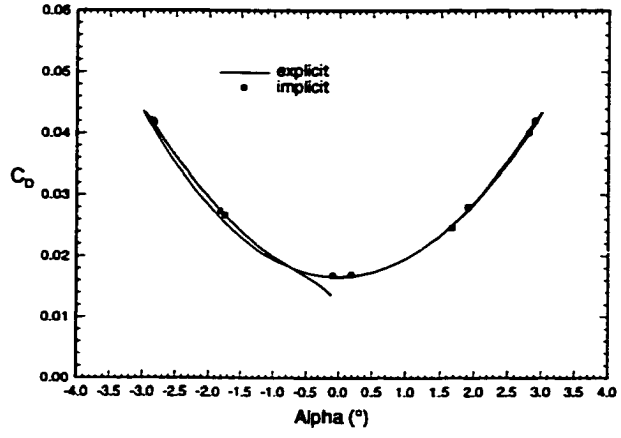


Figure 4.30 Comparison of pressure coefficient at 94% semispan.

4.3.2 Implicit scheme solution

In the present research, the dual-time stepping scheme, which was described in Chapter 3, has been used to obtain the unsteady aerodynamic loading on the wing. The amplitude of the pitching motion was increased from 1.0° to 3.0°. Normally, larger amplitude means larger perturbations to the flow field. Hence, this case is more difficult for the subiterations to converge and better to show the advantage of the dual-time stepping scheme over the conventional explicit scheme. The time step for time integration and for the dynamic mesh module was 1600 and 100 times that of the explicit scheme, respectively. The comparisons of aerodynamic forces in the first, the second, and the third (limit) cycle are depicted in figures 4.31-4.33. It was concluded that the results from the explicit scheme and the implicit scheme matched very well, and the multigrid acceleration was not necessary. This was credited to the fast convergence of subiterations, whose history is shown in figures 4.34-4.36. CPUs needed per cycle decreased from 113,000 seconds in the explicit scheme to 11,612 seconds in the implicit scheme on a SGI R10000 workstation. From this computation, it is concluded that the convergence characteristic of subiterations has a close relationship with the features of a flow field. The more nonlinear phenomena the flow field has, the lower convergence rate the subiterations have.





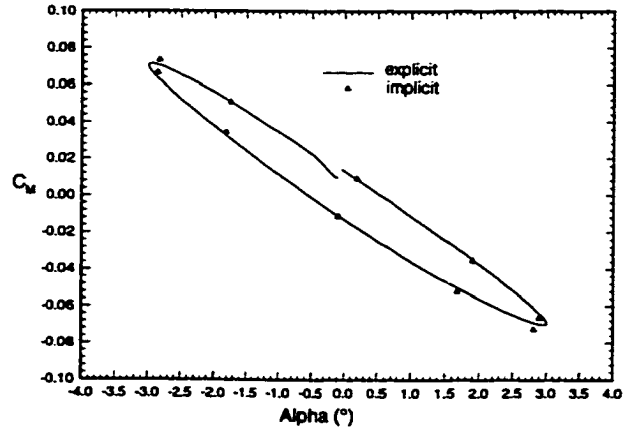


Figure 4.31 Comparison of aerodynamic forces in first cycle: $\Delta t_i = 1600\Delta t_e$.

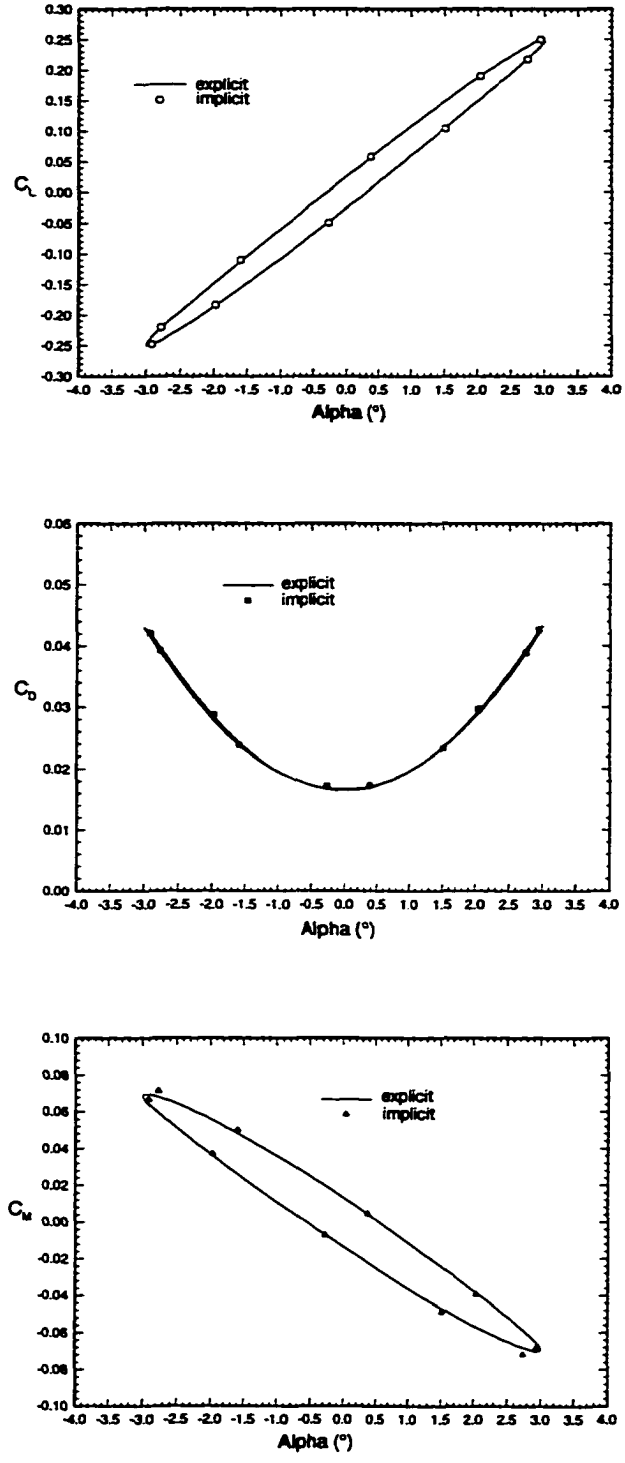
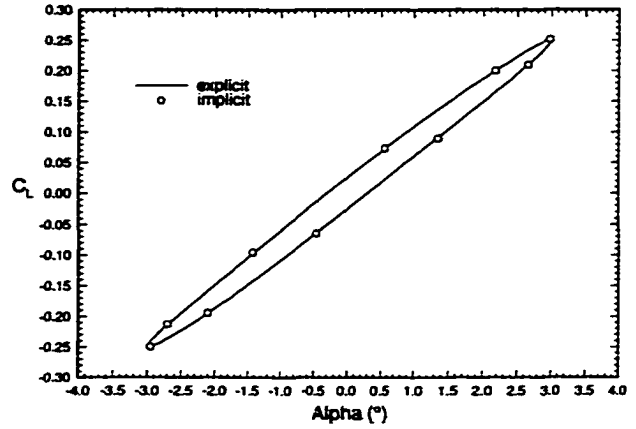
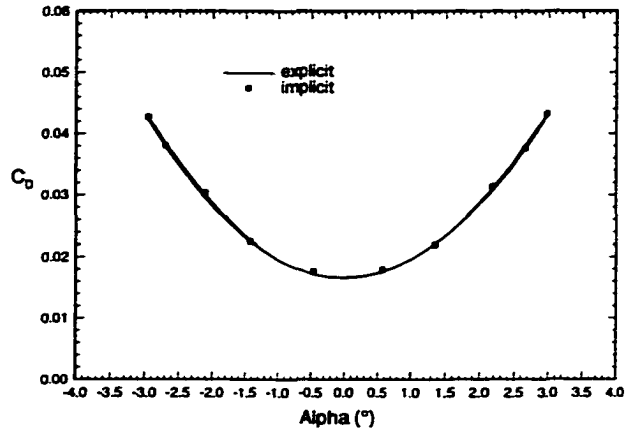


Figure 4.32 Comparison of aerodynamic forces in second cycle: $\Delta t_i = 1600\Delta t_e$.





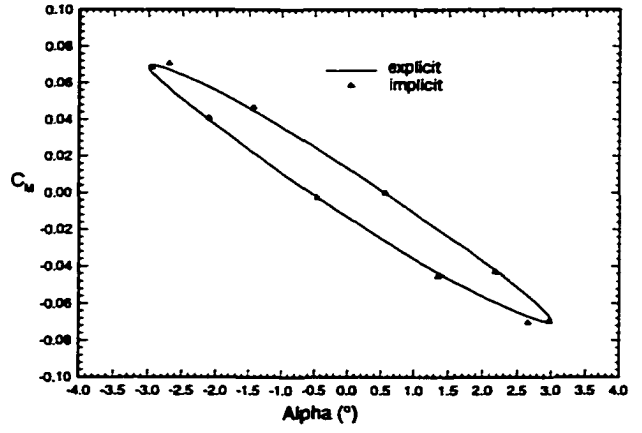


Figure 4.33 Comparison of aerodynamic forces in third cycle: $\Delta t_i = 1600\Delta t_e$.

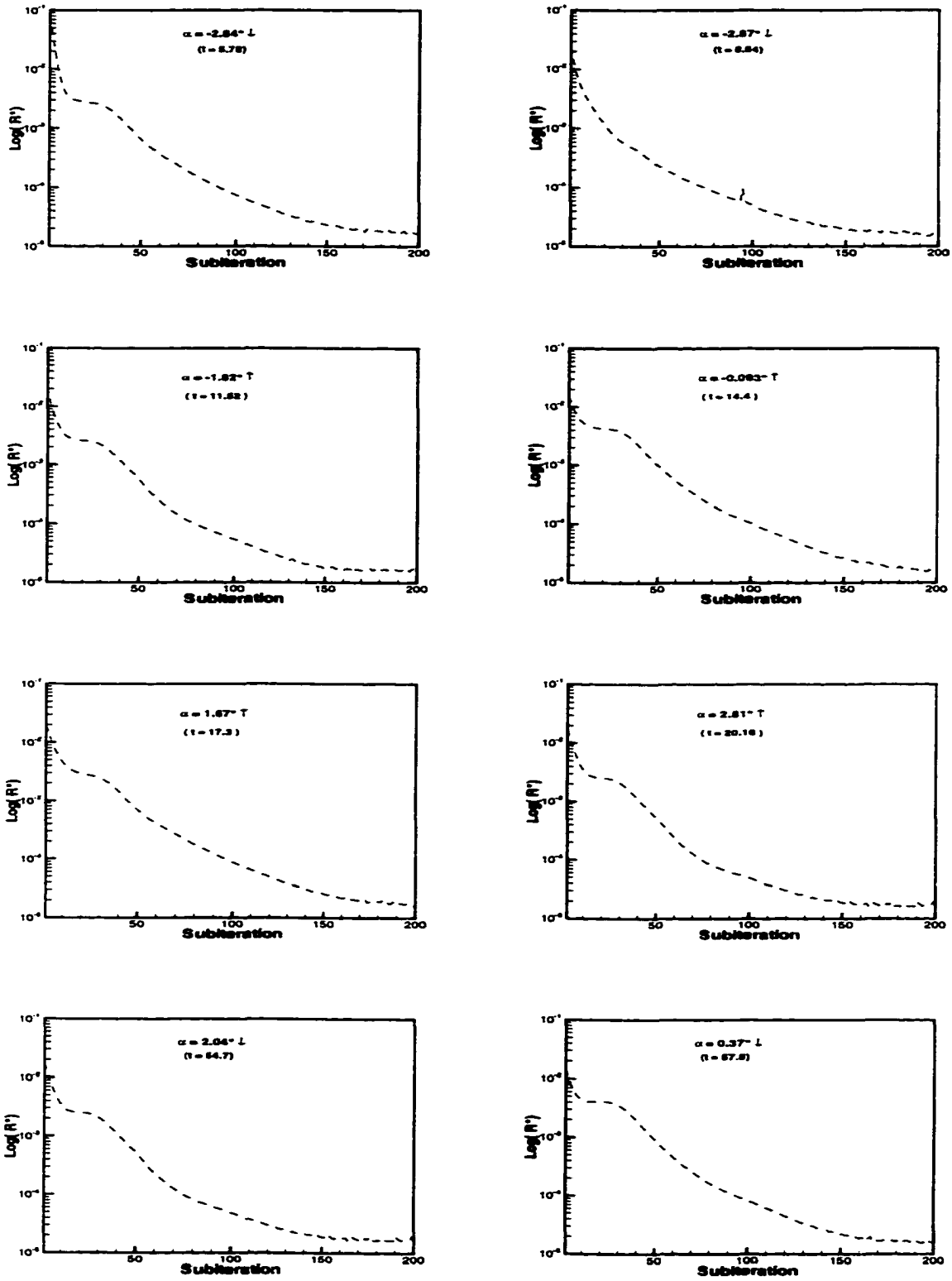


Figure 4.34 History of instantaneous subiterations in first cycle: $\Delta t_i = 1600 \Delta t_e$.

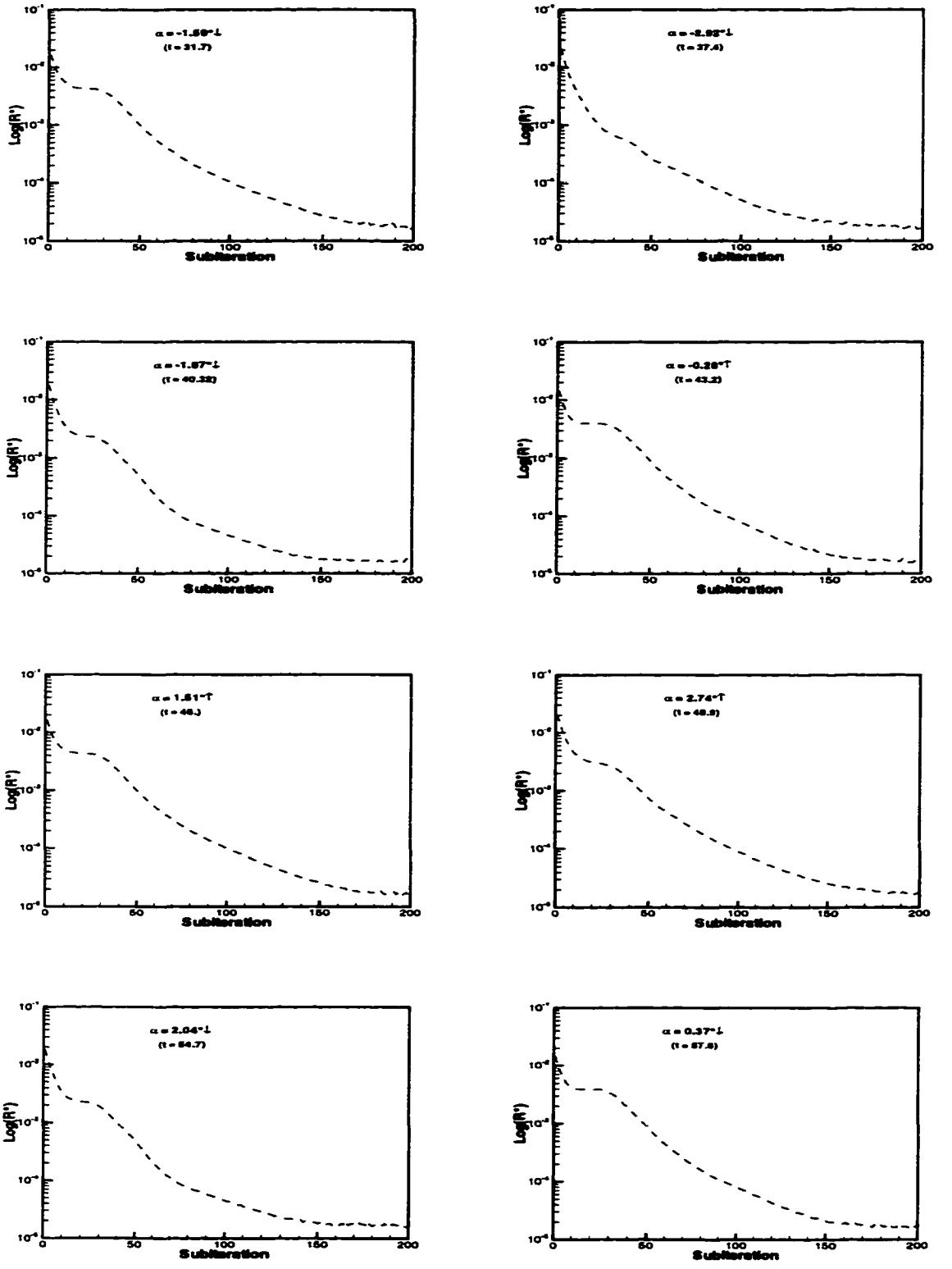


Figure 4.35 History of instantaneous subiterations in second cycle: $\Delta t_i = 1600\Delta t_e$.

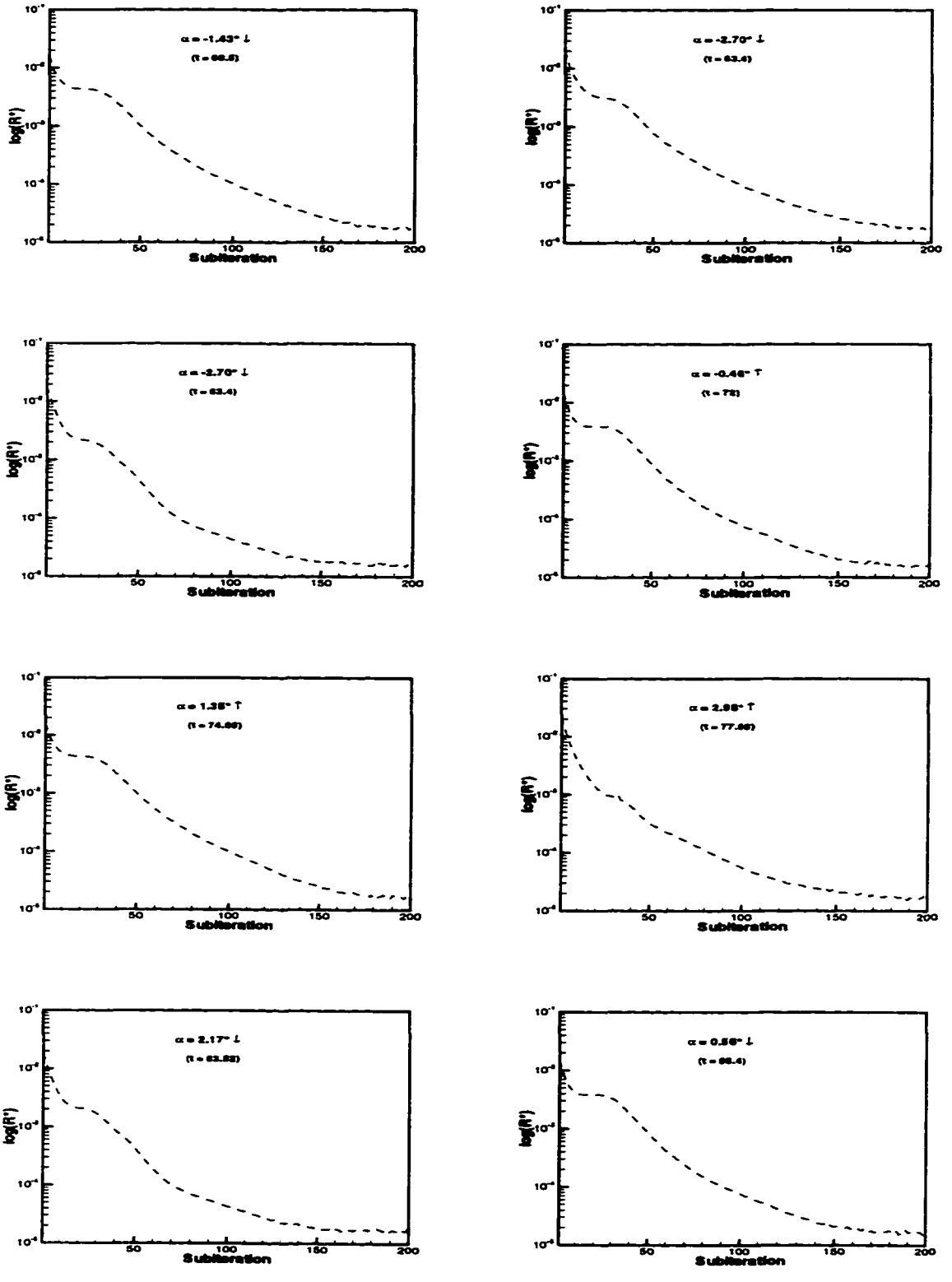


Figure 4.36 History of instantaneous subiterations in third cycle: $\Delta t_i = 1600 \Delta t_e$.

In this chapter, three implicit computations were performed on a pitching airfoil and one was on a pitching rectangular wing, using the methods described in Chapter 3. From these computations, the following conclusions are made. The dual-time stepping scheme is an efficient approach to solve the problems with a large time domain. Almost one order of CPU time savings was achieved for both the pitching airfoil and the pitching rectangular wing. The multigrid method is a powerful approach to improve the convergence of subiterations when the flow field has apparent nonlinear phenomena and the size of time steps is large.

The next chapter will present the results of simulating the staging of a hypersonic vehicle, using the upwind finite volume method on the longitudinal symmetry plane and the central differencing finite element method in full three-dimensional computations.

CHAPTER 5

DYNAMIC SIMULATION OF HYPERSONIC VEHICLE STAGING

5.1 Introduction

5.1.1 Background of Hyper-X program

In recent years, the interest in building a hypersonic vehicle has increased. Several programs, such as X-33, X-34 and X-38, are ongoing at NASA and Industry. They are integral parts of the reusable launch vehicle technology programs. The primary objective of these projects is to develop the core technologies needed to build and operate the reusable space transportation systems, which will significantly lower the cost of access to space.

Airbreathing propulsion provides substantial advantages for a hypersonic flight, as shown in figure 5.1. The mission effectiveness could be improved in ramjet/scramjet (supersonic combustion ramjet) engines by the way of reducing the on-board propellant load in favor of payload and by increasing the operational flexibility.

In order to enhance the hypersonic airbreathing propulsion technology for applications in the long term, NASA has initiated the Hyper-X program. The main goal of this program is to demonstrate and validate the technology, the experimental techniques, the computational methods, and the tools for the design and the performance predictions of a hypersonic vehicle with an airframe-integrated dual-mode scramjet

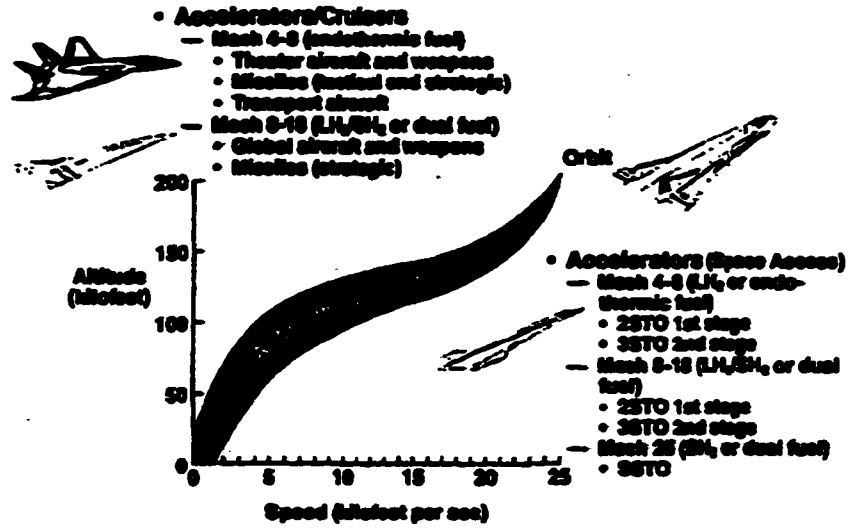


Figure 5.1 Potential hypersonic airbreathing vehicle applications [55].

propulsion system. To accomplish this goal, several in-flight measurements have been scheduled in order to validate the computational fluid dynamics codes and engineering tools used to design the vehicle [46].

Three autonomous research flights with speeds up to Mach 10 have been scheduled between 2000 and 2002 to demonstrate and validate the readiness of scramjet technology for the next century. Each of the 12-foot-long, 5-foot-wingspan hypersonic research aircraft (the operational aircraft is conceptualized to be 200 ft), shown in figure 5.2, will have a single airframe integrated scramjet. The layout of Hyper-X equipment is shown in figure 5.3.

Mach 7 and 10 flights will be tested with the dynamic pressure equal to 1,000 lb/ft^2. A modified Pegasus booster, Hyper-X Launch Vehicle (HXLV), will boost the

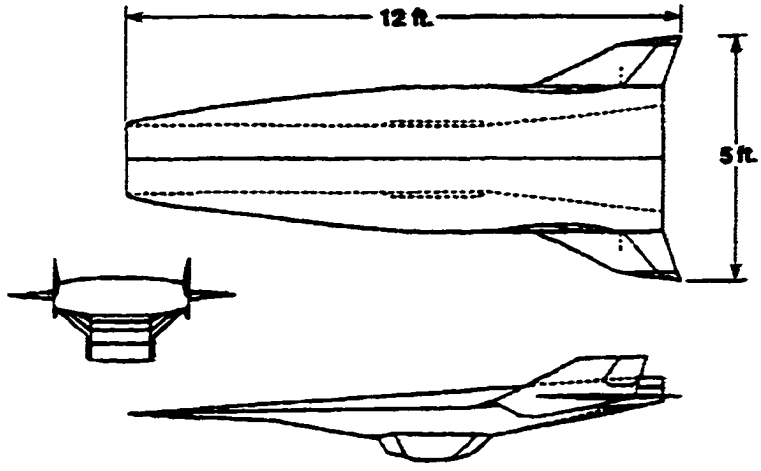


Figure 5.2 Hyper-X research vehicle configuration [55].

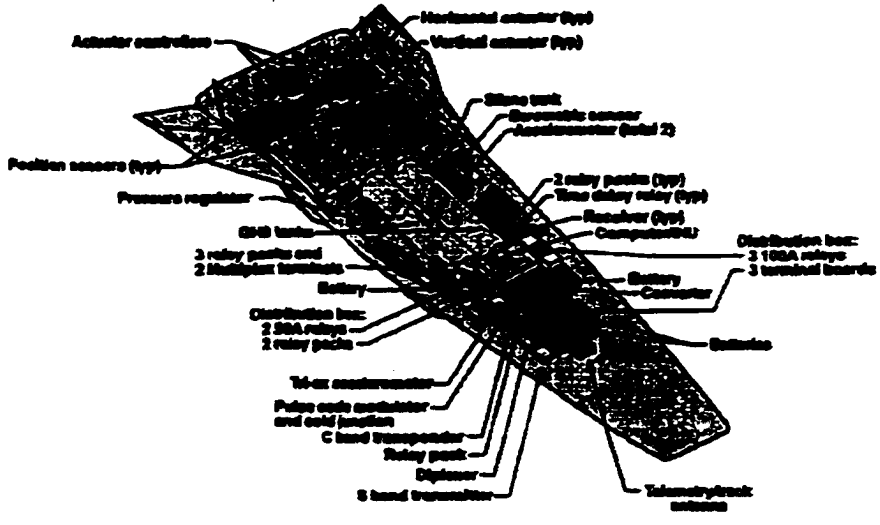


Figure 5.3 Layout of Hyper-X equipment [55].

research vehicle to the flight test conditions. The flight trajectory will start with a B-52 aircraft, which will carry the HXLV-mounted research vehicle to about 40,000 ft for Mach 10 flight or less for lower speed flights. Figure 5.4 shows that the research vehicle

is mounted on the B-52. After separating from the B-52, the HXLV will climb to about 100,000 ft for Mach 7 flight, and 110,000 ft for Mach 10 flight.

Figure 5.4 Hyper-X research vehicle on B-52 [55].

The research vehicle will be mounted on an adapter attached to the HXLV, and the planned stage separation will begin with the ignition of the explodable bolts fastening the vehicle to the adapter [56]. Then, the forward part of the adapter which links with the nozzle part of the research vehicle will swing down about a hinge to let the research vehicle free. The model of drop jaw adapter is shown in figure 5.5.

Once separated from the booster, the research vehicle will establish a stable unpowered flight. However, whether this stage separation process will end with the non-pilot Hyper-X vehicle having a favorable initial attitude for its flight is not certain at this time. Impingement of time-dependent shocks from the booster/adapter stack on the

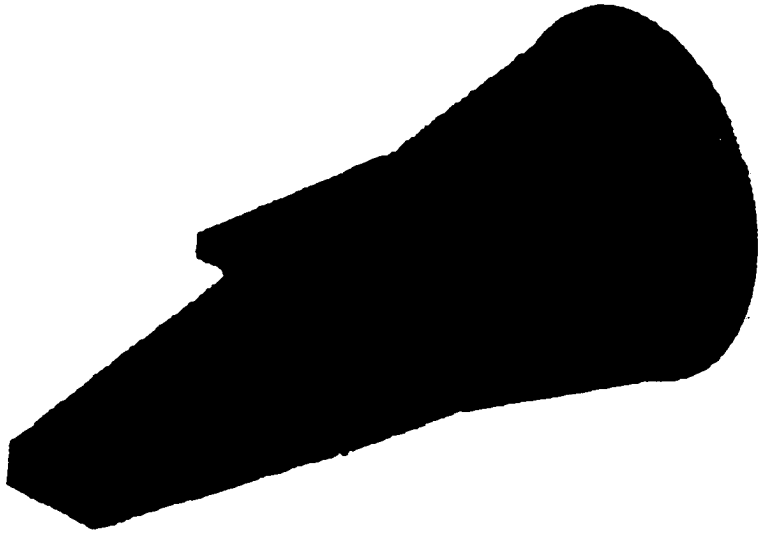


Figure 5.5 Model of drop jaw adapter [55].

research vehicle will inevitably influence the aft portion of the research vehicle. A strong aerodynamic interference pressurizing the nozzle area, therefore imparting a nose down moment on the research vehicle, is expected due to the shocks generated by the adapter jaw and the booster. The control system on the research vehicle might not be able to handle it.

Tests have been conducted at both Langley Research Center and Arnold Engineering Development Center to qualify the interference effects of the adapter/booster with its dropping jaw on the research vehicle. Tests have also been conducted at Langley Research center to qualify the sting interference effects on the data from both sets of tests. The test data have been corrected for these sting interference effects. The photograph of the model in the wind tunnel is shown in figure 5.6.

Figure 5.6 Photograph of Hyper-X testing model [56].

In this respect, physics-based computational modeling can offer valuable augmentation to the tests. The objective of the present research is to perform an adequate prediction of hypersonic vehicle staging until the aerodynamic interference is negligible, which should offer a close scrutiny of the resulting loads on the research vehicle.

After the development in the past twenty years, CFD has been gradually applied in modeling the hypersonic flows, and the industry slowly accepts it as a viable design tool alternative to the wind tunnel test. The simulations of 3-D flow fields around the Hyper-X research vehicle in free flight have been reported [46, 55]. Figure 5.7 shows computed Mach contours at three positions along the streamline direction.

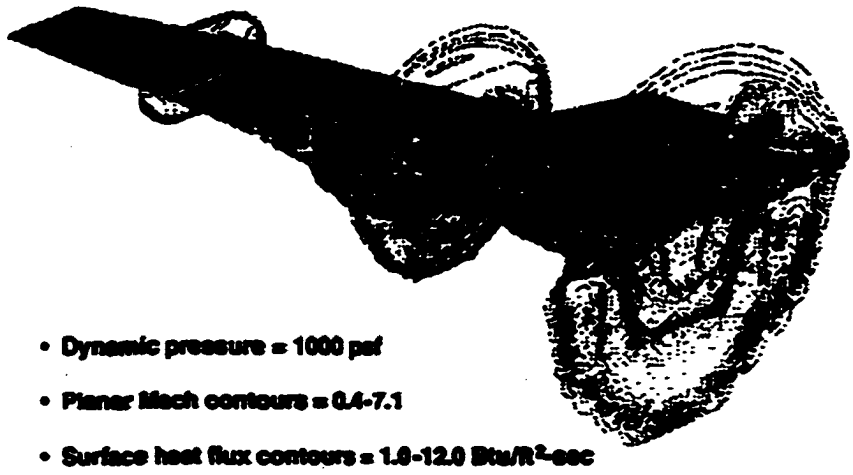


Figure 5.7 CFD image of Hyper-X at Mach 7 test condition [46].

It should be stressed here that in the present proof-concept investigation, the geometry used to construct meshes is not exactly the same as that in the flight test because of non-technical reasons, and all of the data are available from published materials. The terms associated with viscosity and heat transfer are omitted because of the size of the problem.

5.1.2 Computational methods

The effects of terms associated with viscosity and conductive heat transfer are discussed here briefly. In hypersonic flights, a very thin boundary layer will exist in the regions close to bodies where the gradients of velocity and temperature are very high. The interaction of the boundary layer and shocks will induce unexpected separations. The high temperature gradients will influence the velocity profiles through the variations of density. Normally, in hypersonic flows, the viscous terms have less influence on the

pressure distribution in smooth regions, as compared to compressible flows, because the boundary layer in hypersonic flows is much thinner. However, during the staging, velocities in the nozzle are slow with respect to free stream, due to strong shock interactions. Therefore, viscous terms play a relatively more important role in the staging, as compared to free flights. The flow at the corner of the engine outlet may be separated.

Hence, it is desirable to simulate the staging of a hypersonic vehicle by solving time-dependent N-S equations. However, considering the size of the problem (a full three-dimensional simulation), the complexity of computational domain (three components in relative motion), and the insufficient maturity of high-order accurate computational methods on viscous type unstructured meshes, the effects of viscosity and conductive heat transfer have been ignored. Therefore, in the present research, the time-dependent Euler equations were solved.

Since the present dynamic mesh method (section 3.2.6) does not allow adding/deleting cells, the dynamic mesh module imposes limitations on the scale of the assigned relative motion. As a result, the time domain to be simulated is relatively small. In turn, the total number of time steps using an explicit scheme is still acceptable. For problems with a small time domain, explicit schemes are desirable because they have low memory requirement, and they are much more efficient per iteration, as compared to implicit schemes with multigrid-accelerated subiterations.

On the other hand, unlike the problems of pitching airfoils or wings, the history of aerodynamic loading in staging is important. An implicit scheme only offers the aerodynamic forces at corresponding instants because of the large time step used.

Therefore, the important information about the history of loading is missed in an implicit scheme.

Usually, the speed of sound is taken as the reference of velocity in transonic flows. But, in hypersonic flows, using the velocity of free stream as the reference will increase the size of allowable time step in an explicit scheme for time-dependent problems by Mach number times. Therefore, in hypersonic flows, taking the velocity of free stream as the reference is also an efficient approach to increase the size of an allowable time step in an explicit scheme.

For the three above reasons, the Runge-Kutta explicit scheme was used to perform the time integration in the simulation of hypersonic vehicle staging, not the dual-time stepping implicit scheme.

5.2 Two Dimensional Computations

To obtain some experience, two-dimensional computations were performed on the longitudinal symmetry plane, which is present in this section. The USM2D code was taken as the baseline solver. Although significant spanwise effects will exist in reality, they are ignored here. They will, however, be considered in 3-D computations present in the next section. Meanwhile, a tail was imposed on the upper surface of the research vehicle.

5.2.1 Mesh generation

At the beginning of the stage separation, the three components, research vehicle, adapter "jaw" and Pegasus booster, are close to each other. For this kind of complicated multi-body configuration, the unstructured mesh method is desirable to discretize the computational domain into triangles or tetrahedra. Because the baseline solver requires the connectivity of meshes to be unchanged, the present research assumed that before CFD simulations, the staging had already started, and there was a certain small distance between the research vehicle and the booster. The advancing front method was employed to generate the mesh, in which the ideal distribution of mesh nodes was achieved by placing point sources and line sources in the regions where a strong aerodynamic interference was expected. The final mesh consisted of 63,936 triangles, 32,355 nodes and 96,293 edges. Figure 5.8 shows the entire mesh, while a close up view is displayed in figure 5.9.

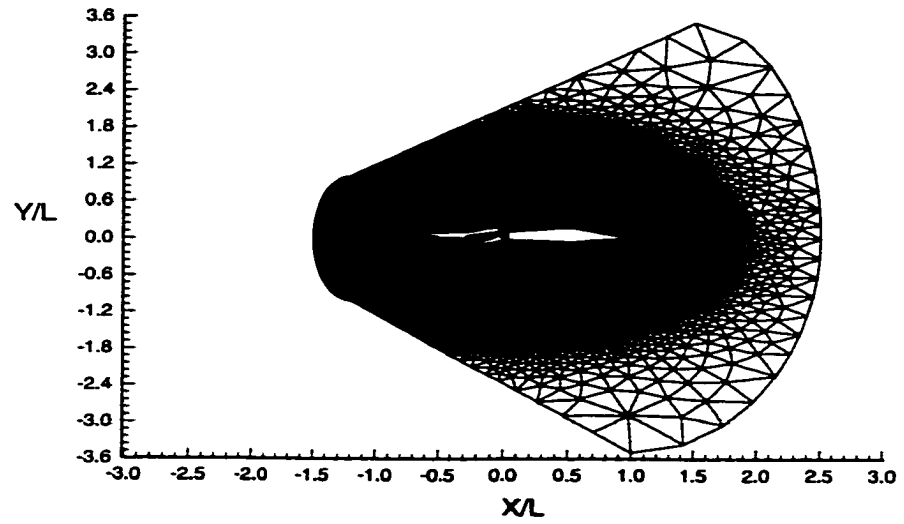


Figure 5.8 Overview of unstructured mesh.

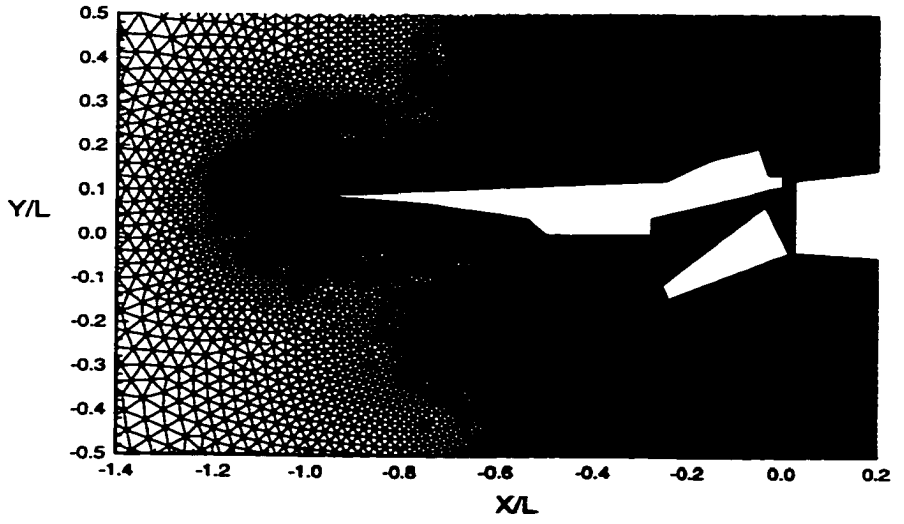


Figure 5.9 Close up view of unstructured mesh.

5.2.2 Simulation of two-degrees-of-freedom prescribed motion

In flight tests, the engine cowl door will not be opened until stable flight has been achieved after separation. Therefore, blocked engine cowl with no flow through was assumed. The static solutions in Mach 5 and Mach 10 flights were performed on a SGI R10000 workstation, based on the mesh shown in the previous subsection. Per iteration took about 7 seconds of CPU time in a static solution. Table 5.1 presents the computed static aerodynamic coefficients on the research vehicle in two flights.

Table 5.1 Static aerodynamic loading on the vehicle.
($\bar{X}_C = X_C / L = -0.5$, $\bar{Y}_C = Y_C / L = 0.$)

	Normal force	Axial force	Moment
$M_\infty = 5.$	0.4605	-0.0446	0.0425
$M_\infty = 10.$	0.3009	0.0306	0.0384

In contrast to the Mach 10 flight, an overall thrust in the axial direction on the research vehicle was predicted in the Mach 5 flight. With the existence of the booster and the jaw, a nozzle area with high pressure was formed in the base region of the research vehicle. As a result, flows imposed a force on the research vehicle, which had a negative component in the axial direction. The absolute value of the imposed force in the axial direction was larger than pressure drag in the Mach 5 flight; therefore, a thrust was resulted in. However, the overall axial force became positive again in the Mach 10 flight, because the pressure drag overwhelmed the thrust from aerodynamic interference. With the increase of the non-dimensional inertial force, the coefficients of normal force and

moment on the research vehicle in the Mach 10 flight were smaller than the values in the Mach 5 flight.

For dynamic computations, after ignoring the acceleration in the staging process, it was better to assume that the research vehicle was fixed, and the jaw and the booster had relative motions with respect to the research vehicle, since more mesh points were placed around the research vehicle. Then, a window was built up around the research vehicle, the jaw, and the booster, enclosing 11,356 triangles and 6,103 nodes. A two-degrees-of-freedom relative motion was prescribed on the jaw and the booster to simulate the staging process by assigning a translational motion at a constant speed and superposing a rotation on the jaw. The following are the equations of relative motion:

$$\Delta X = V_x \, t \tag{5.1}$$

$$\theta = \omega \, t \tag{5.2}$$

The values for dimensional translation and rotation were arbitrarily assigned as $V_X = 9.8 \; ft/s$ and $\omega = 7.1 \; rad/s$. Two dynamic simulations with non-dimensional time domain equal to 1.0 (12.2 ms in dimensional domain) were performed. Figure 5.10 shows the instantaneous positions of the jaw and the booster with respect to the research vehicle along the assumed staging path.

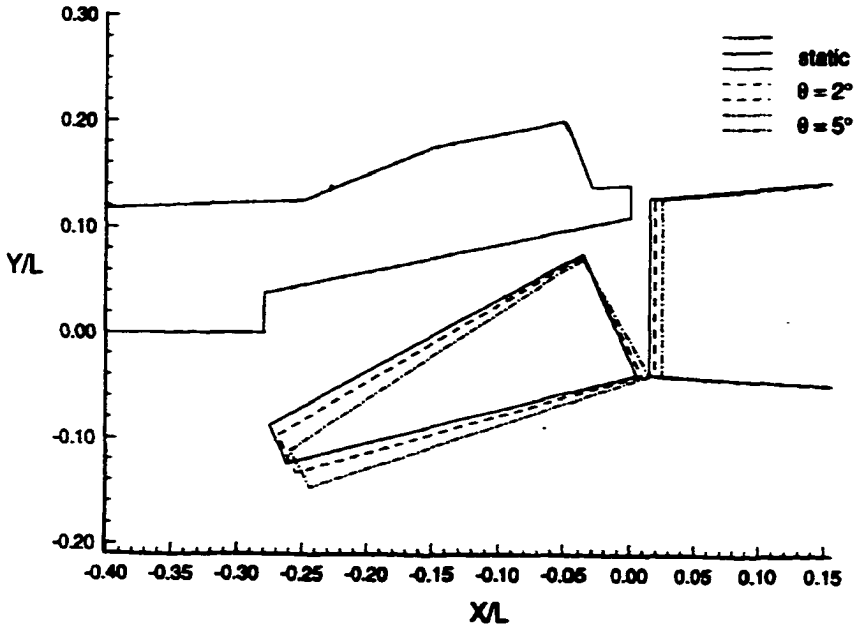


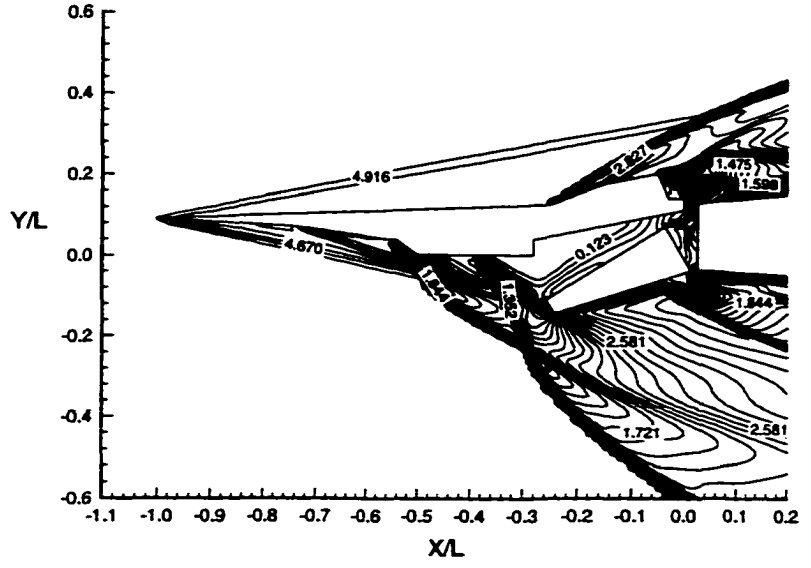
Figure 5.10 Instantaneous position in staging process.

After the numerical stability analysis, the non-dimensional time steps were taken as 2.5e-4 in the Mach 5 flight and 1.25e-4 in the Mach 10 flight, resulting in 4,000 and 8,000 iterations, respectively, each of which took about 8.6 CPUs. Therefore, the dynamic mesh module and the update of boundary conditions took about 20% of running CPUs in a static solution.

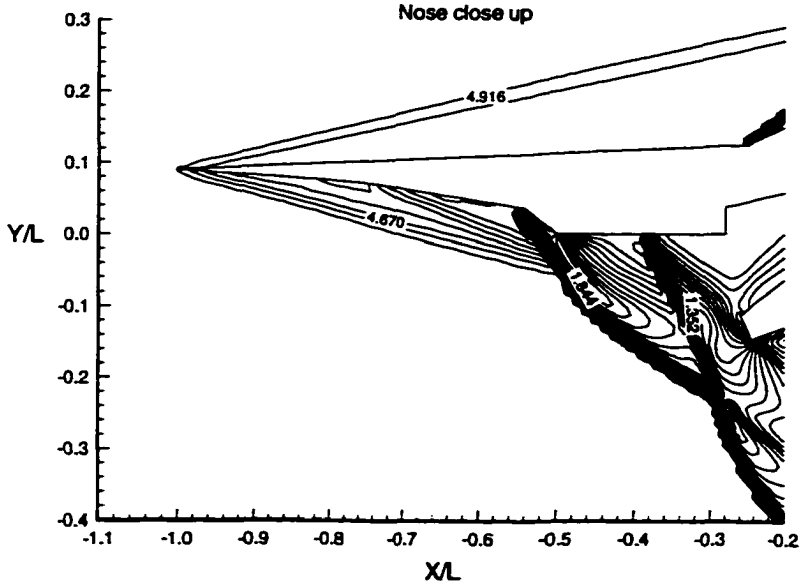
Figures 5.11 and 5.12 present the instantaneous off-surface Mach and pressure contours in a Mach 5 flight. Corresponding results in a Mach 10 flight are shown in figures 5.13 and 5.14. As observed in these figures, a very complex network of strong shocks, expansions and their interaction dominated the predicted flow-field. In the Mach 5 flight, the detached shocks from the engine-inlet were intersected with the detached shocks from the jaw, which extended to approximately the middle of the engine body. Behind the detached shocks, strong expansions existed. The expansion region of the

detached shocks from the jaw was divided by the detached shocks from the engine-inlet after their intersection. A subsonic region with high pressure was formed in the inboard between the research vehicle and the jaw, resulting in two jets, one bleeding into the base region of the research vehicle and the other to the lower surface of the booster nose. The latter jet incurred an additional shock on the lower surface of the jaw, and the former jet was compressed in the base region of the research vehicle.

In the Mach 10 flight, the shocks were stronger, and were blown closer to corresponding bodies by incoming flow. The detached shocks from the jaw were intersected with those from the booster, and two jets existed as well. It is concluded that the interference among components is highly dependent on the Mach number of free flight.

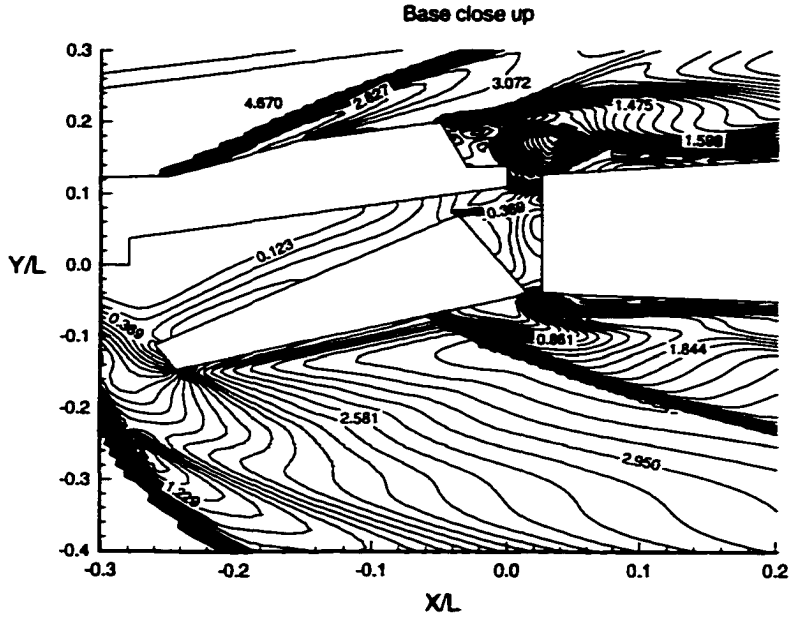


(a)



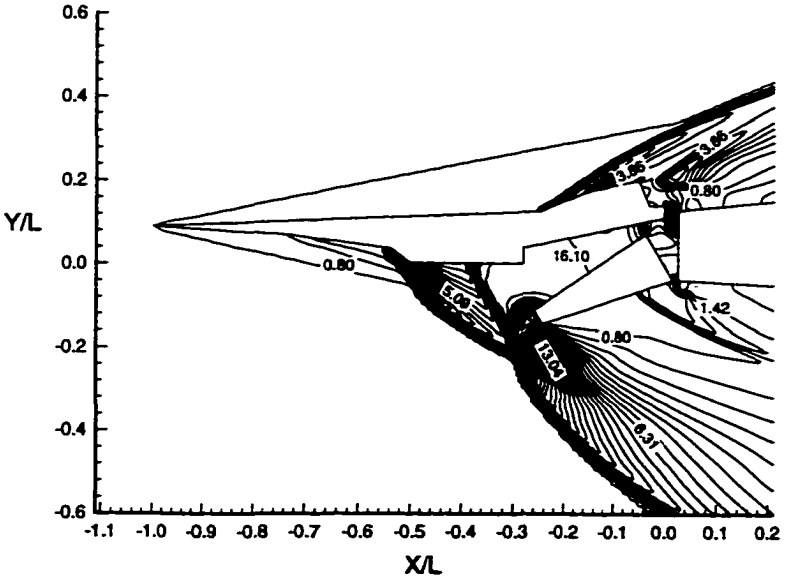
(b)

Figure 5.11 Staging during $M_\infty = 5.$ flight: instantaneous ($6 = 5^\circ$) Mach contours when vehicle separates from booster at 85,000 ft altitude ($q_\infty = 1{,}000 lbf / ft^2$).
(a) overview , (b) nose close up, (c) base close up.



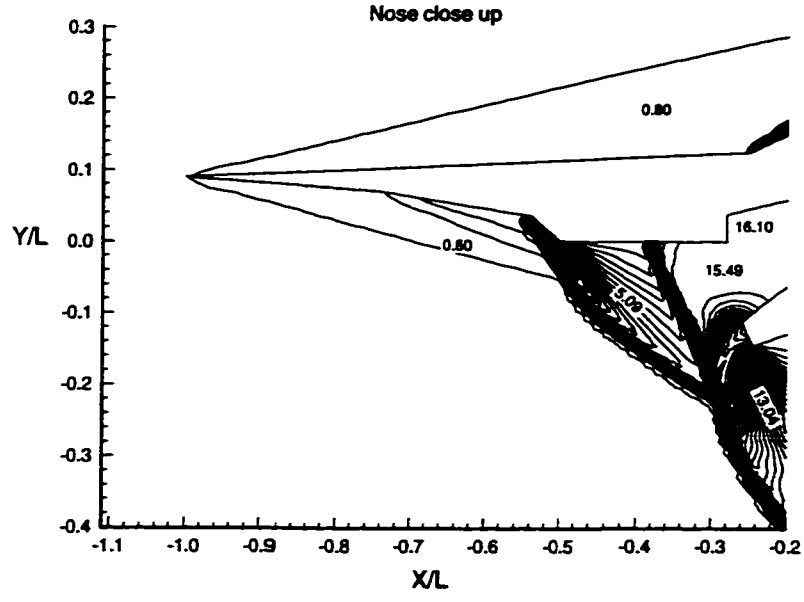
(c)

Figure 5.11 Concluded.

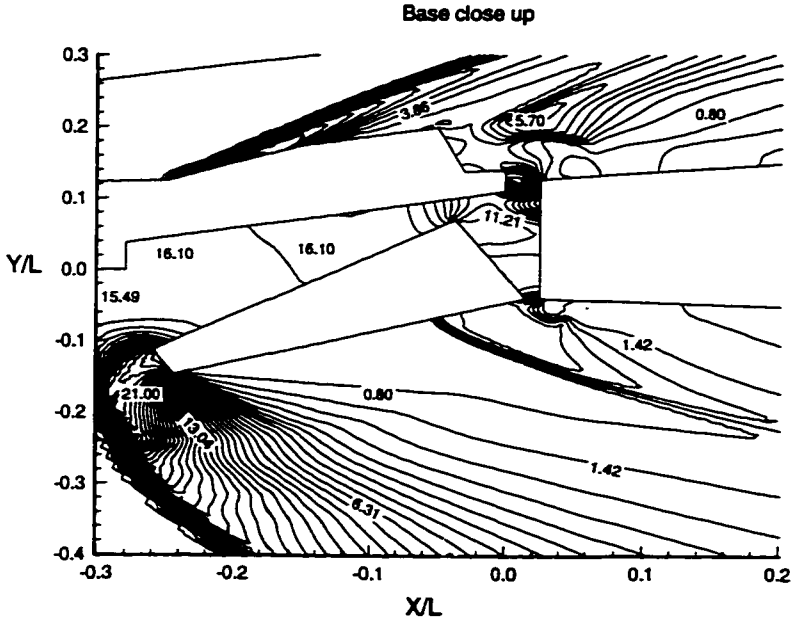


(a)

Figure 5.12 Staging during $M_\infty = 5$. flight: instantaneous ($6 = 5^\circ$) pressure contours when vehicle separates from booster at 85,000 ft altitude ($q_\infty = 1{,}000 lbf / ft^2$). (a) overview , (b) nose close up, (c) base close up.

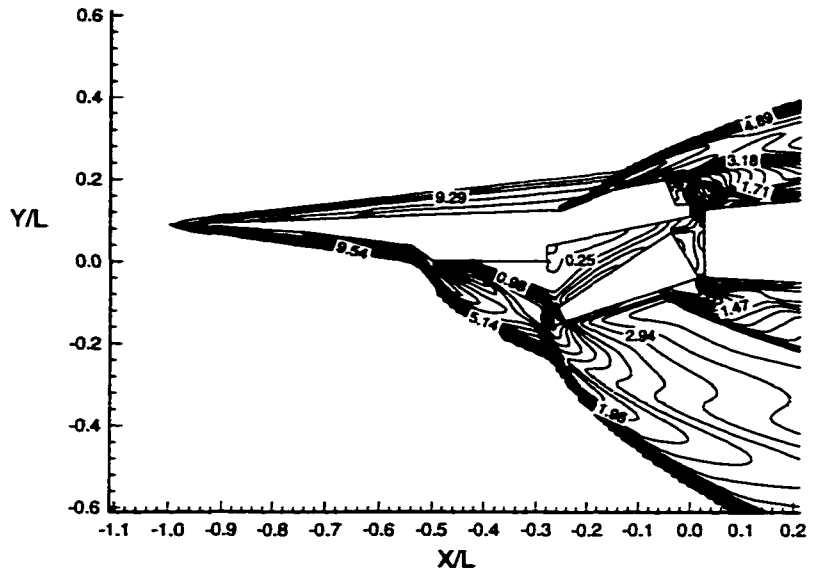


(b)

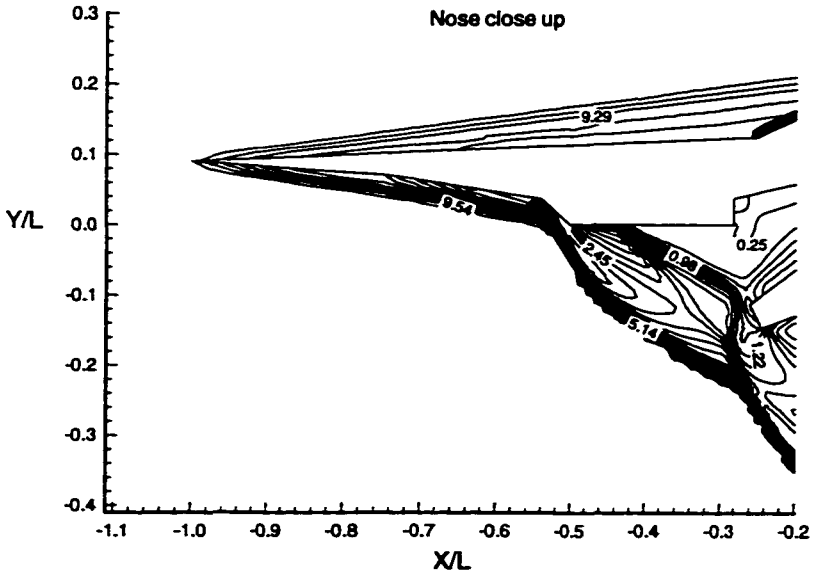


(c)

Figure 5.12 Concluded.



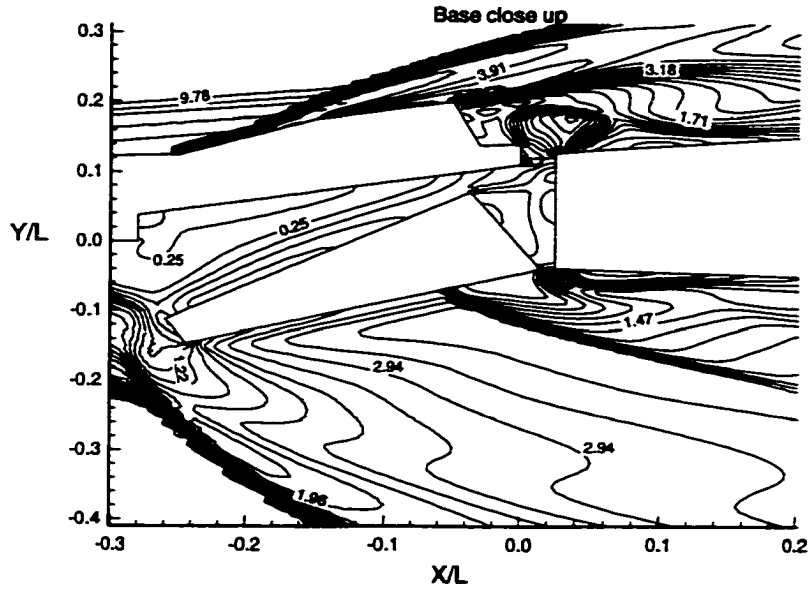
(a)



(b)

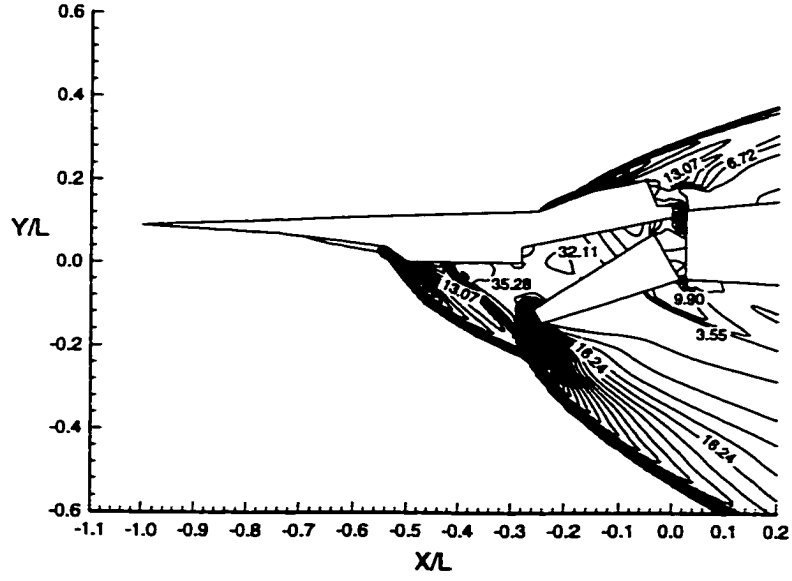
Figure 5.13 Staging during $M_\infty = 10$. flight: instantaneous ($6 = 5^\circ$) Mach contours when vehicle separates from booster at 110,000 ft altitude ($q_\infty = 1{,}000 lbf / ft^2$).

(a) overview , (b) nose close up, (c) base close up.



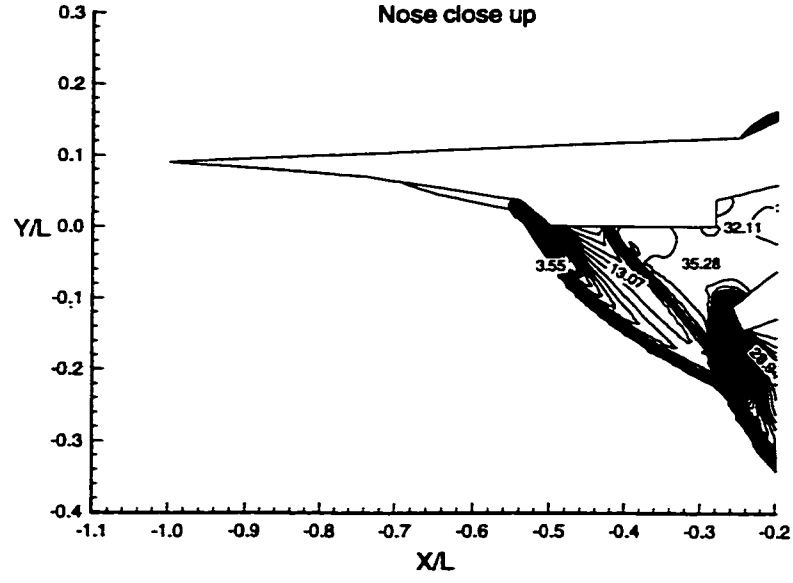
(c)

Figure 5.13 Concluded.

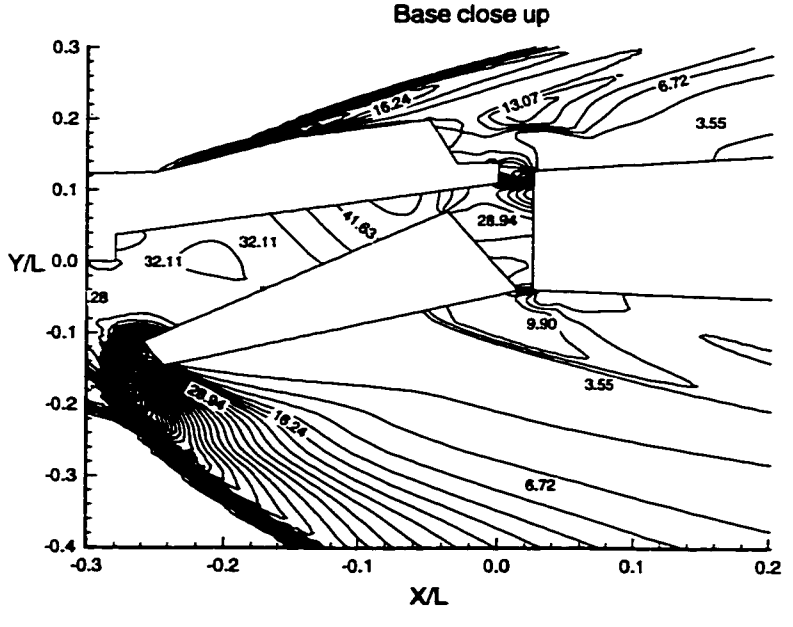


(a)

Figure 5.14 Staging during $M_\infty = 10.$ flight: instantaneous ($\theta = 5^\circ$) pressure contours when vehicle separates from booster at 110,000 ft altitude ($q_\infty = 1{,}000 lbf / ft^2$). (a) overview , (b) nose close up, (c) base close up.



(b)



(c)

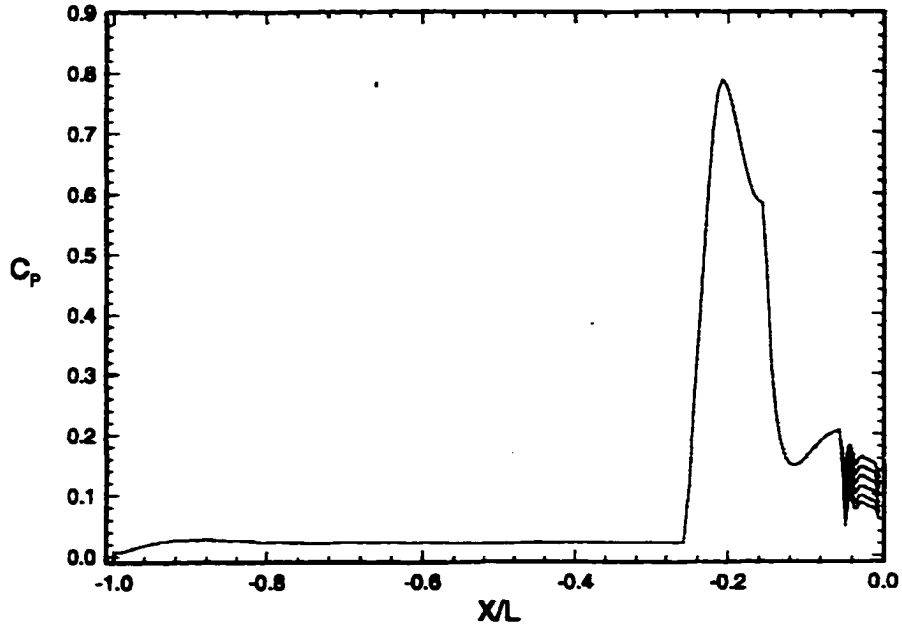
Figure 5.14 Concluded.

To demonstrate the history of aerodynamic interference on the research vehicle during the staging process in Mach 5 and Mach 10 flights, the pressure coefficient distributions on the surfaces of the research vehicle at instants are plotted in figures 5.15-5.18, respectively. From these figures, the range where pressure varied with time was confined to only about 5% of vehicle length on the upper surface, whereas it extended to 20% on the lower surface.

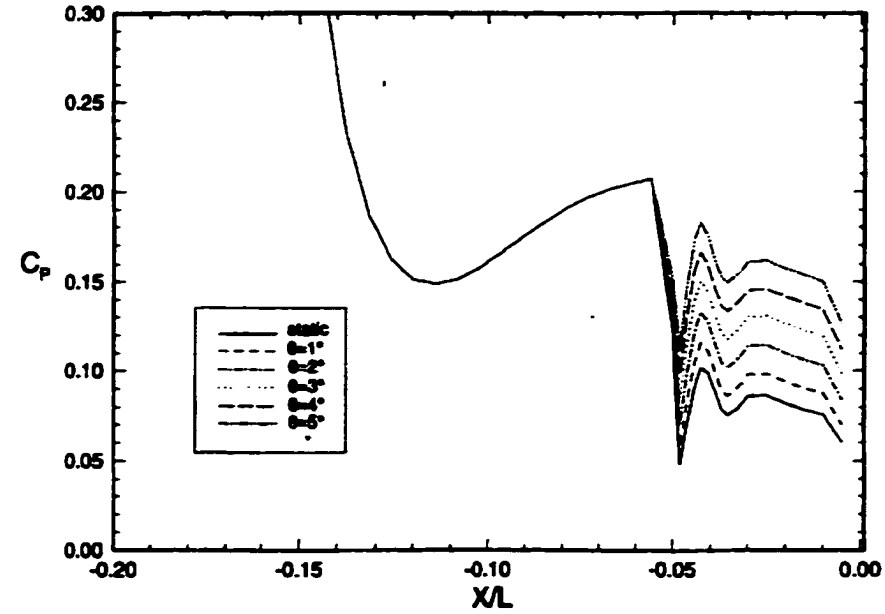
The history of overall aerodynamic forces on the research vehicle during the staging is shown in figures 5.19 and 5.20 for a Mach 5 and Mach 10 flight, respectively. In a Mach 5 flight, as the jaw rotated and translated along with the booster, the rise of pressure on the upper surface was larger than that on the lower surface, resulting in decreasing normal force and moment. The thrust (negative axial force) kept increasing. However, the trends of change of overall aerodynamic forces on the research vehicle in flights were not similar. In the base area of the research vehicle, the pressure rise on the lower surface was slightly greater than that on the upper surface. Hence, the normal force increased along with the increasing thrust contribution, while the axial force decreased. The increase of pitching moment was attributed to the longer torque arm on the upper surface to the point where the resultant pressure was effective.

Normally, the aerodynamic interference ought to have a trend to diminish as the booster and the jaw moved away from the research vehicle. In the Mach 5 flight, the overall axial thrust was completely credited to the aerodynamic interference. Therefore, its value should have a trend to drop to zero and become positive again as the staging advances. However, as observed in figure 5.19, the absolute value of thrust increased, not as expected. The reason may be explained as follows. The existence of a set of

complicated shocks brought up nonlinear effects on the history of aerodynamic interference. As a result, the variation of aerodynamic forces may be a nonlinear function in terms of the distance among components. In dynamic computations, the distance between the research vehicle and the booster increased from 2% of the research vehicle length to 3%. Because the nonlinear axial interference had not yet achieved its peak within the above motion, the interference increased. However, in the Mach 10 flight, the nonlinearity of aerodynamic interference was not so obvious as that in the Mach 5 flight. It is concluded that the history of aerodynamic interference was also highly dependent on the Mach number of free flight.

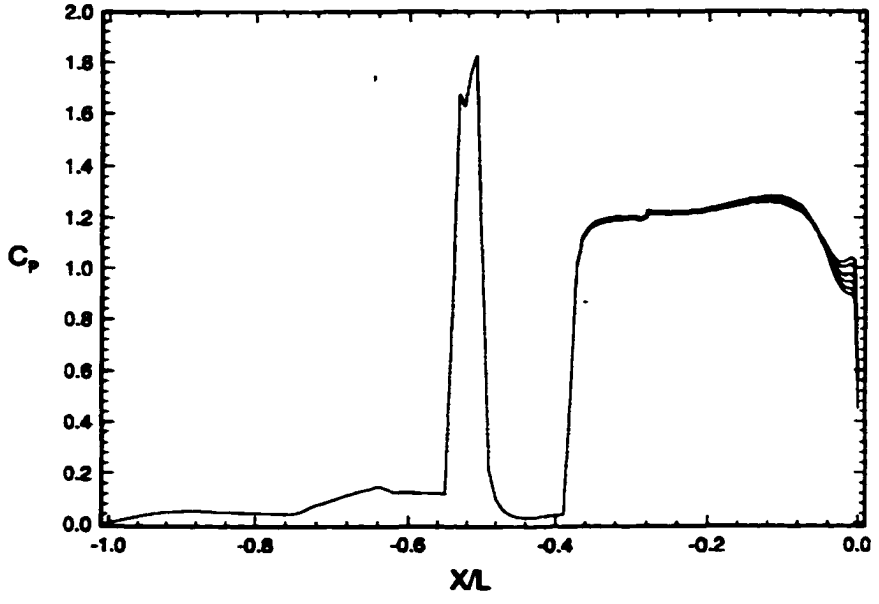


(a)

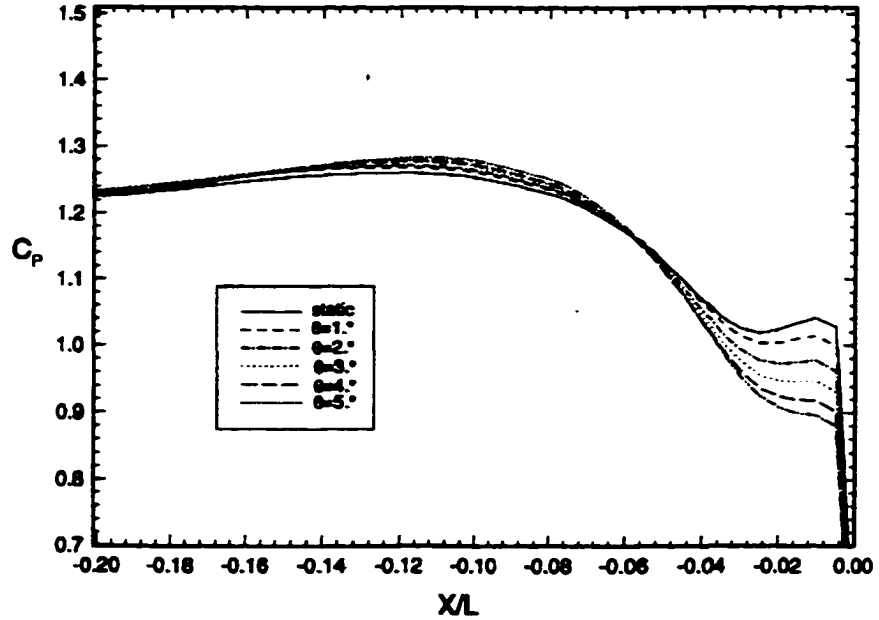


(b)

Figure 5.15 Staging during $M_\infty = 5$. flight: instantaneous pressure coefficient along vehicle's upper surface. $P_{max} = 1828 lbf / ft^2, P_{min} = 63 lbf / ft^2$.
(a) overview, (b) base close up.

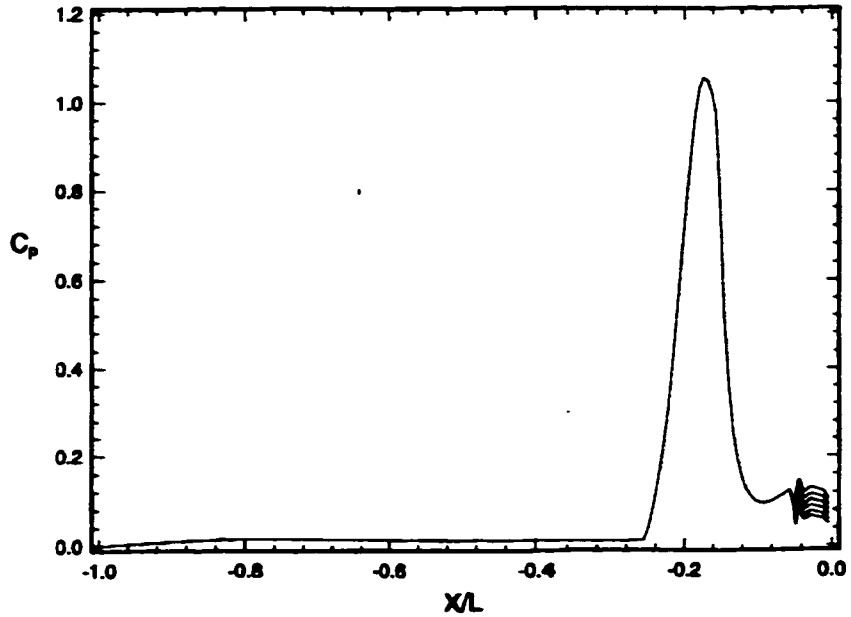


(a)

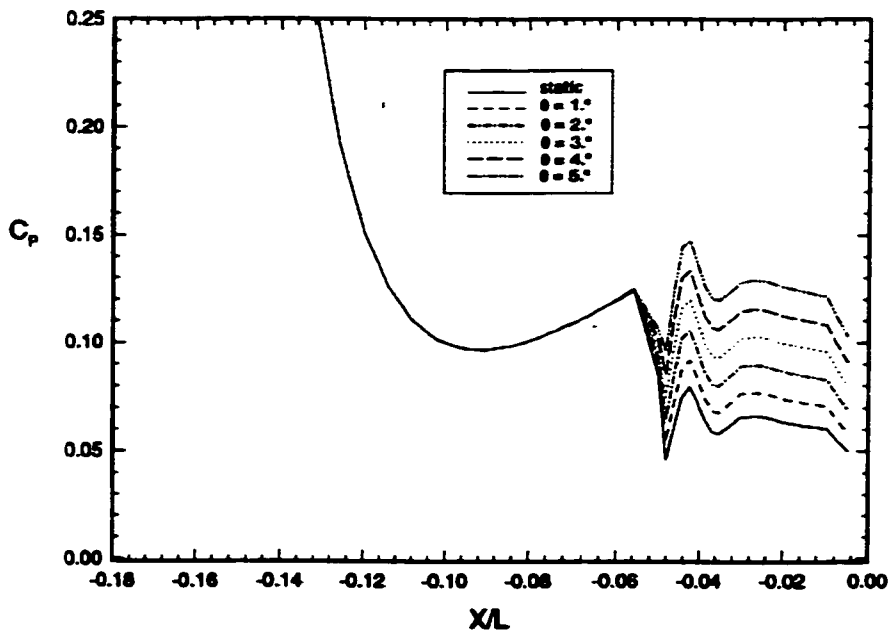


(b)

Figure 5.16 Staging during $M_\infty = 5.$ flight: instantaneous pressure coefficient along vehicle's lower surface. $P_{\max} = 1828 lbf / ft^2, P_{\min} = 63 lbf / ft^2.$
(a) overview, (b) base close up.

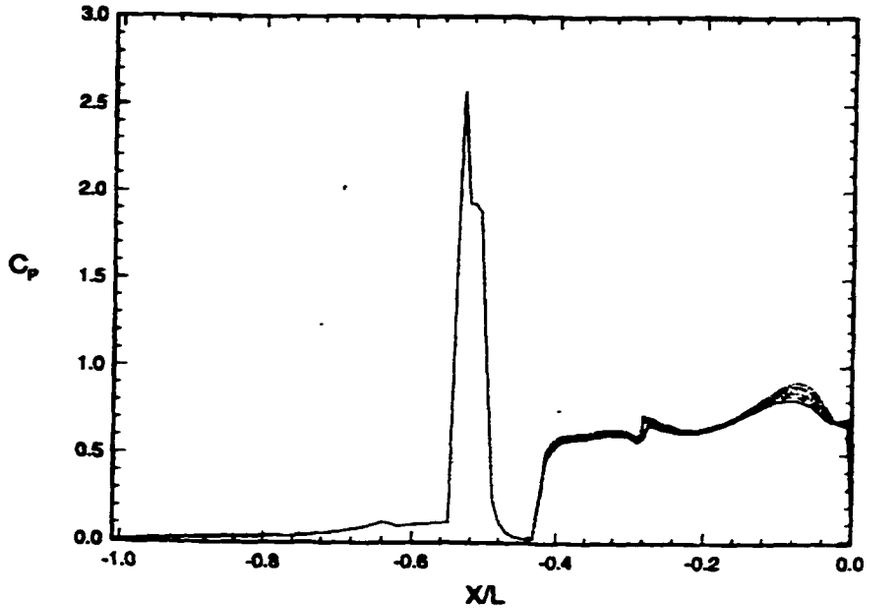


(a)

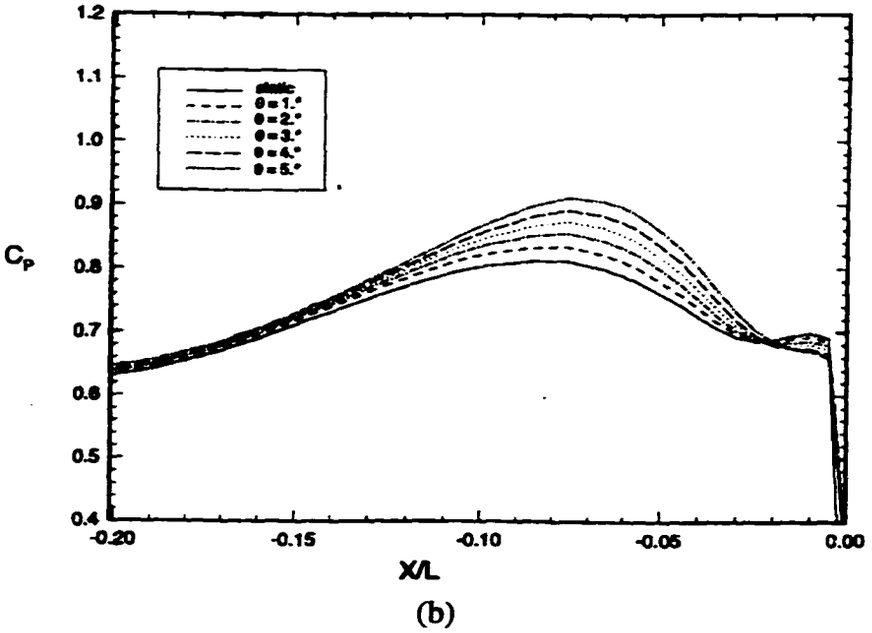


(b)

Figure 5.17 Staging during $M_\infty = 10.$ flight: instantaneous pressure coefficient along vehicle's upper surface. $P_{\max} = 2594 lbf / ft^2, P_{\min} = 33 lbf / ft^2$.
(a) overview, (b) base close up.



(a)



(b)

Figure 5.18 Staging during $M_\infty = 10.$ flight: instantaneous pressure coefficient along vehicle's lower surface. $P_{max} = 2594 lbf / ft^2, P_{min} = 33 lbf / ft^2$.
(a) overview, (b) base close up.

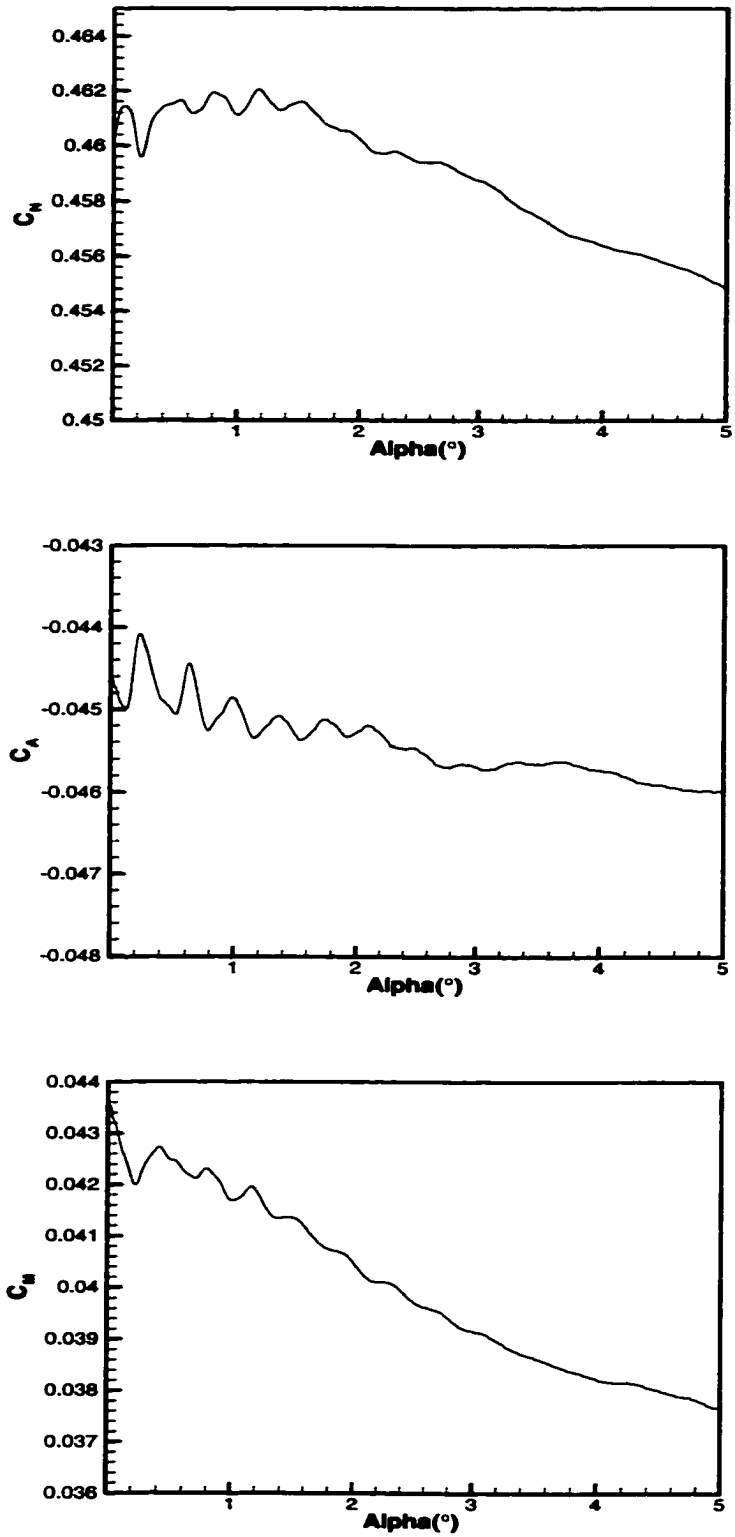


Figure 5.19 Force and moment history on vehicle in staging process $M_\infty = 5$. prescribed motion: $\theta = \omega t$, $\Delta X = V_X t$.

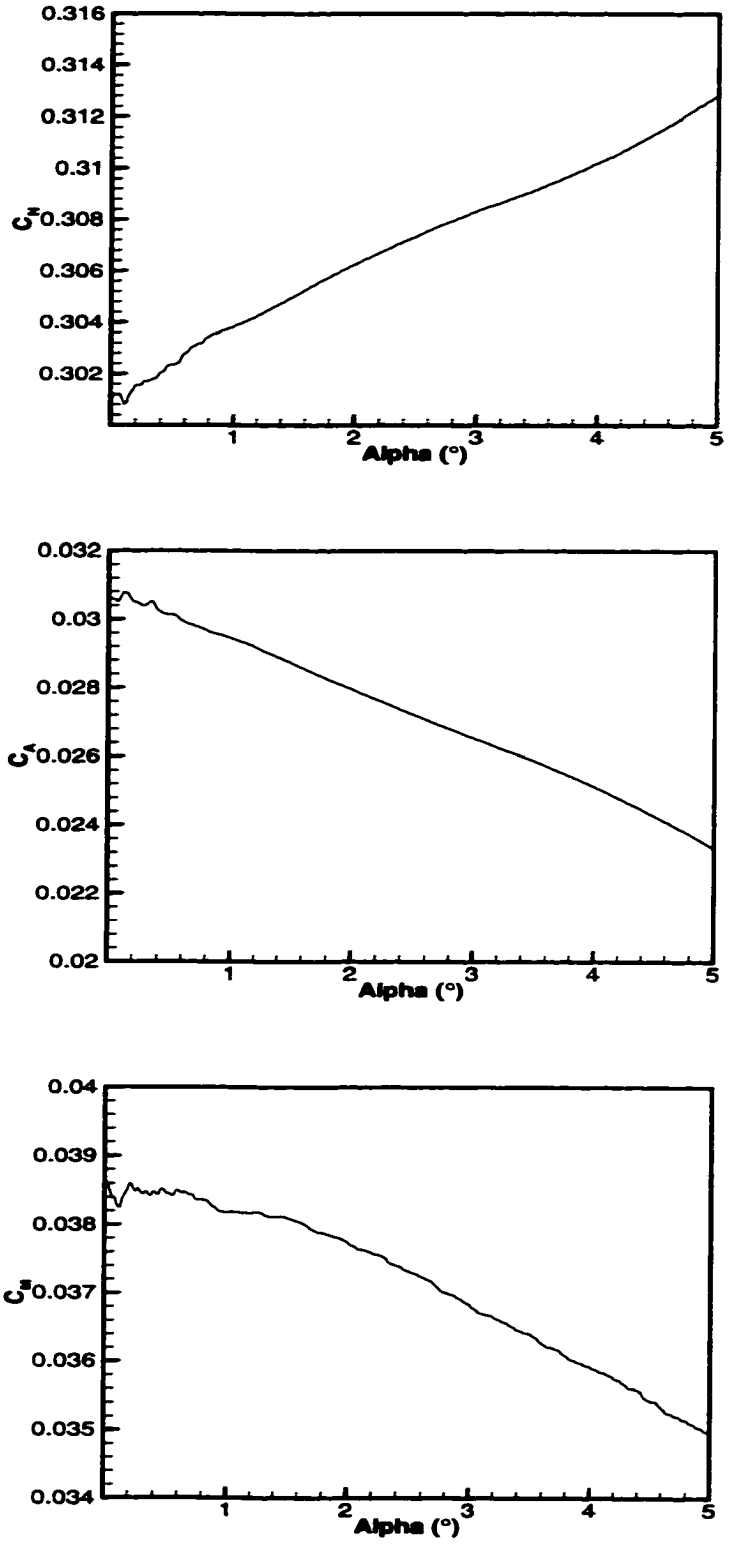


Figure 5.20 Force and moment history on vehicle in staging process $M_\infty = 10$. prescribed motion: $\theta = \omega t$, $\Delta X = V_x t$.

5.3 Three Dimensional Computations

Actually, the hypersonic vehicle in its staging will involve a large translational motion with respect to the booster and the jaw. In the spring analogy method, this kind of large translational motion will result in the deformed mesh with poor quality or edges even crossing each other, which violates the Delaunay property. In fact, two factors, the aerodynamic interaction at extremely close up positions, and the overall history of aerodynamic interference, are most important in a staging. How far away the vehicle is from the booster that aerodynamic interference will completely diminish is a typical example of important information about the stage separation. The interference at extreme close up positions can be handled easily by the dynamic unstructured method, while the first level of assumption, i.e., the animation method, is proven to be a powerful tool for acquiring global information in stage separation. In the animation method, the relative motions of the jaw and the booster with respect to the research vehicle are frozen.

5.3.1 Overall history of aerodynamic interference by animation method

In order to capture the interference history in Mach 5 flight, the assumed staging path was divided into six positions, where static analyses were performed. The definitions of these positions are given in Table 5.2. Here, ΔX is the distance between the research vehicle and the booster in the longitudinal direction; δ is the angle between the normal direction of the jaw base and the vertical axis; L is the research vehicle length. The symbol * indicates that the jaw is dropped from the computational domain.

Table 5.2 Cases definitions in terms of relative positions during staging.

	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
ΔX	0.0455 L	0.225 L	0.589 L	0.862 L	1.771 L	$\rightarrow \infty$
δ	20°	25°	25°	25°	*	*

Since the booster is removed in Case 6, it represents the free flight. The meshes for these cases were generated by the advancing front method. The pertinent parameters for these meshes are presented in Table 5.3.

Table 5.3 Mesh parameters for different cases.

	Ncell	*Nnode*	*Nedge*	*Ntriangle*	*Npointb*
Case 1	971,021	179,646	1,171,706	42,080	21,042
Case 2	1,082,772	199,338	1,303,608	42,998	21,501
Case 3	1,126,647	207,366	1,356,244	44,464	22,234
Case 4	1,129,329	207,942	1,359,640	44,740	22,372
Case 5	1,035,935	190,539	1,246,766	40,586	20,295
Case 6	837,727	153,678	1,006,959	31,090	15,547

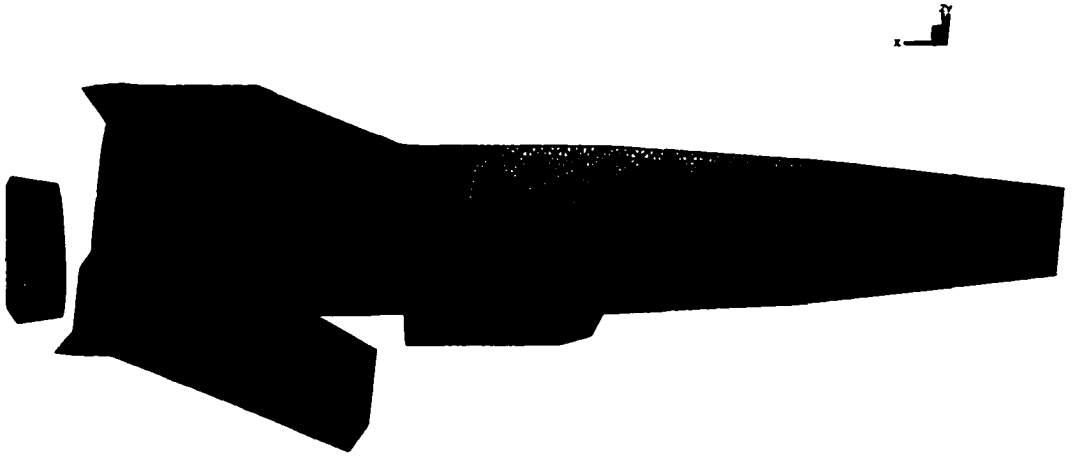
Here, *Ncell*, *Nnode*, and *Nedge* are the number of cells, nodes, and edges inside the computational domain; denoted by *Ntriangle* and *Npointb* are the number of triangles and nodes on the boundary surfaces, respectively.

Presented in figure 5.21 are two views of the surface mesh used in Case 1. Since it is difficult to display a 3D unstructured mesh, symmetry-plane meshes in all cases are shown in figure 5.22. Two dynamic simulations were performed in Case 1 and Case 2, where the booster was extremely close to the research vehicle, with the same two-degrees-of relative motion as 2D cases.

Using a finite element method with an efficient edge-based data structure, only several ten Mbytes running memory was incurred on these unstructured meshes, as compared to several hundred Mbytes for the cell-center data structure.

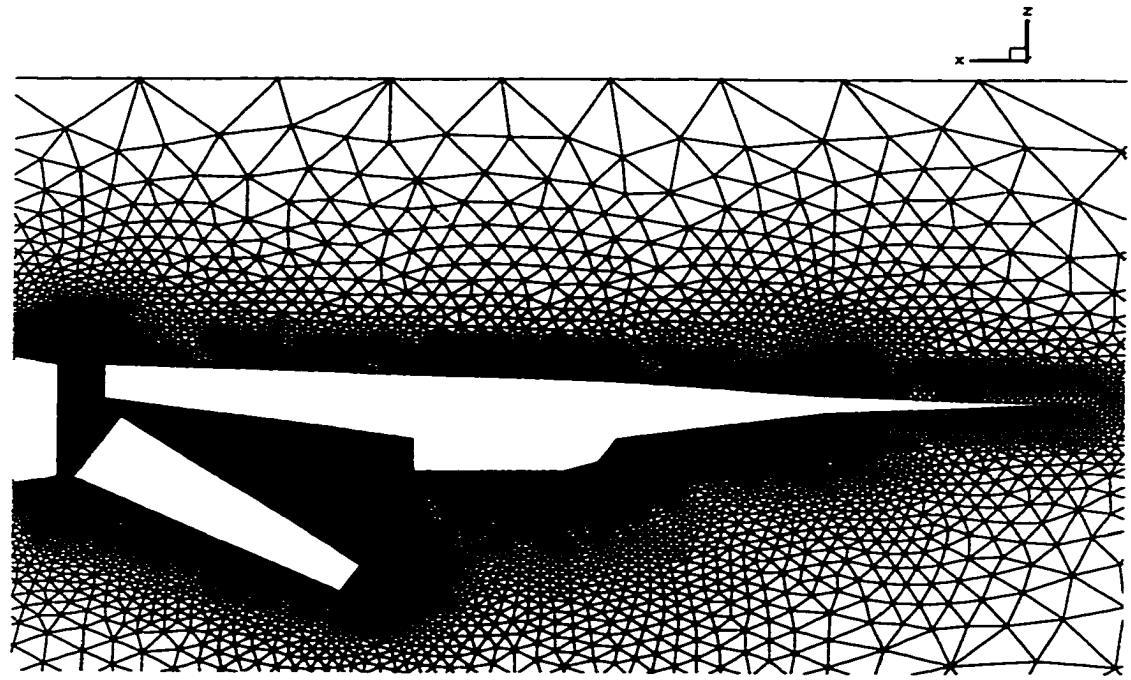
The Mach and pressure contours on the symmetry plane and the off-surface Mach contours and pressure contours on surfaces are shown in figure 5.23~5.28, respectively.

(a)

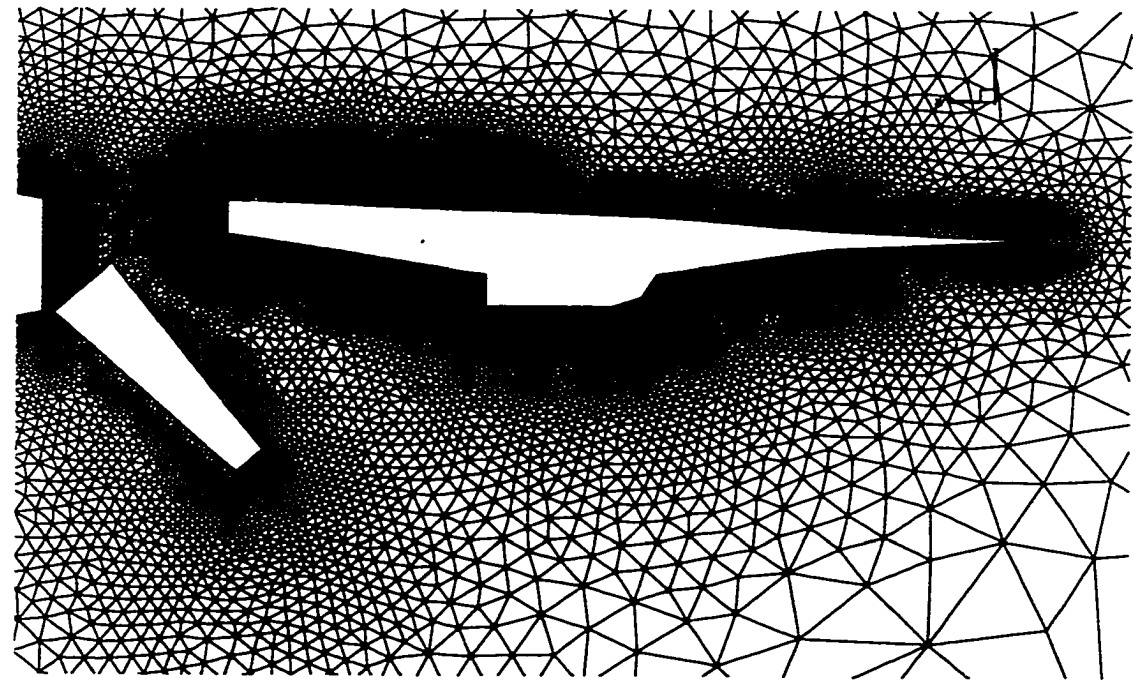


(b)

Figure 5.21 3-D unstructured mesh in case1: (a) top view, (b) bottom view.

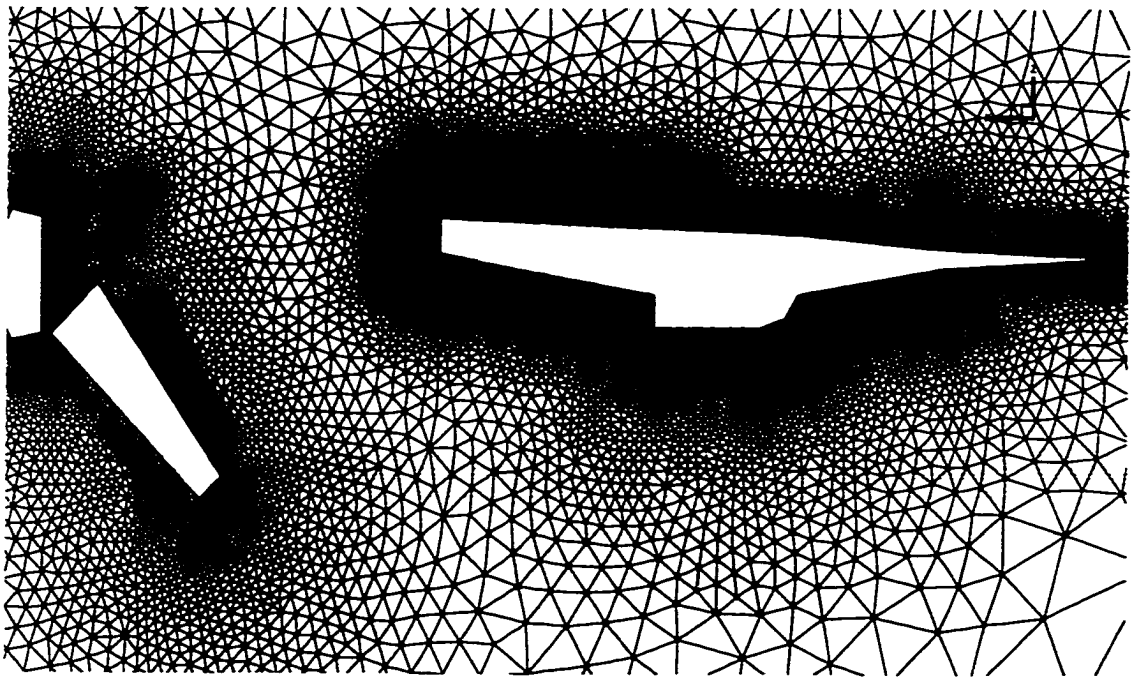


(a)

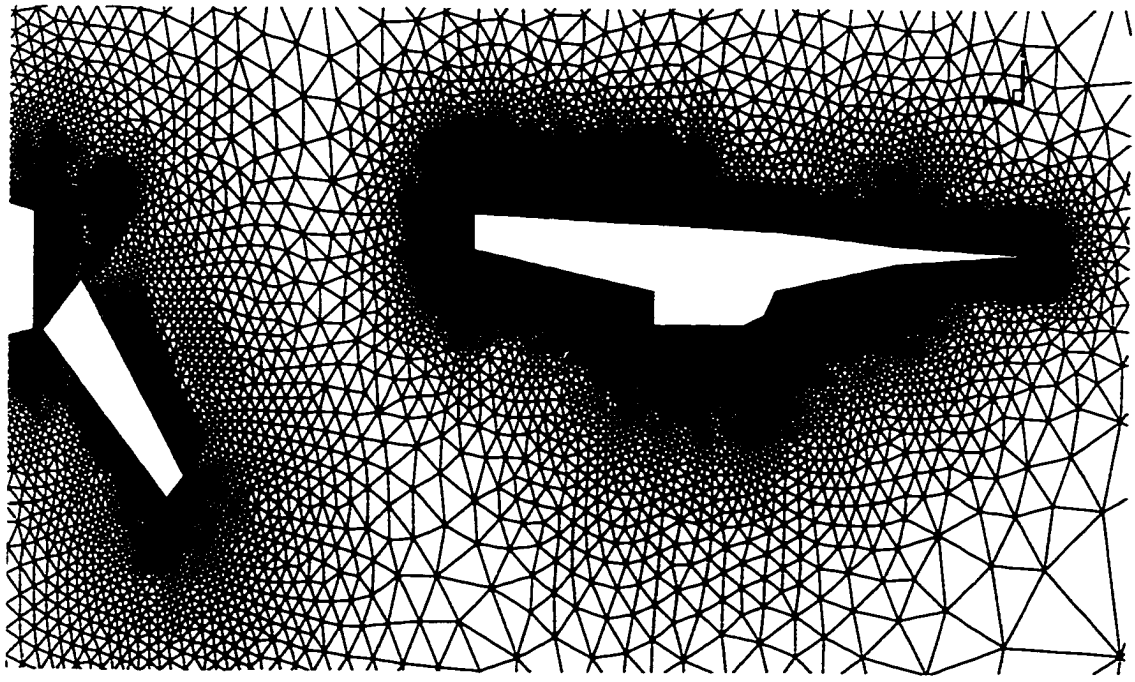


(b)

Figure 5.22 Symmetry-plane meshes for different cases:
Case 1, (b) Case 2, (c) Case 3, (d) Case 4, (e) Case 5, (f) Case 6.

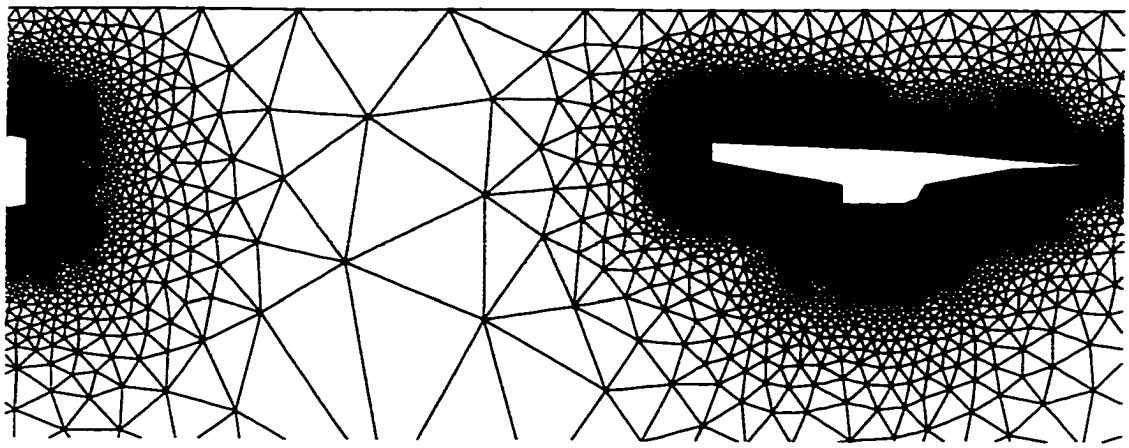


(c)

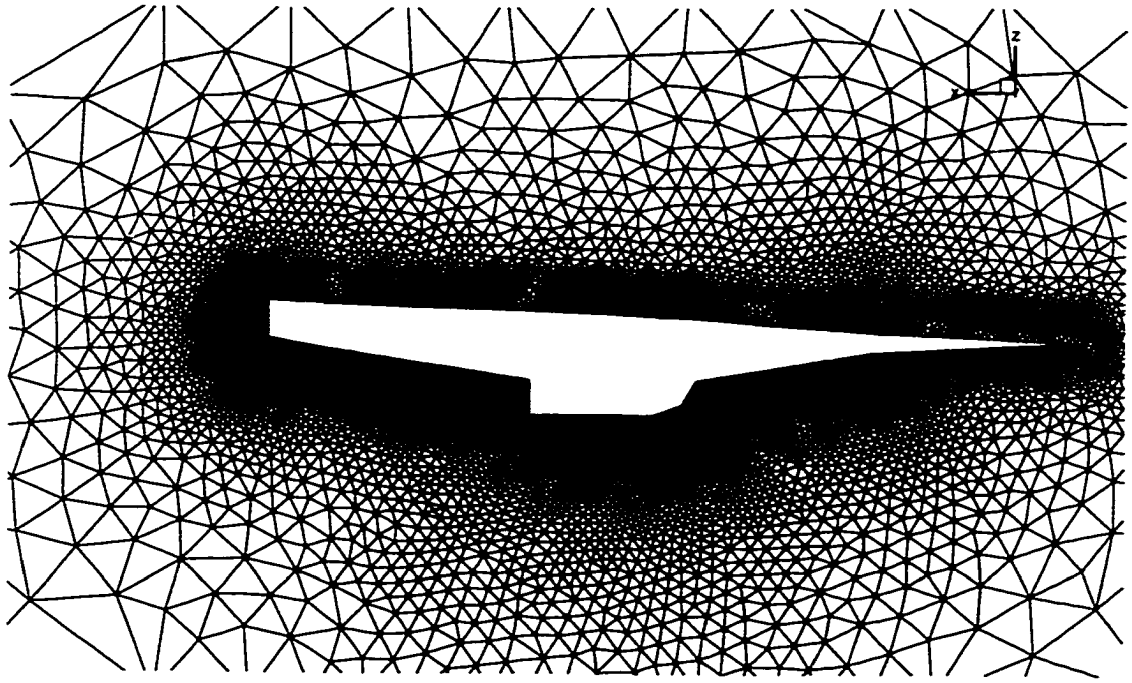


(d)

Figure 5.22 Continued.

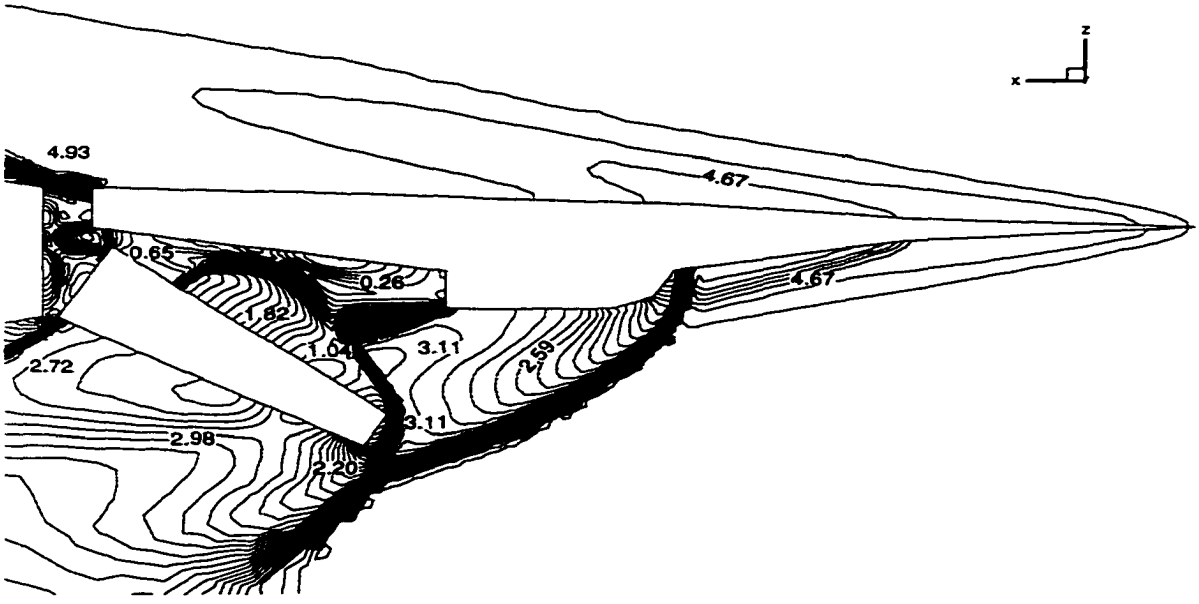


(e)

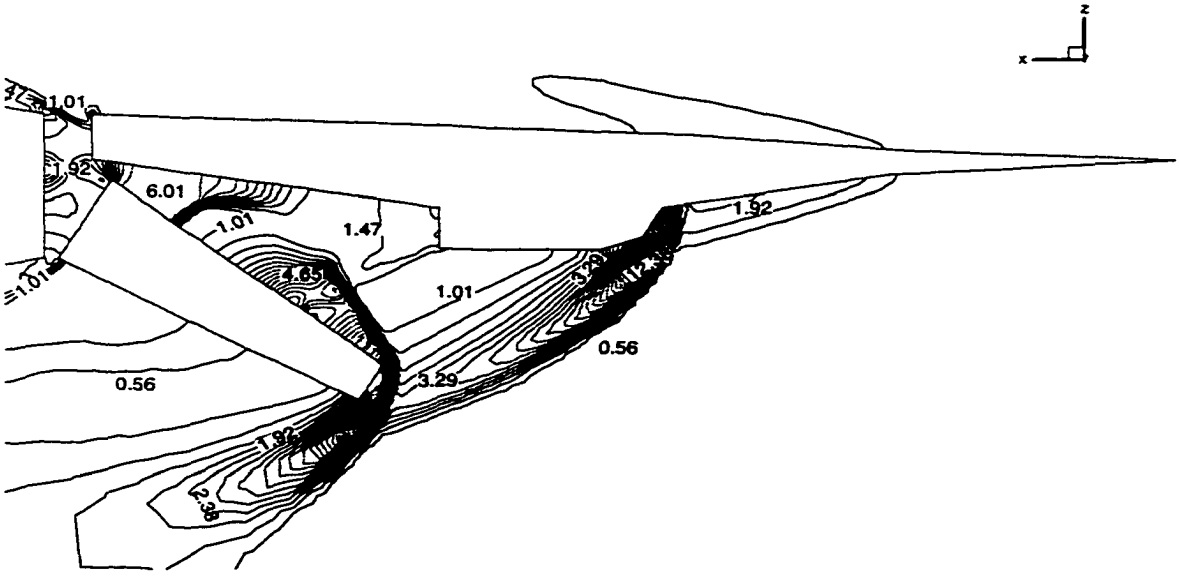


(f)

Figure 5.22 Concluded.



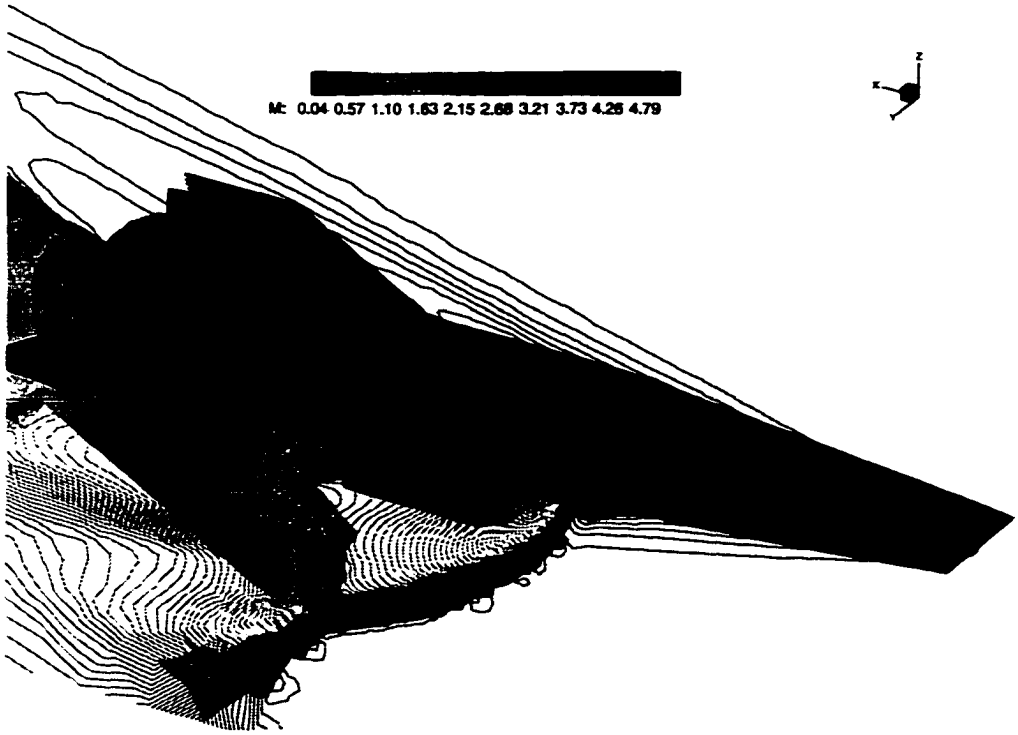
(a)



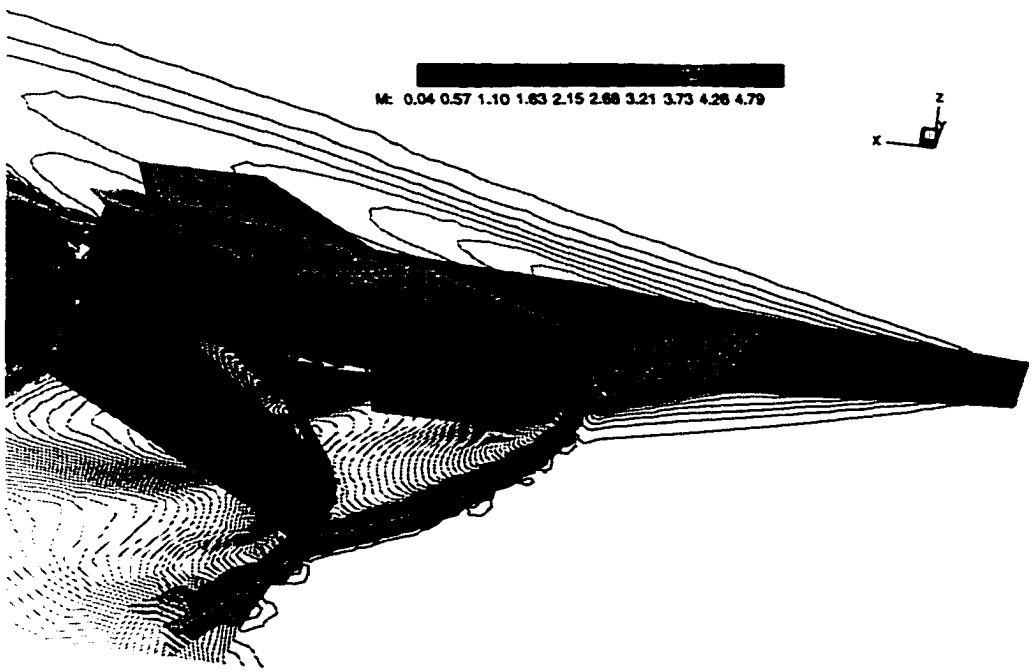
(b)

Figure 5.23 Instantaneous values for Case 1 during stage separation.

(a) Mach contours on the symmetry plane, (b) Pressure contours on the symmetry plane,
(c) Top view of Mach contours, (d) Bottom view of Mach contours,
(e) Top view of pressure contours, (f) Bottom view of pressure contours.

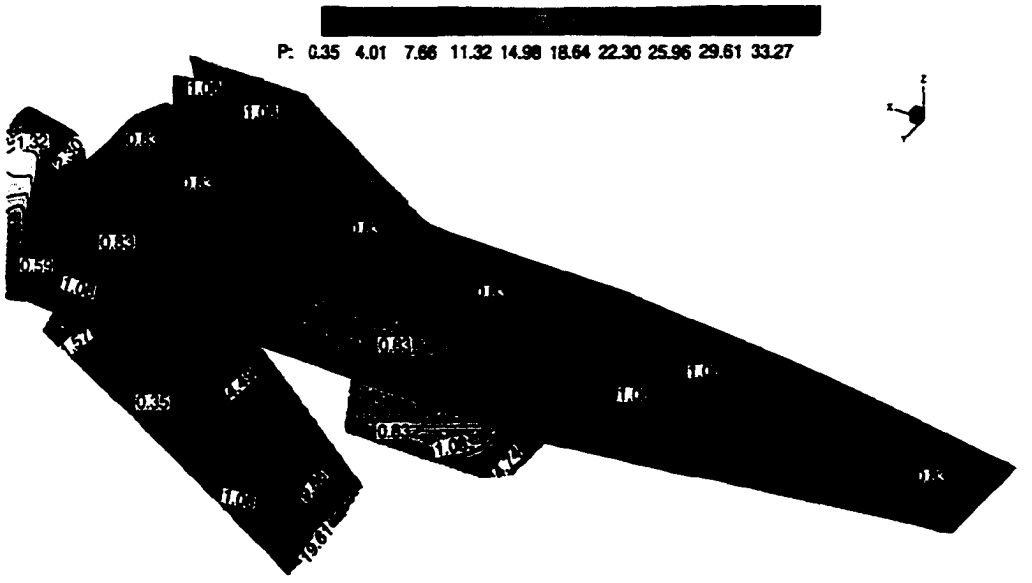


(c)

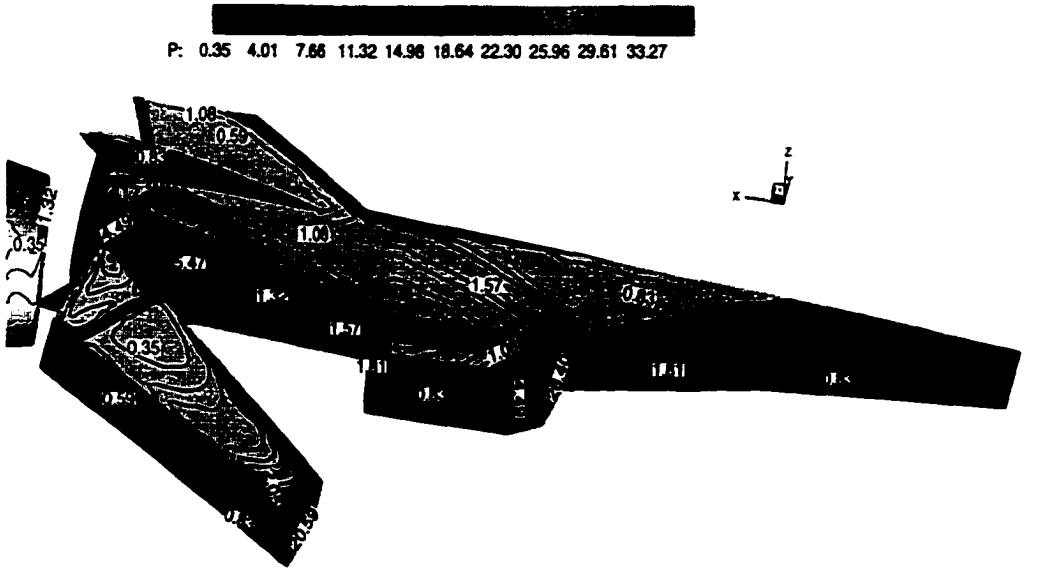


(d)

Figure 5.23 Continued.

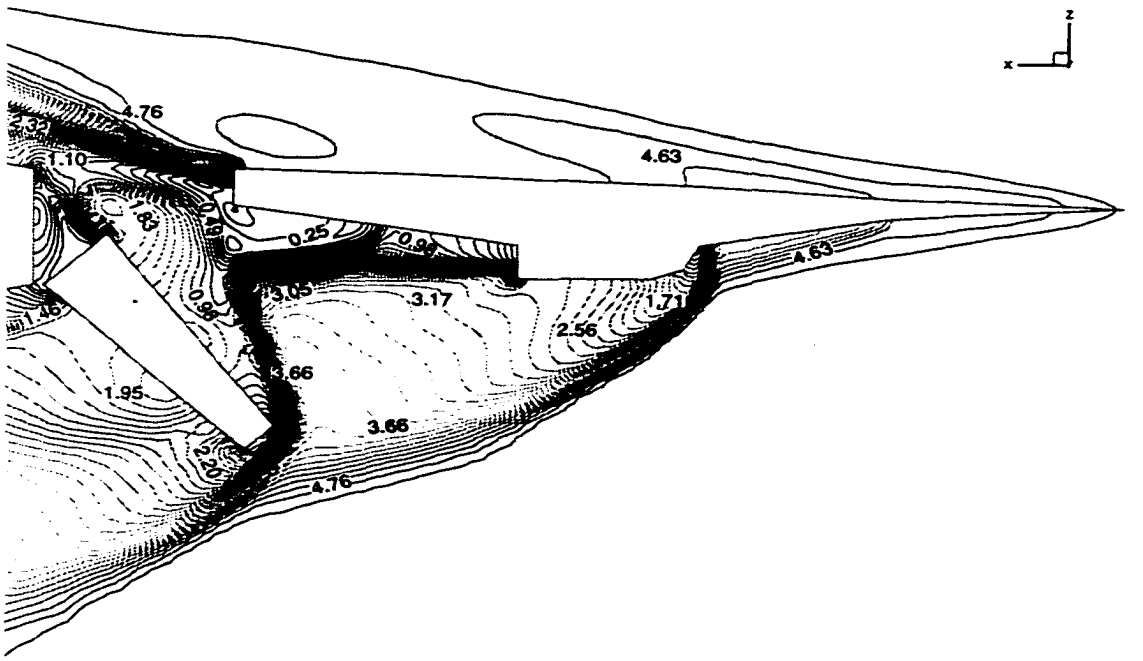


(e)

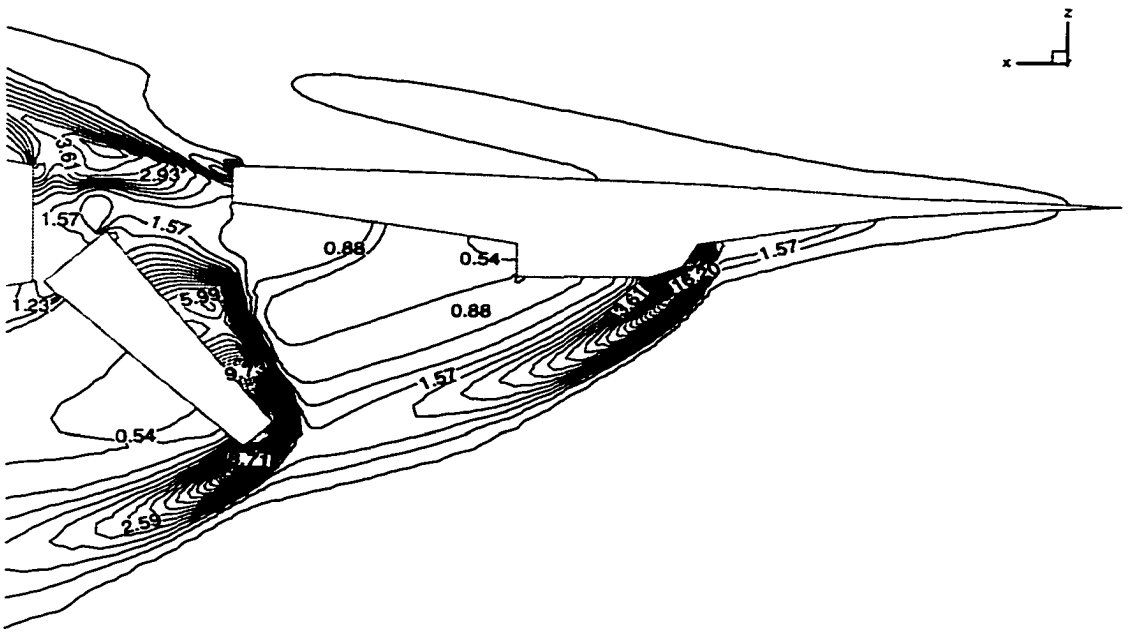


(f)

Figure 5.23 Concluded.



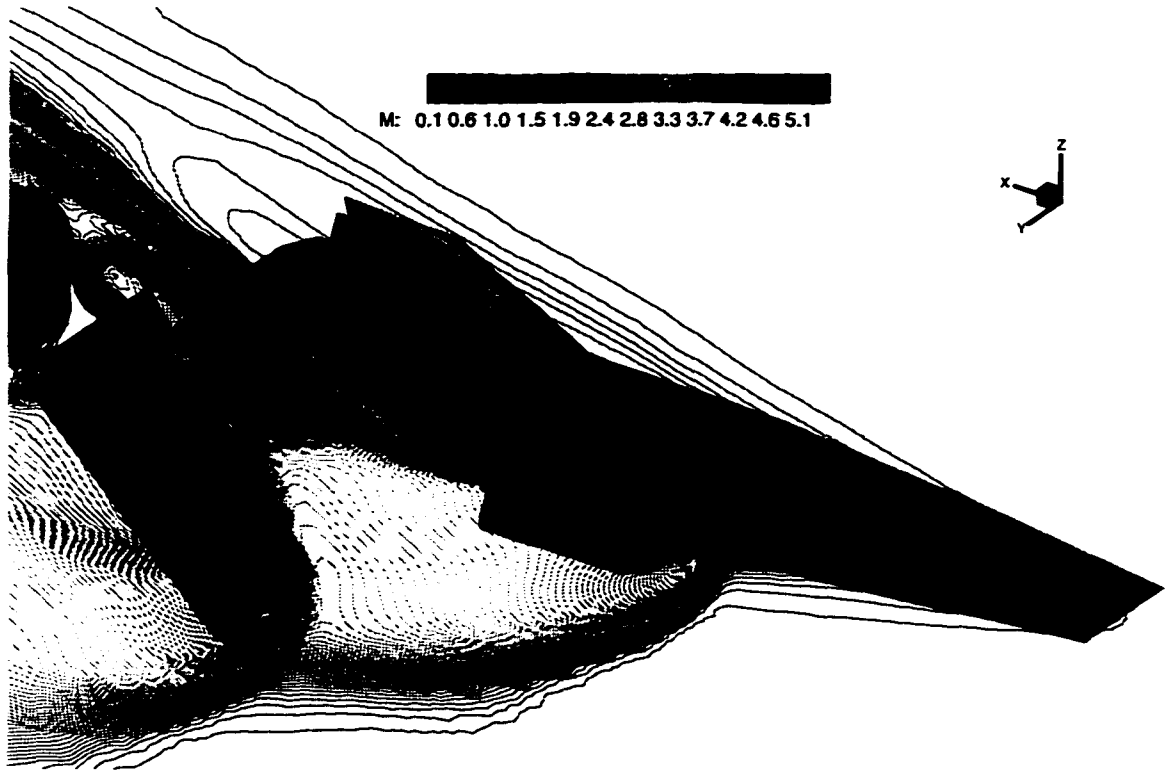
(a)



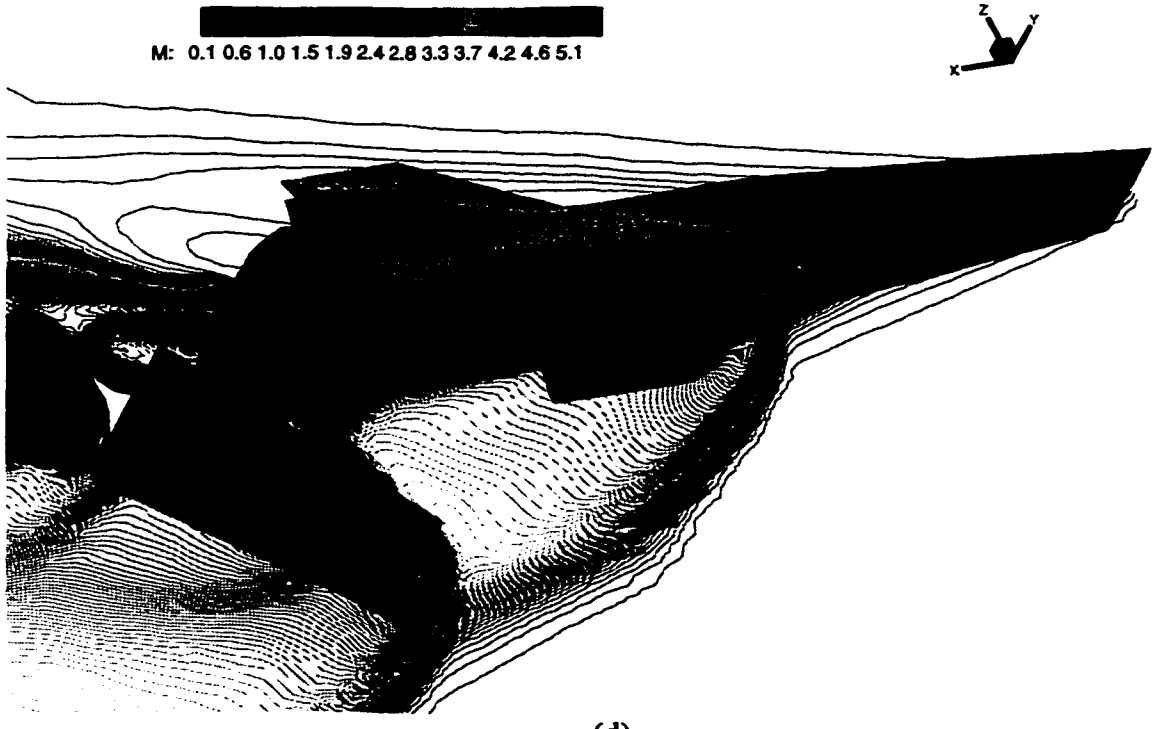
(b)

Figure 5.24 Instantaneous values for Case 2 during stage separation.

(a) Mach contours on the symmetry plane, (b) Pressure contours on the symmetry plane,
(c) Top view of Mach contours, (d) Bottom view of Mach contours,
(e) Top view of pressure contours, (f) Bottom view of pressure contours.

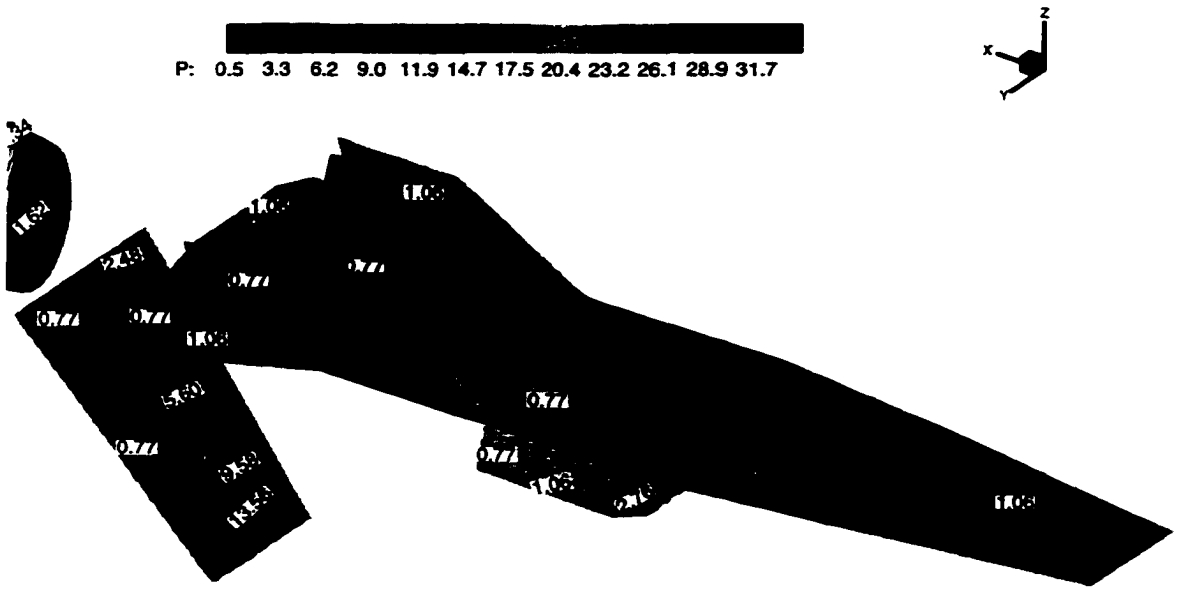


(c)

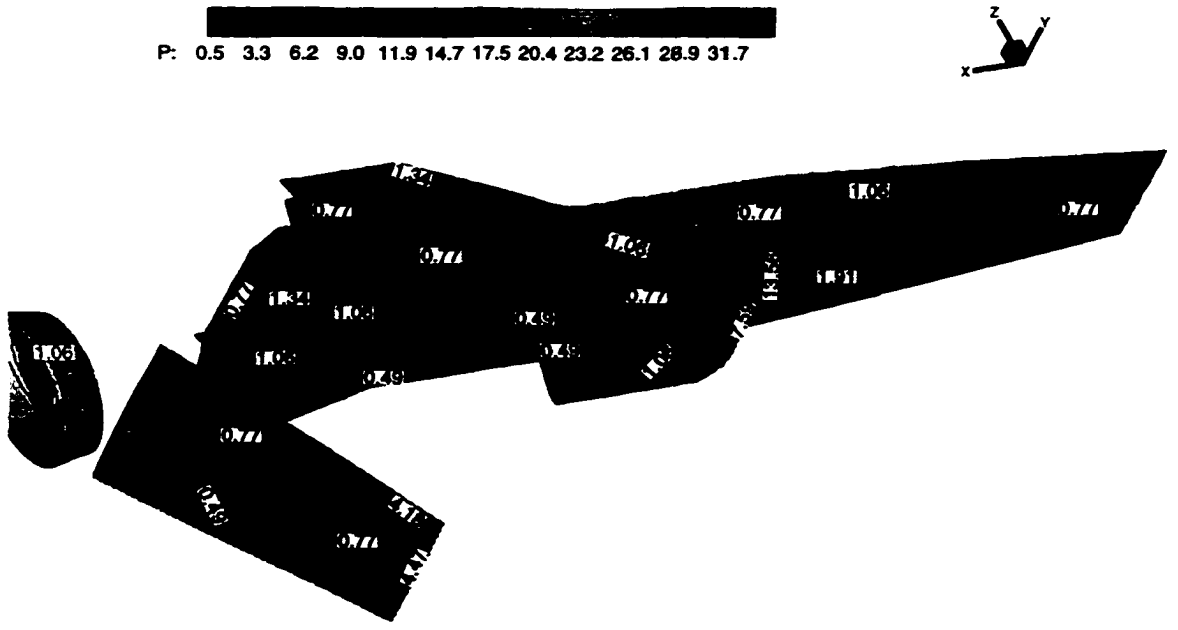


(d)

Figure 5.24 Continued.

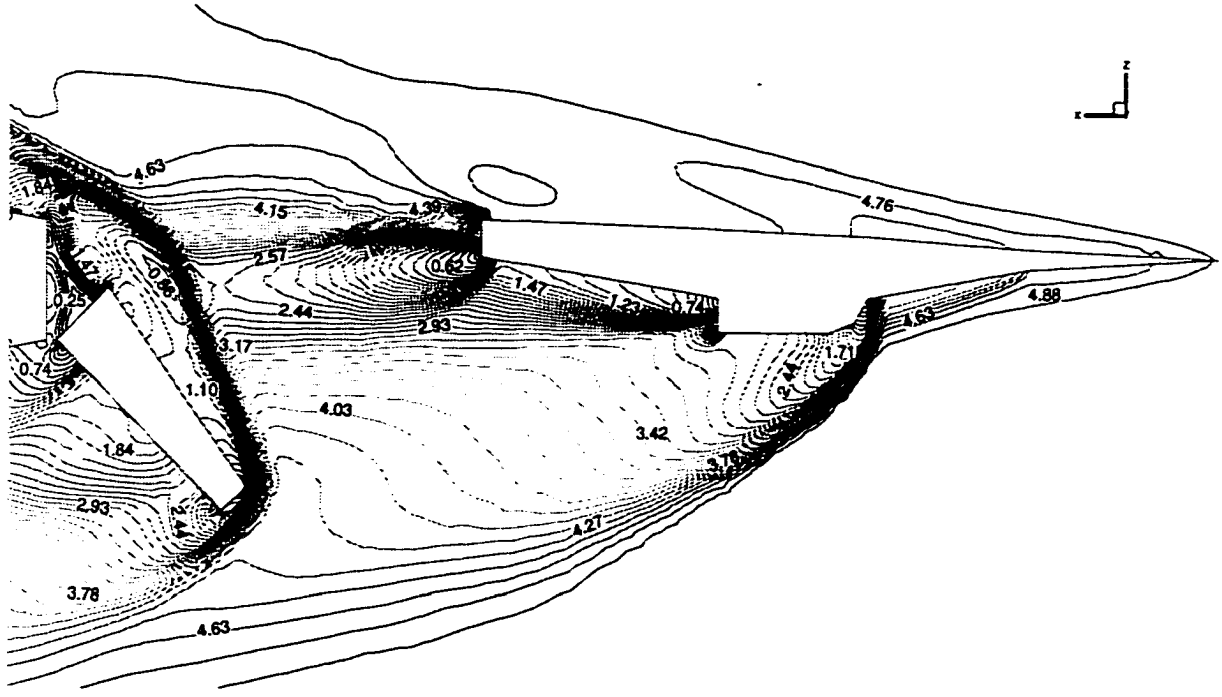


(e)

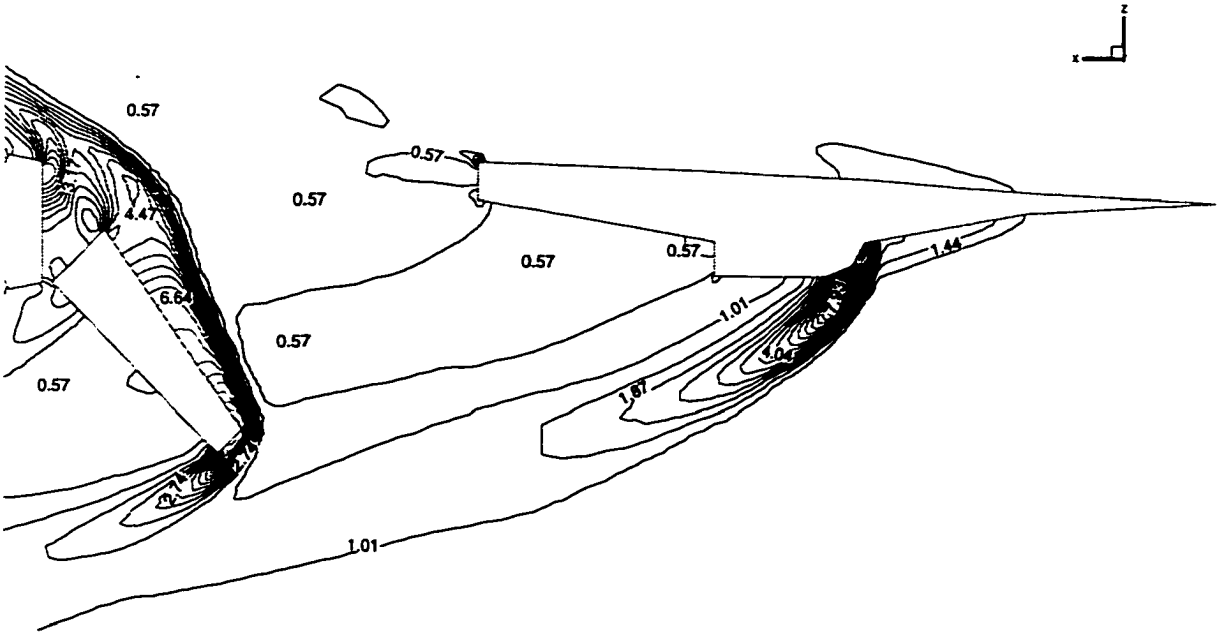


(f)

Figure 5.24 Concluded.



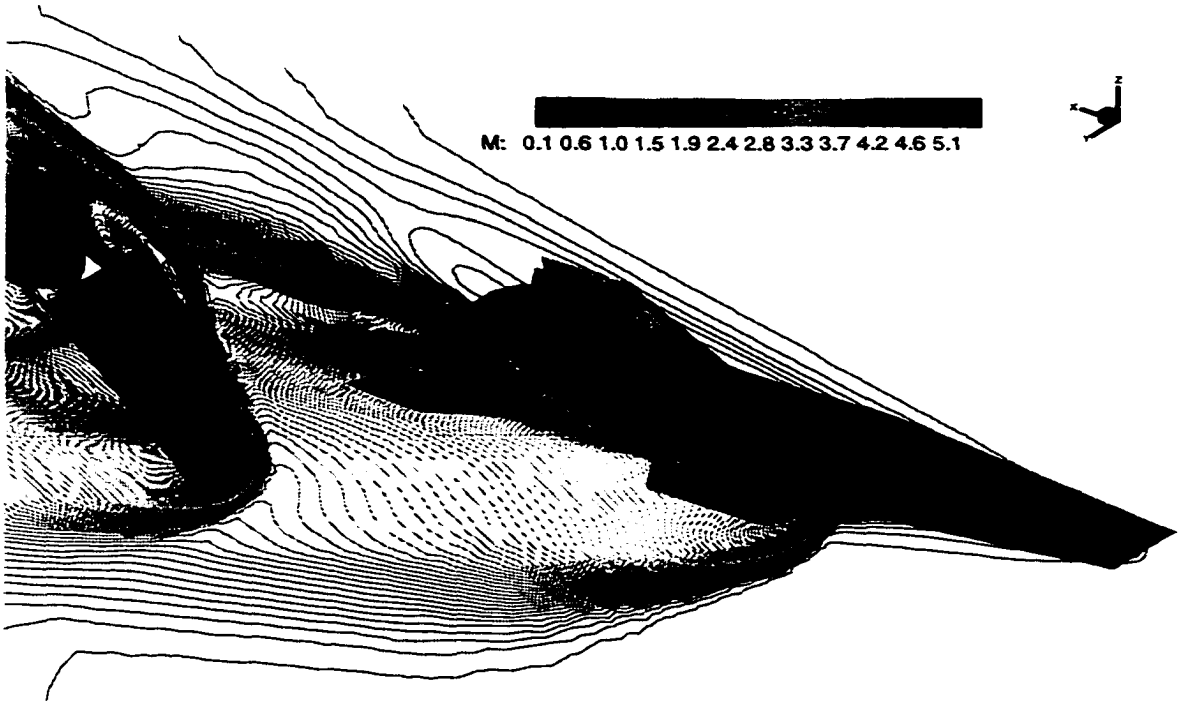
(a)



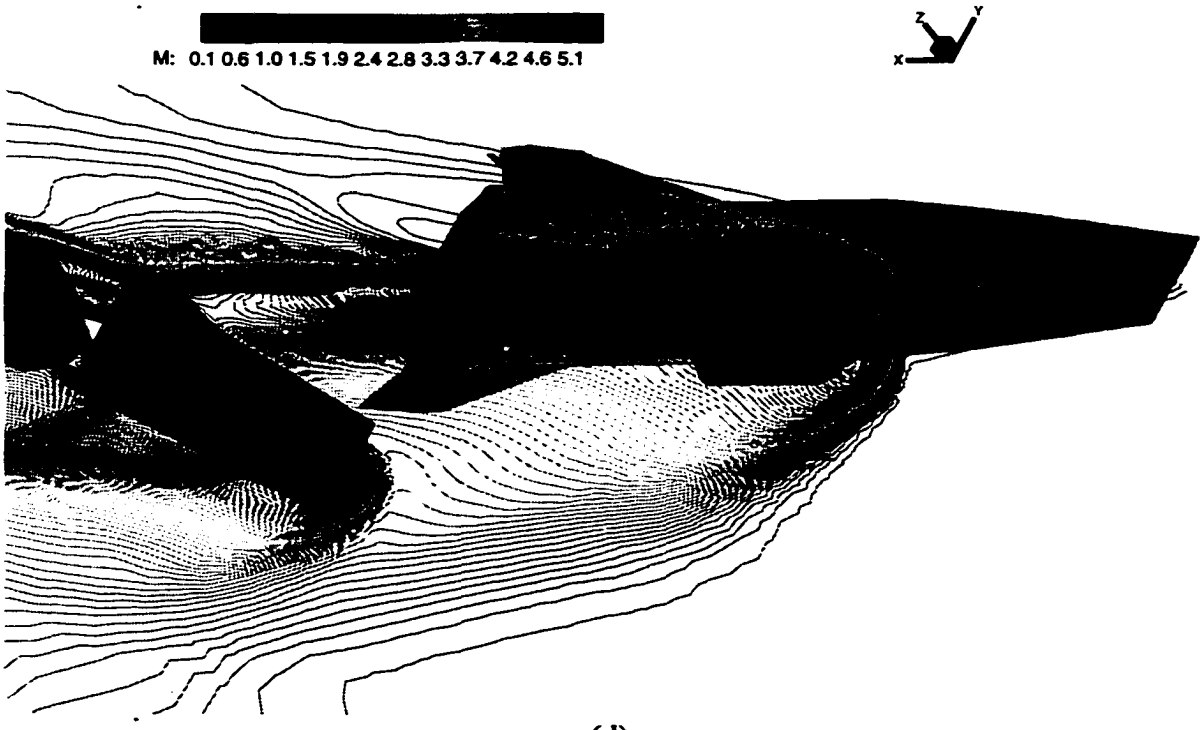
(b)

Figure 5.25 Instantaneous values for Case 3 during stage separation.

(a) Mach contours on the symmetry plane, (b) Pressure contours on the symmetry plane,
(c) Top view of Mach contours, (d) Bottom view of Mach contours,
(e) Top view of pressure contours, (f) Bottom view of pressure contours.

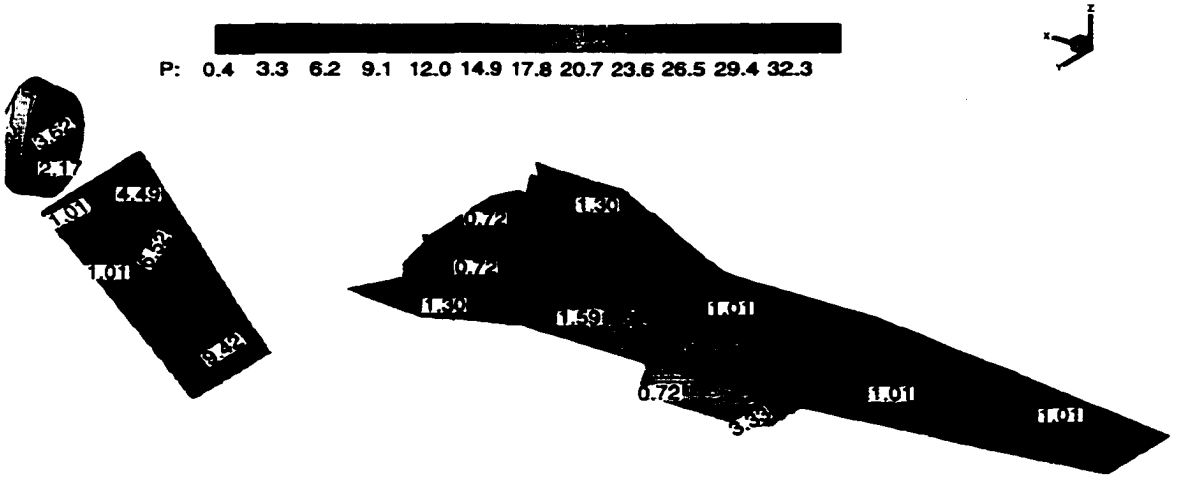


(c)

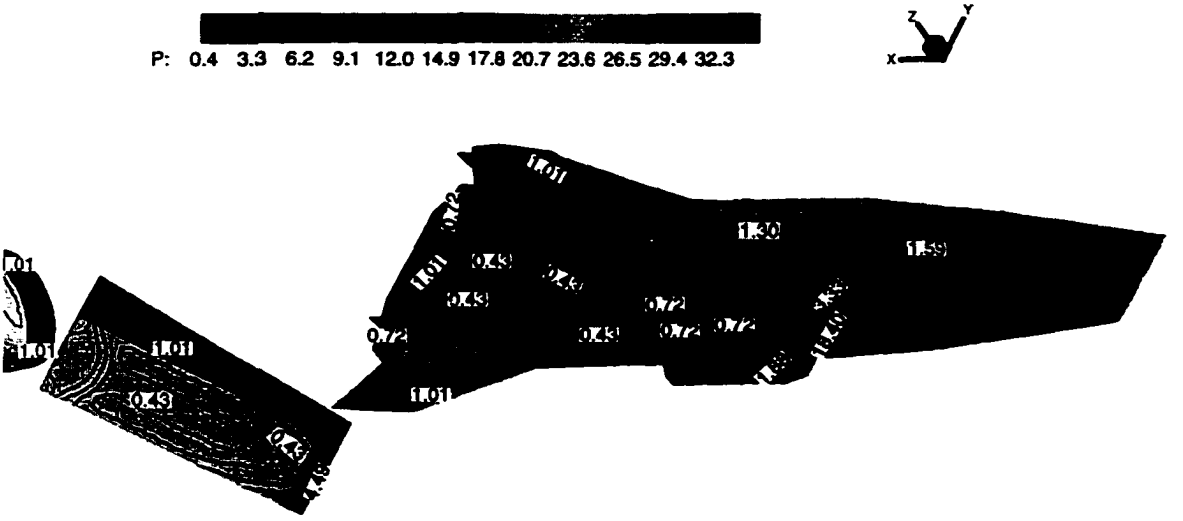


(d)

Figure 5.25 Continued.

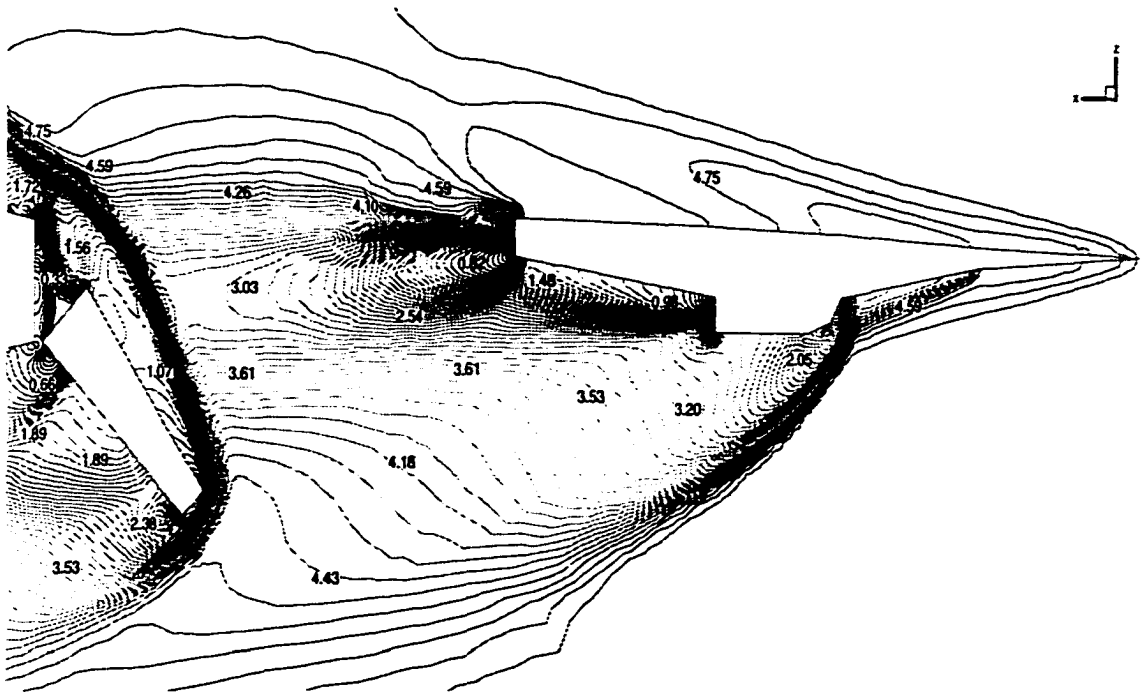


(e)

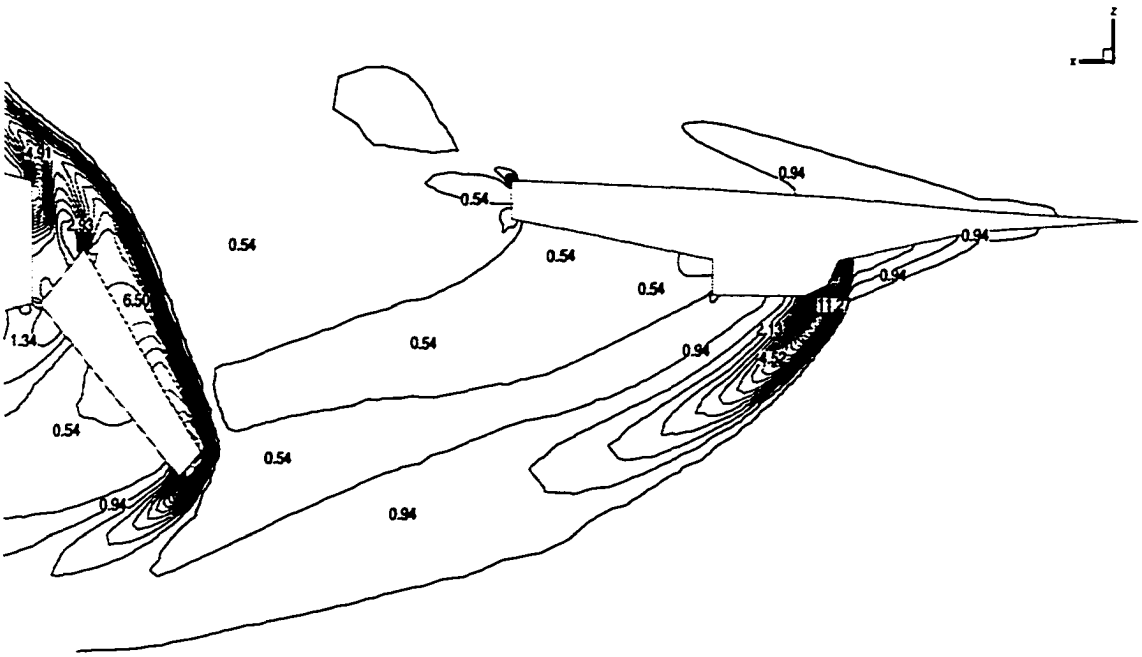


(f)

Figure 5.25 Concluded.



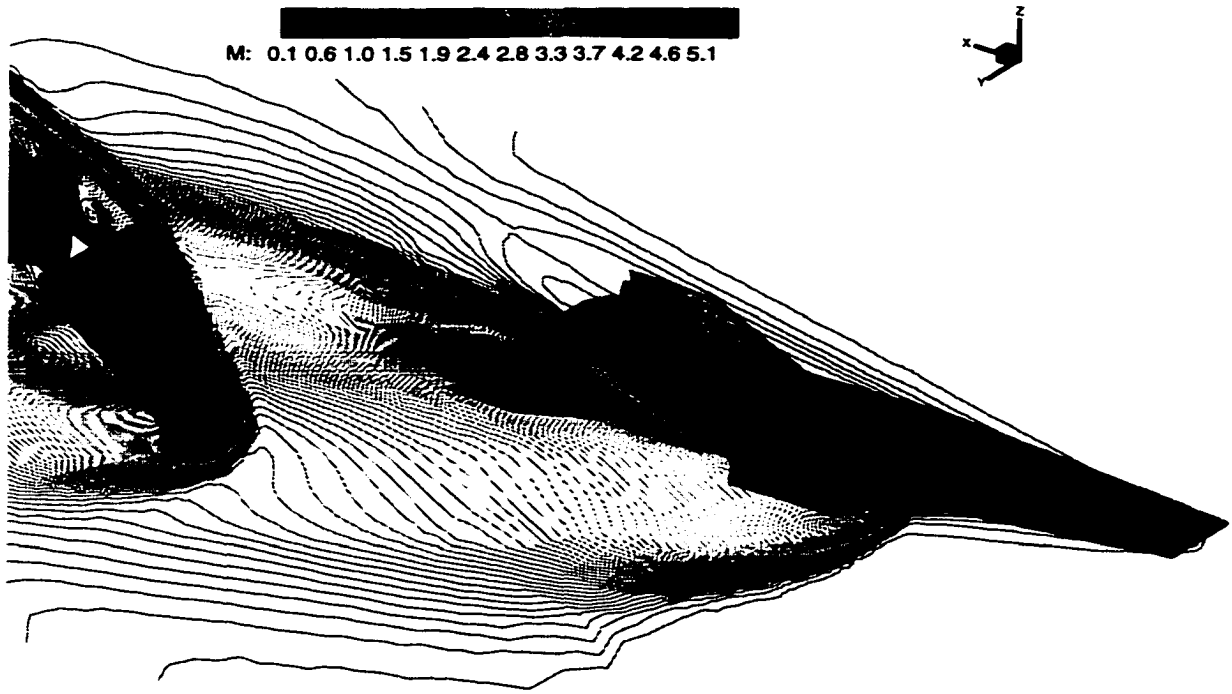
(a)



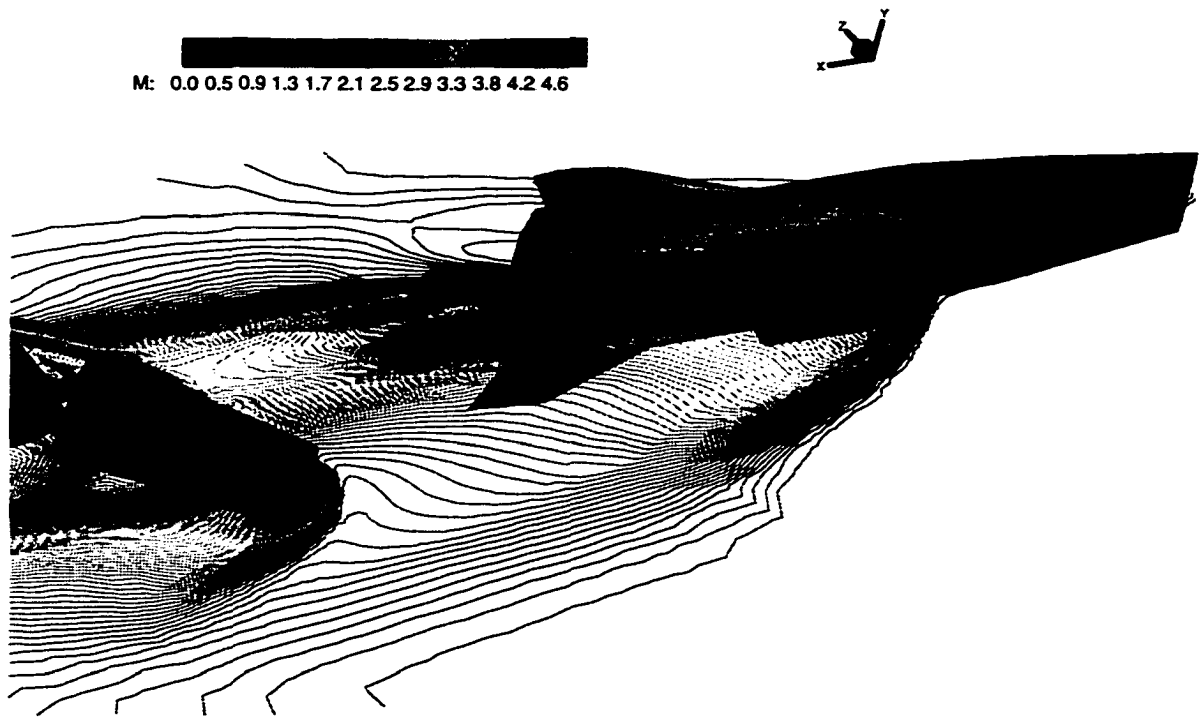
(b)

Figure 5.26 Instantaneous values for Case 4 during stage separation.

(a) Mach contours on the symmetry plane, (b) Pressure contours on the symmetry plane,
(c) Top view of Mach contours, (d) Bottom view of Mach contours,
(e) Top view of pressure contours, (f) Bottom view of pressure contours.

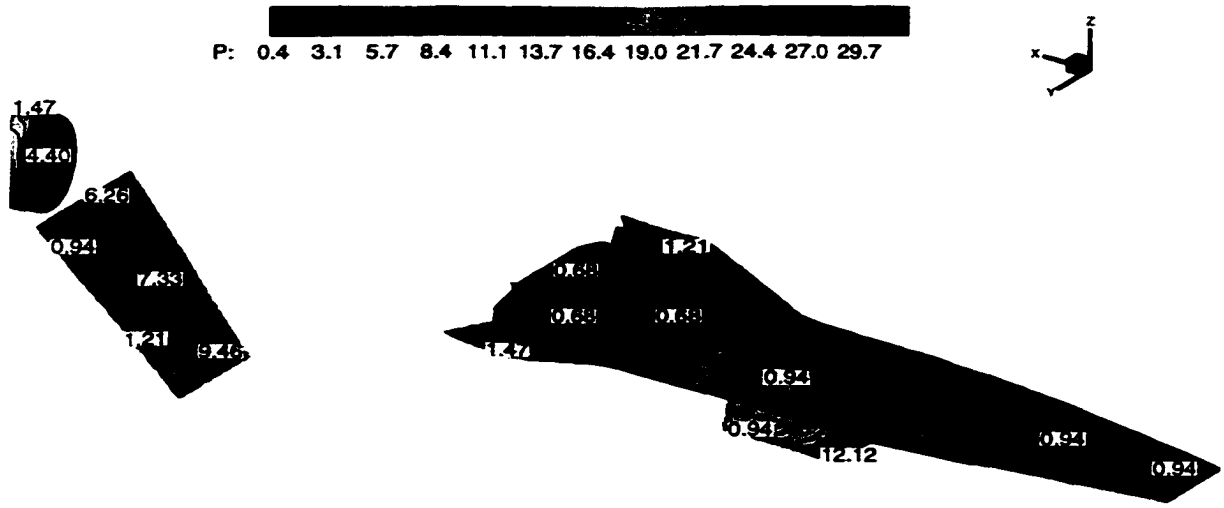


(c)

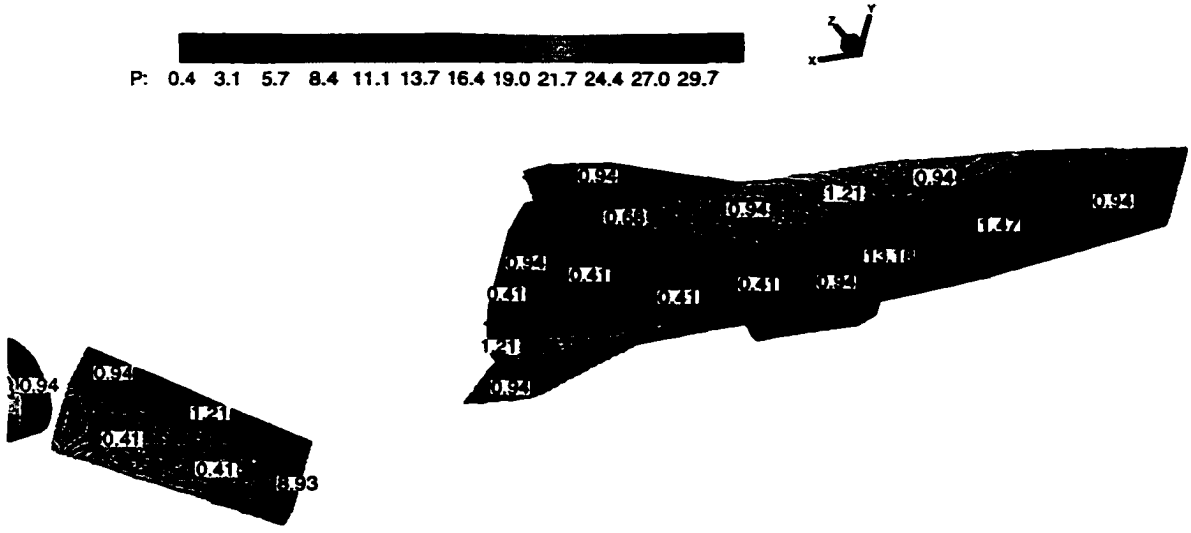


(d)

Figure 5.26 Continued.

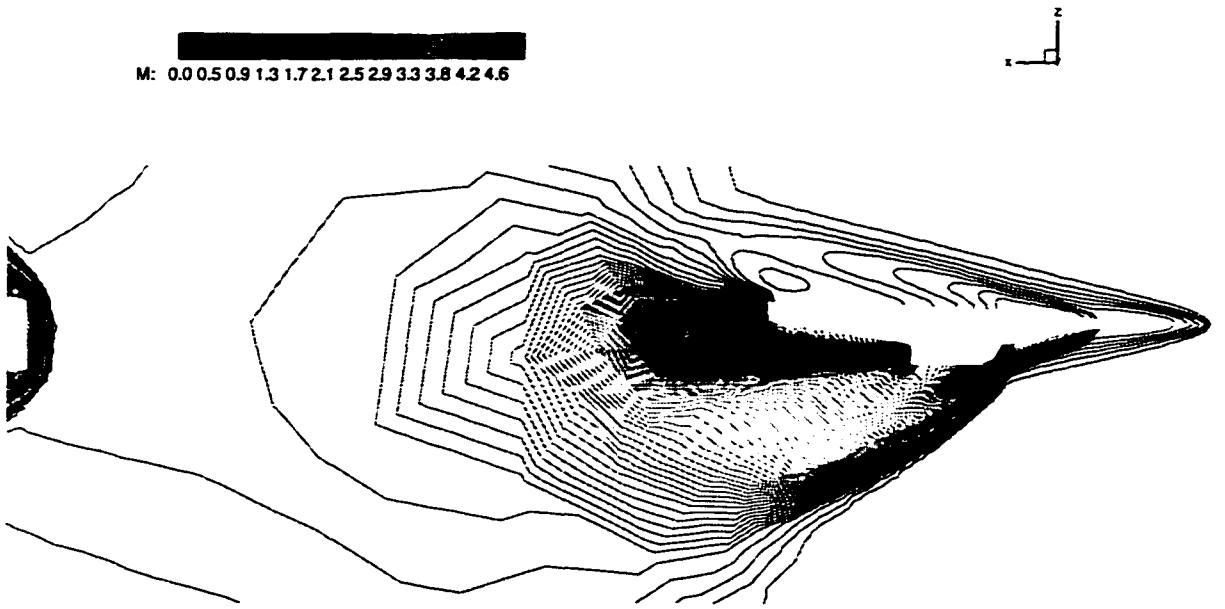


(e)

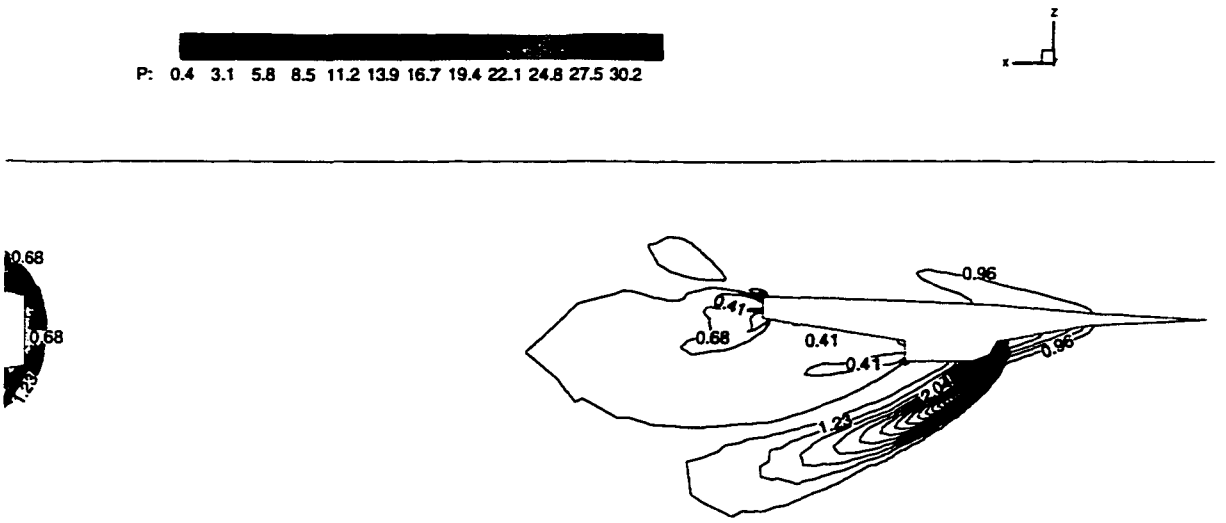


(f)

Figure 5.26 Concluded.



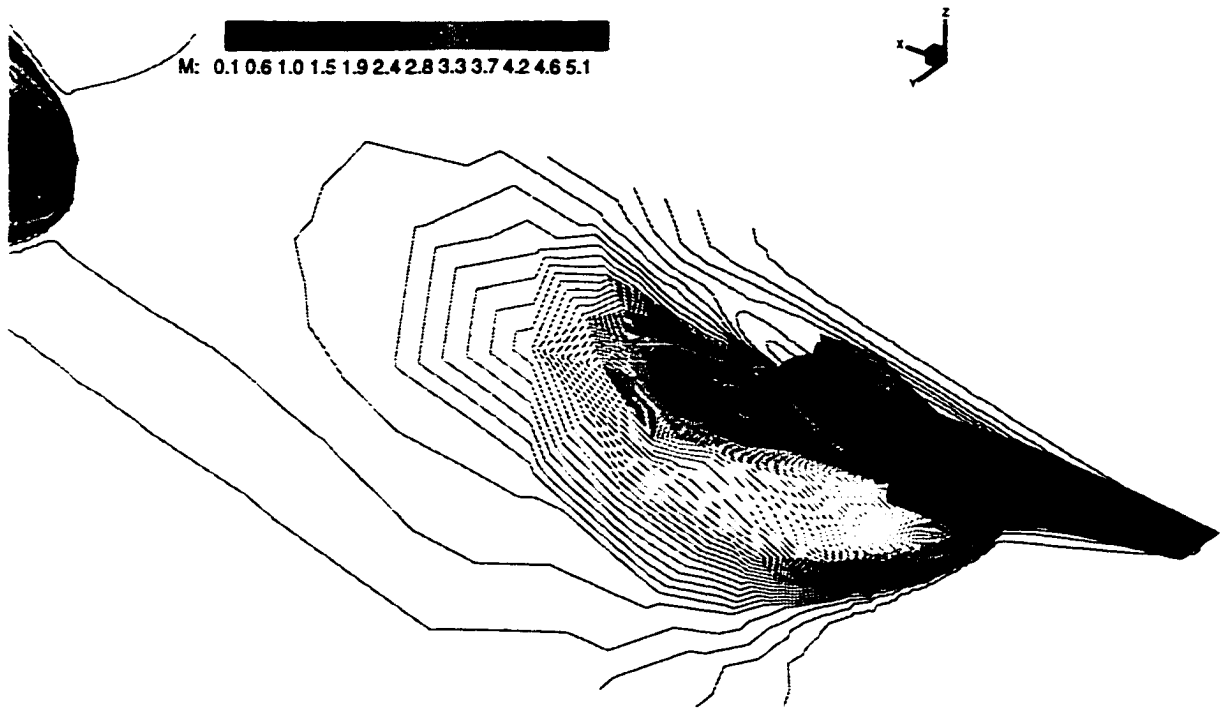
(a)



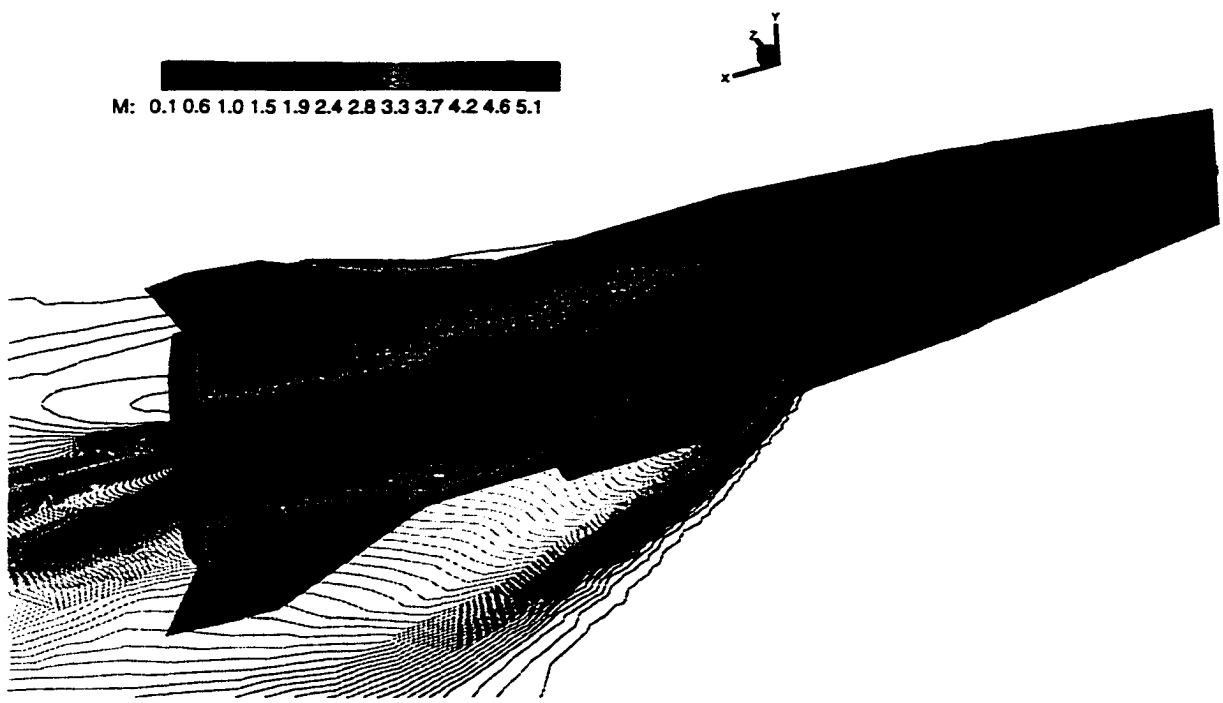
(b)

Figure 5.27 Instantaneous values for Case 5 during stage separation.

(a) Mach contours on the symmetry plane, (b) Pressure contours on the symmetry plane,
(c) Top view of Mach contours, (d) Bottom view of Mach contours,
(e) Top view of pressure contours, (f) Bottom view of pressure contours.



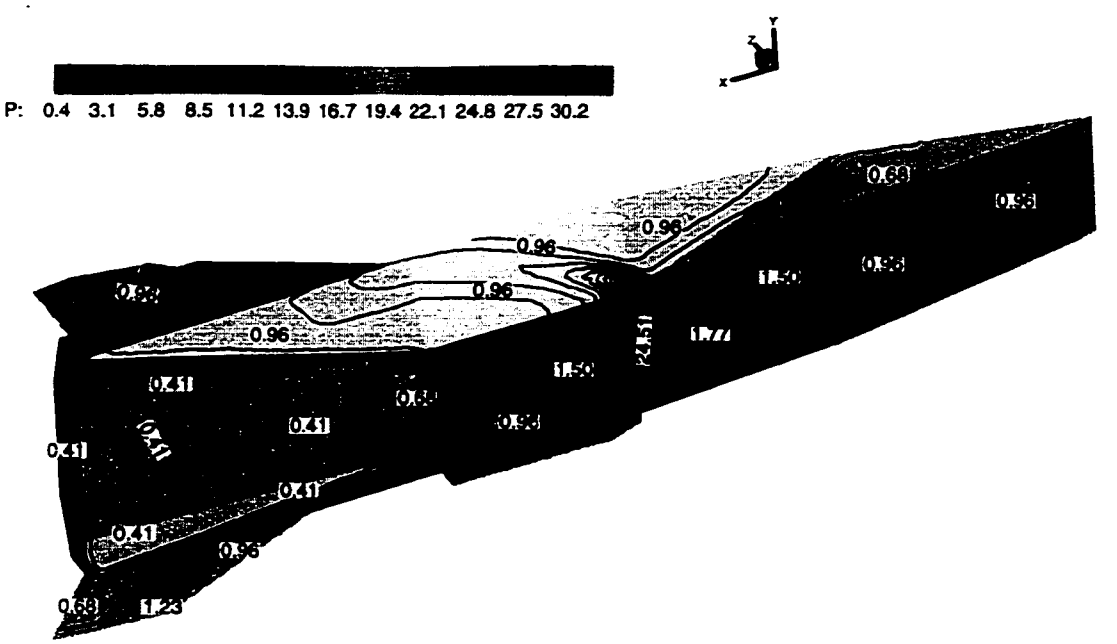
(c)



(d)

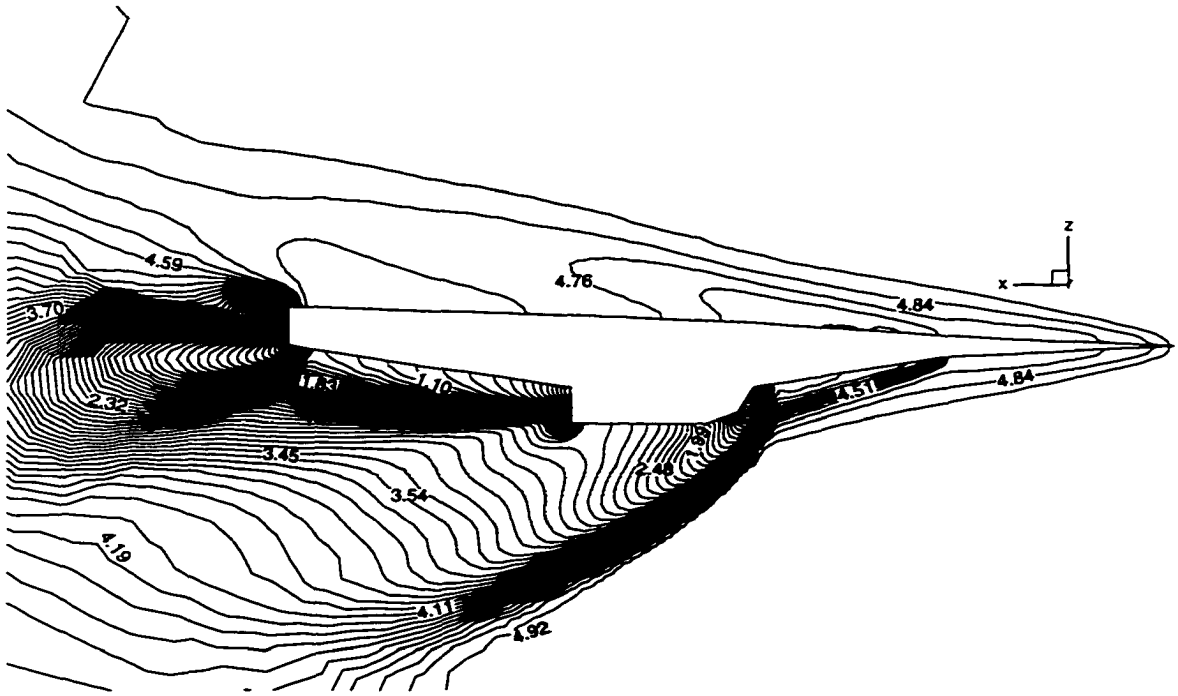
Figure 5.27 Continued.

(e)

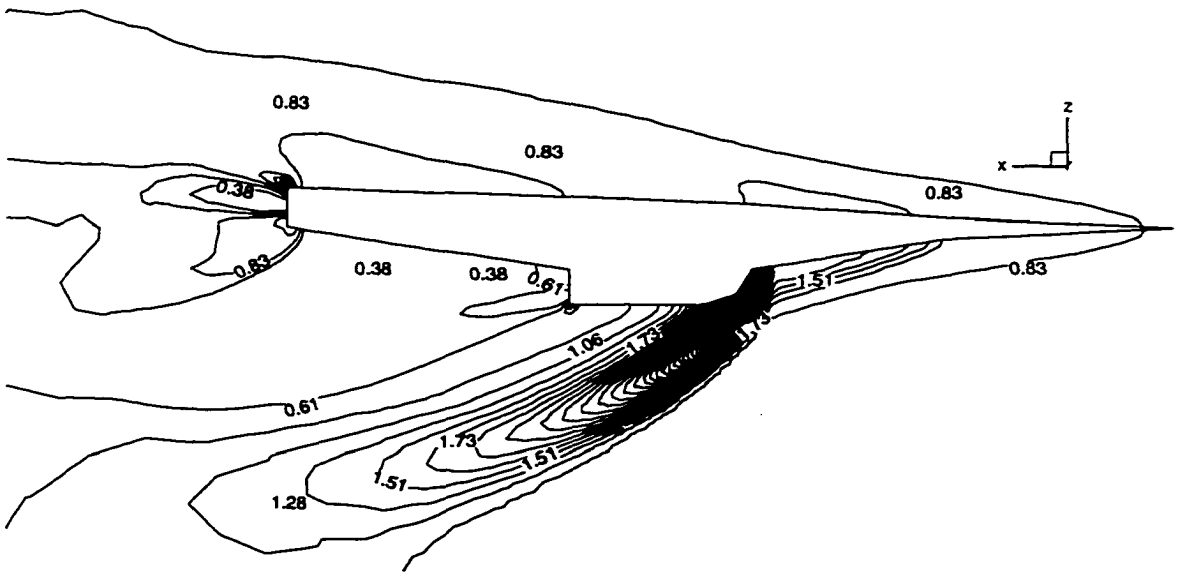


(f)

Figure 5.27 Concluded.



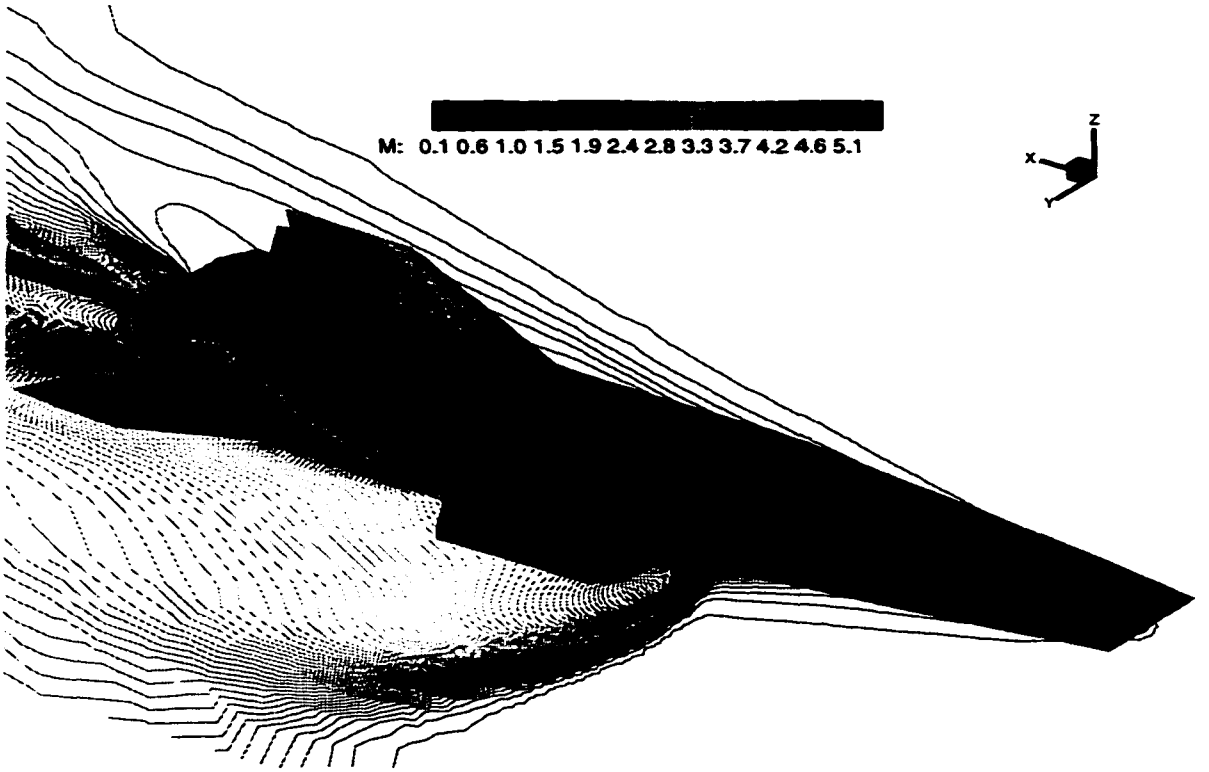
(a)



(b)

Figure 5.28 Instantaneous values for Case 6 during stage separation.

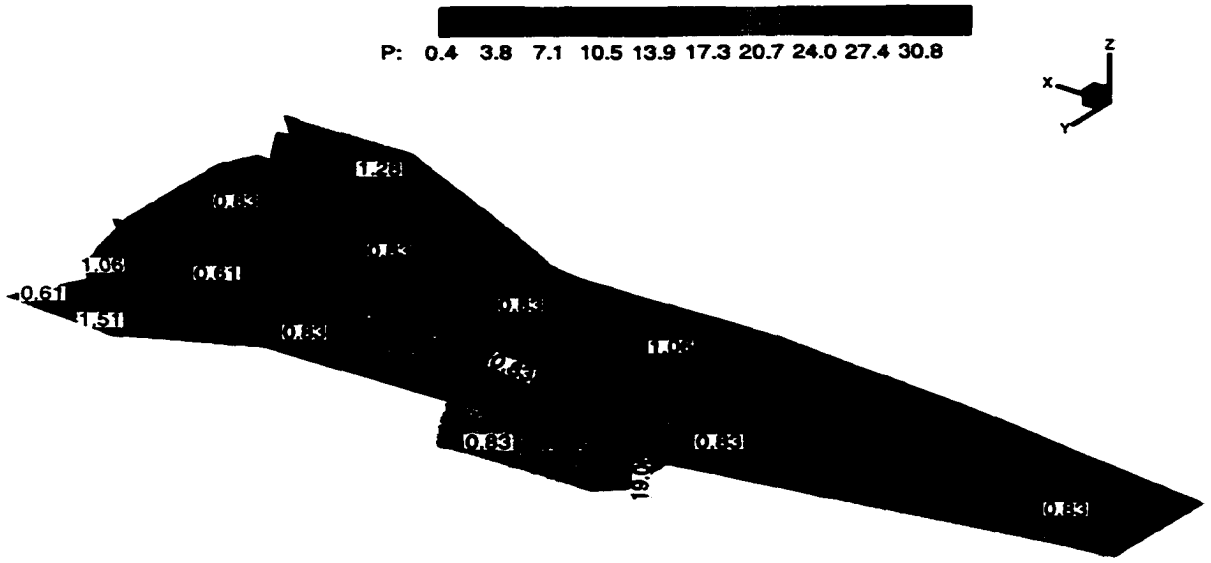
(a) Mach contours on the symmetry plane, (b) Pressure contours on the symmetry plane,
(c) Top view of Mach contours, (d) Bottom view of Mach contours,
(e) Top view of pressure contours, (f) Bottom view of pressure contours.



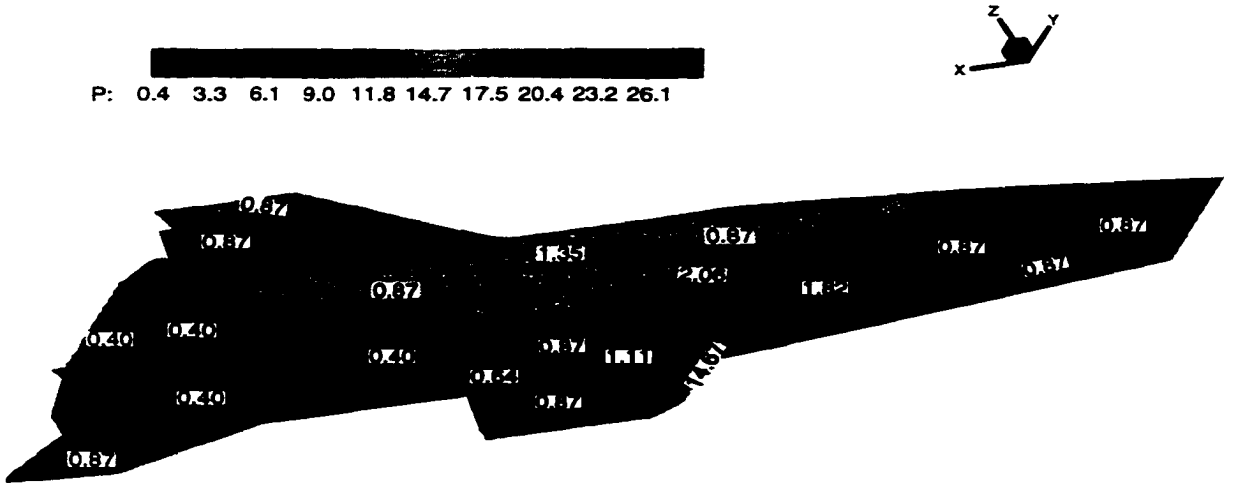
(c)

(d)

Figure 5.28 Continued.



(e)



(f)

Figure 5.28 Concluded.

In Case 1 (figure 5.23), the detached shocks from the jaw merge with those from the engine inlet and reattach the jaw again. Two discontinuous surfaces are formed in the base of the research vehicle, which are replaced by a series of expansion waves in free flight (Case 6). A large low-velocity and high-pressure area is formed at the nozzle of the research vehicle, mainly due to the existence of the jaw, which induces additional shocks, originating at the engine outlet and merging with the detached shocks from the jaw. With increased distance between the research vehicle and the booster in Case 2 (figure 5.24), the detached shocks from the jaw do not reattach to the jaw again but extend to the lower surface of the research vehicle. They will merge with the shocks from the engine inlet, resulting in a reduced low-velocity and high-pressure region. A discontinuity surface also appears in the gap between the booster and the research vehicle.

In Case 3 and Case 4 (figure 5.25 and 5.26), the jaw and the booster move further away from the research vehicle. The effects of the booster and the adapter appear to be confined to the wake flow of the research vehicle, and the detached shocks from the jaw and the research vehicle do not merge again. The discontinuity surface, shown in Case 2, is no longer apparent. In Case 5 (figure 5.27), even the wake flow of the research vehicle does not seem to be affected by the detached shocks from the booster. That is, there is no significant aerodynamic interference between the research vehicle and the booster. Therefore, it is concluded that the booster will have no influence on the flight of the research vehicle after it separates from the research vehicle by more than 1.7 times the vehicle length.

Present in Table 5.4 and figure 5.29 are the normal and axial force coefficients and the pitching moment coefficient. In an attempt to elucidate the decreasing interference effect as the research vehicle separates from the booster and the jaw, the relative departure of these coefficients from the values in free flight are tabulated in Table 5.5. The axial force in Case1 was negative, which was consistent with the results in 2-D computations (figure 5.18). With the decrease of low-velocity range on the inclined inboard surface of the research vehicle, the axial force becomes positive again in Case 2.

Table 5.4 Steady-state force and moment coefficients on the plane at difference relative locations.

	C_N	C_A	C_M
Case 1	2.950E-03	-1.287E-02	-7.132E-02
Case 2	1.550E-02	1.534E-02	-5.778E-02
Case 3	1.247E-02	1.464E-02	-3.939E-02
Case 4	1.218E-02	1.447E-02	-5.868E-02
Case 5	9.638E-03	1.186E-02	2.825E-02
Case 6	9.591E-03	1.180E-02	2.748E-02

Table 5.5 Relative departure of force and moment coefficients from free-flight (Case 6) values due to booster-adapter-vehicle interference.

	Case 1	Case 2	Case 3	Case 4	Case 5
$(C_L-C_{L_0})/C_{L_0}$	-69	62%	29%	27%	0.5%
$(C_D-C_{D_0})/C_{D_0}$	-209%	30%	24%	22%	0.48%
$(C_m-C_{m_0})/C_{m_0}$	-359%	-310%	-243%	-270%	2.8%

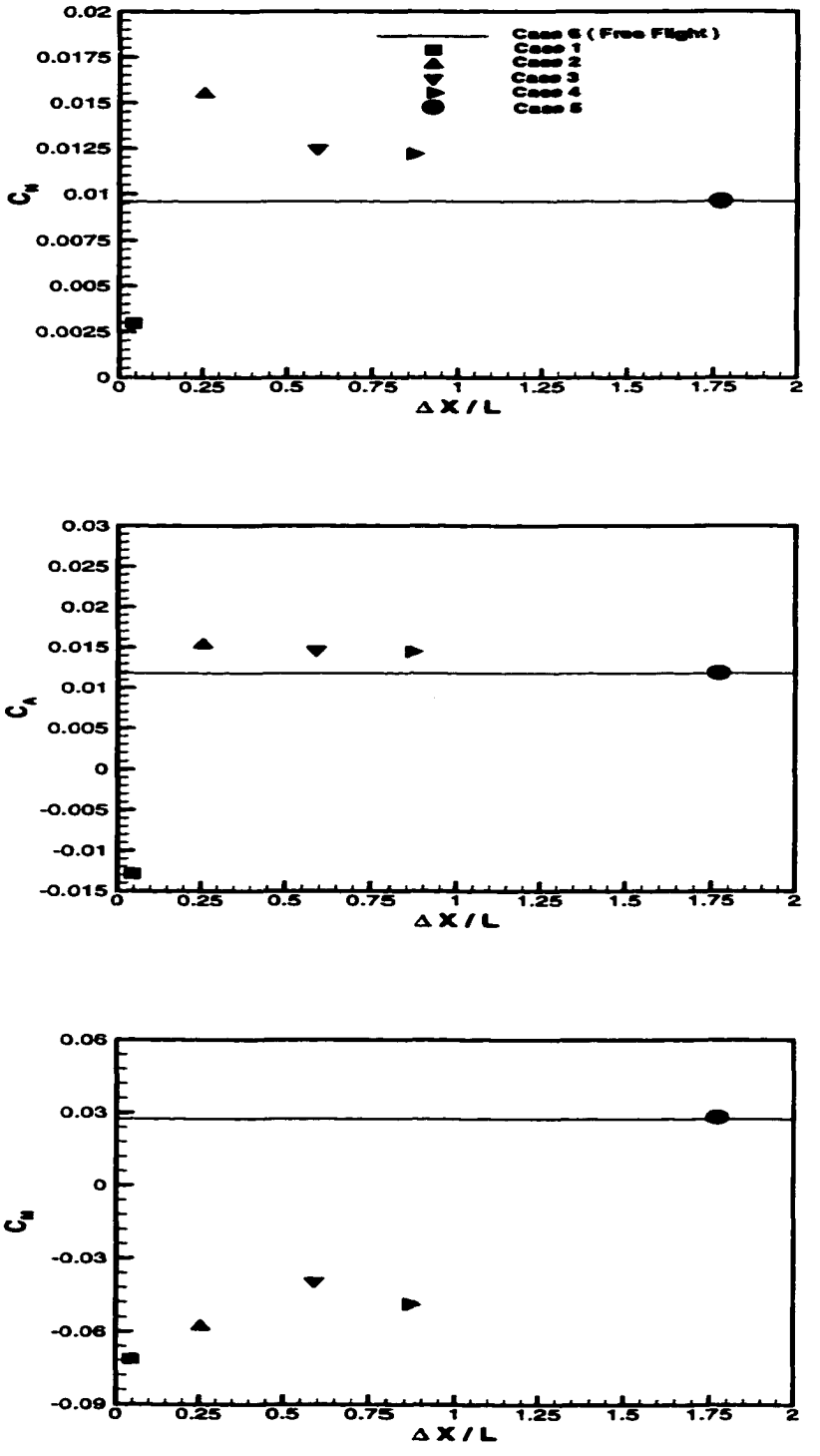


Figure 5.29 Steady state force and moment coefficients on the hypersonic plane at different relative locations.

It can be concluded from figure 5.29 that the history of aerodynamic forces is a nonlinear function in terms of ΔX, as the staging advances. This is again consistent with the 2D dynamic computation in Mach 5 flight, where the absolute value of negative axial force increases with ΔX. More importantly, it is observed that with the increase of ΔX during the stage process, the forces and moment have a trend to their values in free flight.

Although the normal and axial forces in Case 5 converge to values in free flight within 0.5%, there is still an apparent difference in the values of moment. The reason might be that the moment is associated with the pressure distributions, whereas forces only depend on their integration. Therefore, the moment is much more sensitive to the discrepancies among meshes. Note that the meshes are generated somewhat independently. As a result, the node distributions in Case 5 and Case 6 are not exactly the same.

5.3.2 Dynamic simulations

After obtaining the overall history of aerodynamic interference along the assumed staging path by the animation method, the dynamic-motion simulations were performed starting with Case 1 and Case 2, where the booster is extremely close to the research vehicle. The same two-degrees-of-freedom relative motion as that in two-dimensional simulations was imposed on the booster and the jaw. The static meshes in Case 1 and Case 2 were modified by the spring analogy method, and windows were built up around bodies. As a result, 42,263 nodes and 201,1844 tetrahedra fell within the created window in Case 1, and the corresponding numbers were 36,443 and 168,974 in Case 2. The explicit scheme was employed to advance in physical time domain. After numerical stability analysis, the time step for Case 1 and Case 2 was taken as 0.02, resulting in 11,00 iterations.

The histories of dynamic forces in Case 1 and Case 2 are presented in figure 5.30 and 5.31. The variations of normal force, axial force, and pitching moment are about 50%, 1.5%, and 18%, respectively, in Case 1. Whereas, they are about 2%, 0.3% and 3.5% in Case 2. It may be concluded that with the increased distance between the booster and the research vehicle, the aerodynamic interference is less sensitive to further relative motion of the jaw and the booster.

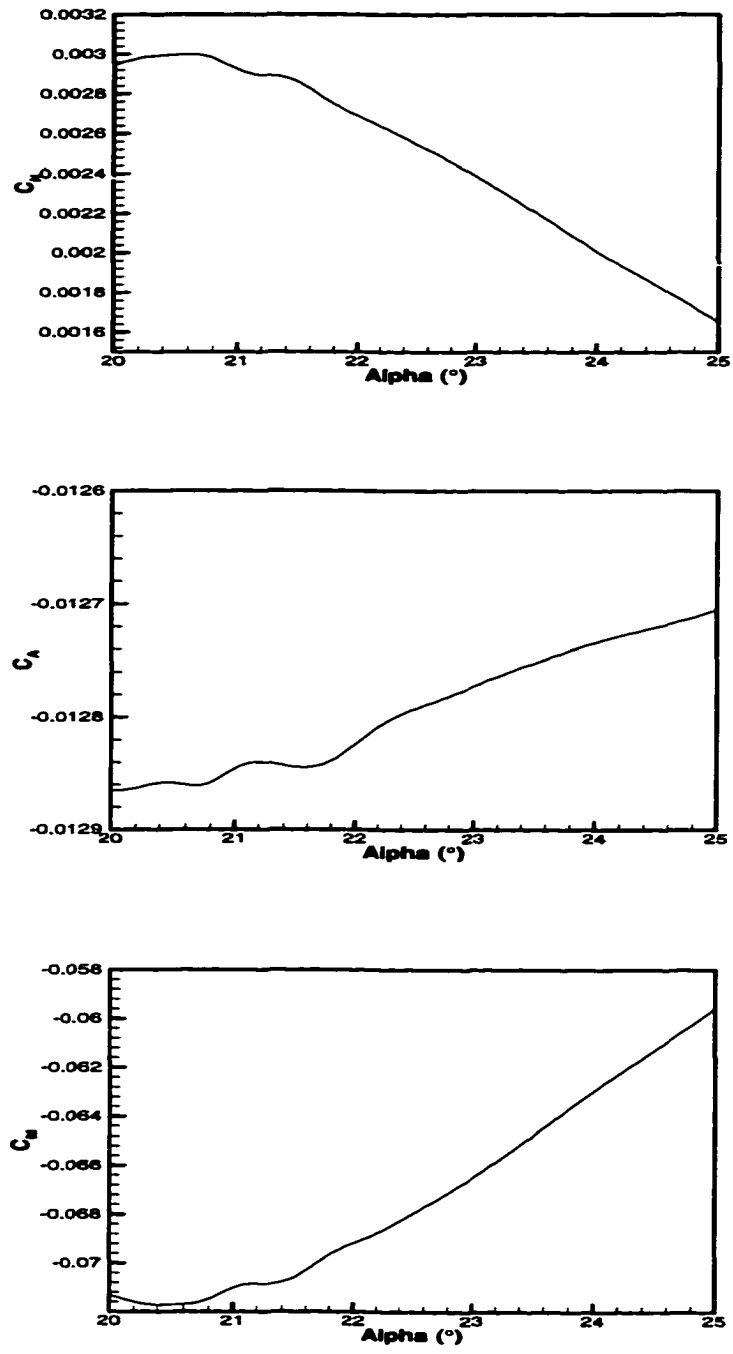


Figure 5.30 Histories of force and moment coefficients on the hypersonic plane in Case 1.

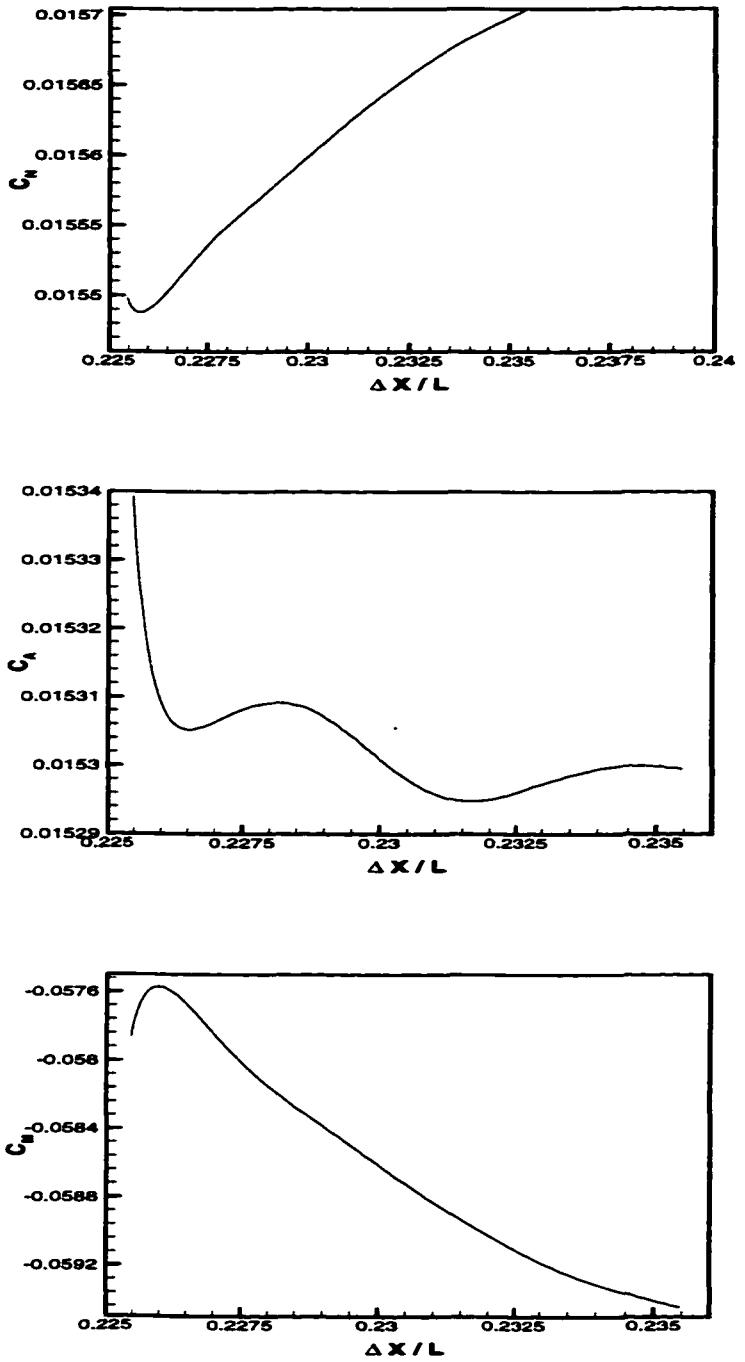


Figure 5.31 Histories of force and moment coefficients on the hypersonic plane in Case 2.

This chapter presents the simulations of hypersonic vehicle staging using dynamic unstructured methods. A network of strong shocks and expansion waves were captured. A prescribed two-degrees-of-freedom motion was imposed on the booster and the adapter to mimic the staging. The overall history of aerodynamic interference during the staging in Mach 5 and 10 flights was obtained. The following conclusions are made. The aerodynamic interference is a highly nonlinear function in terms of the distance between the vehicle, the booster and the adapter, and it is dependent on the Mach number of free stream. As the stage separation advances, the aerodynamic interference becomes less sensitive to further relative motions.

The conclusions from the current research will be presented in the next chapter. The recommendations for future work will also be included.

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

6.1 On Efficient Dynamic CFD Method

A fully implicit scheme, with the property of second-order accuracy in time domain and low storage requirement, has been developed for unstructured meshes. It was validated by simulating unsteady transonic flows around the pitching NACA0012 airfoil and a rectangular wing with a low-reduced frequency.

In the present computations of unsteady flows around the NACA0012 airfoil, the number of subiterations was fixed arbitrarily as 300. In the computation with the physical time step equal to one thousand times that of the explicit scheme, there was a small difference in the aerodynamic loading between the implicit scheme and the explicit scheme. This was credited to the quick drop of pseudo residuals. Consequently, several times CPU savings were achieved. When the computation was carried out with the physical time step equal to five thousand times that of the explicit scheme, more than one order of CPU savings was available, but a significant difference in the predicted aerodynamic loading between the two schemes existed. The reason was that subiterations were far from converged after the fixed number of iterations. Therefore, efficient accelerating techniques were desirable.

After using three levels of overset-type meshes, the convergence of subiterations in the fine mesh had obviously been improved, and pseudo residuals were dropped to the limit of the baseline solver within 300 iterations. As a result, approximately one order of

CPU savings was achieved, and the difference in the aerodynamic loading between the explicit and implicit scheme was minor.

For the rectangular wing case, subiterations converged much faster because the flow field had no apparent shocks, resulting in approximately one order of CPU savings without compromising the accuracy.

From computations, it is concluded that the efficiency of the dual-time stepping scheme depends highly on the convergence rate of subiterations, which has a close relationship with the size of the physical time steps. That is, the larger the time step is, the lower the convergence rate becomes. Certainly, whether subiterations are converged or not has direct influence on the accuracy of final solutions.

One of the main objectives in the present research, improving the efficiency of dynamic unstructured technique (DUT) for the problems with large time domain without compromising the accuracy, has been achieved. Therefore, it is concluded that the dual-time stepping scheme is an efficient approach to computationally predict the dynamic aerodynamic loading and the details of unsteady flow field at corresponding instants. It can also be concluded that more orders of CPU savings are expected in viscous dominated flows, where highly stretched meshes are required, and the limitation of numerical stability becomes worse.

6.2 On Dynamic Mesh Module

Two key modifications have been implemented to improve the quality of the dynamic mesh module, which is based on the spring analogy method. First, a Laplacian-

type presmoothing process was used to enhance the quality of node distribution on a dynamic mesh. Next, a successful effort was made to repair the mesh, if and when the mesh was tangled, by using an area correction based on minimizing a cost function while keeping the original mesh unchanged as much as possible. This is probably the best solution if the connectivity of meshes cannot be changed. A better solution to dynamic meshes with large-amplitude motion is to change the connectivity of meshes and insert or delete nodes continuously wherever necessary.

6.3 On Simulation of Hypersonic Vehicle Stage Separation on Symmetry Plane

A transonic time-accurate Euler solver was extended to simulate the stage separation of a hypersonic vehicle on the symmetry plane. The Hänel flux splitting and van Albada limiter were resorted to improve the numerical stability in hypersonic flows. These were deemed necessary to predict the strong shocks in hypersonic flows. Mach 5 and Mach 10 flights with dynamic pressure equal to $1{,}000\,lb/ft^2$ were performed, in which the booster and the adapter were extremely close to the vehicle.

Two-degrees-of-freedom motion was imposed on the adapter and the booster by assigning a translational motion on the adapter and the booster and superposing an additional rotational motion on the adapter. With these assumed motions, the region within which pressure varied with time was confined to only about 5% of the vehicle length on the vehicle upper surface and was extended to 20% on the lower surface. The trend of change of the overall aerodynamic forces in the Mach 5 and the Mach 10 flight was not the same. In the Mach 5 flight, with the adapter and the booster moving away

from the research vehicle along the assumed staging path, the aerodynamic interference increased, and did not decrease as was expected. It is concluded that both the flow field and the aerodynamic interference are dependent on the Mach number of free stream, and the aerodynamic interference should be a nonlinear function in terms of the distance between the research vehicle and the adapter and the booster.

6.4 On Simulation of Three-Dimensional Stage Separation of Hypersonic Vehicle

A three-dimensional static Euler solver, with an efficient edge-based data structure and a central difference scheme, was developed to simulate dynamic flow fields around a hypersonic vehicle during its stage separation. The overall history of the aerodynamic interference during its stage separation in the Mach 5 flight was obtained by an animation method, consisting of six static solutions where the relative motion was instantaneously frozen. By practice, almost one order of running memory was saved by the edge-based data structure, as compared to a cell-centered data structure for the same mesh.

As depicted by their Mach contours, the shape of shock systems went through significant changes when the adapter and the booster moved away from the research vehicle. The conclusion from 2-D computations, that the aerodynamic interference was a nonlinear function of the distance between the research vehicle and adapter and booster, was confirmed. An overall thrust was obtained when the booster and the adapter were extremely close to the research vehicle, which was consistent with the 2-D solution as

well. With the decrease of high-pressure area as the booster and the adapter moved further away, the overall drag was predicted again.

From these static solutions, the following conclusions were made. There would be no apparent shock interactions or any other direct effects on the booster and the adapter after they moved away from the research vehicle by more than 60% of the vehicle length. However, aerodynamic forces still had more than 20% departure from their values in free flight due to the interference through the wake flow. The variations in aerodynamic forces were much more rigorous in Case 1 and Case 2 than those in Case 3 and Case 4. Nonetheless, the aerodynamic forces converged to the values in free flight after the booster was away from the vehicle by 1.77 times the vehicle length (Case 5).

The dynamic computations were performed in Case 1 and Case 2, and the same relative motions as 2-D computations were assigned on the adapter and the booster. In Case 1, the amplitudes of variations on lift, drag and moment were about 50%, 1.5% and 18%, respectively. In Case 2, they were 2%, 0.3%, and 3.5%. It may be concluded that with the increased distance between the research vehicle, booster and adapter, the aerodynamic interference was less sensitive to further relative motions.

6.5 Recommendations for Future Work

The dynamic unstructured technique (DUT) has proven to be a very powerful approach to simulate unsteady flows for moving boundary problems when it is engaged in relative motions. Although its efficiency has been improved greatly by the implicit

scheme, there is still room to extend and strengthen its capabilities. A number of outstanding issues for future work are listed as follows:

1. With a robust dynamic mesh module where the connectivity of unstructured meshes is dynamic, the time domain in the simulation of hypersonic vehicle staging may be very large. Then, the efficiency of computations will become a key issue, and the dual-time stepping scheme will be an ideal choice. However, the convergence of subiterations has to be accelerated by multigrid methods because the flow field will be dominated by nonlinear phenomena. For hypersonic flows with strong shocks, using bilinear functions to construct the restriction and prolongation operators may not be accurate enough; therefore, higher order polynomials are suggested.
2. Although a central differencing scheme promises a stable solution by adding sufficient artificial dissipation, based on the features of flow fields, too much artificial dissipation will incur a solution with poor quality in smooth regions. Hence, further research to derive an upwind scheme with sufficient dissipation, which works for the three-dimensional simulations of hypersonic vehicle stage separation, is recommended. Also, it is recommended to compare the solutions by the central differencing scheme with those by upwind schemes.

3. A local remeshing routine should be added to dynamic mesh module. For large-amplitude moving boundary problems, the present spring analogy method usually results in edge crossing or meshes with poor quality. This can easily degrade the solution or cause a fatal error in running codes. The best approach to avoid the edge

crossings is to take advantage of a dynamic connectivity for the mesh nodes. By implementing a local remeshing algorithm, the nodes can be inserted or deleted automatically when and where necessary.

4. For the hypersonic flows in relatively small regions, accounting for the viscous effects is essential. With the improvements in the grid generation technology and computer power, three-dimensional, unsteady, viscous computations on highly stretched unstructured meshes are becoming possible. Including the viscous terms will, therefore, extend the applications using DUT and improve its accuracy for relative-motion problems.

5. The dynamic loading on a flexible structure induces such complex phenomena as the flutter of rotating blades, the buffet of tails, and the force response of blades. Replacing the prescribed motions of components, by the displacements of a flexible structure, computed using structural dynamic theory, would extend DUT to handle aeroelastic problems.

6. Another measurement, probably the best choice to improve the efficiency of simulating moving boundary problems, is the implementation of a parallel-computing algorithm. There are parallel computers now available to the aerodynamics community, but some preprocessing of the computer codes is still necessary.

LIST OF REFERENCES

1. Singh, K. P., and Baysal, O., "3-D Unstructured Method for Flow Past Bodies in 6-DOF Relative Motion," Proceedings of Sixth the Internationls Symposium of Computational Fluid Dynamics, Lake Tahoe, NV, September 1995.
2. Singh, K. P., Newman, J. C., Baysal, O., "Dynamic Unstructured Method for Flows Past Multiple Objects in Relative Motion," *AIAA Journal,* Vol. 33, No.4, 1995, pp. 641-649.
3. Singh, K. P., Dissertation for Ph.D, "Dynamic Unstructured Method for Prescribed and Aerodynamically Determined Relative Moving Boundary Problems," Old Dominion University, Norfolk, VA, August 1995.
4. Bibb, K. L., Peraire, J., and Riley, C. J., "Hypersonic Flow Computations on Unstructured Meshes," AIAA 97-0625, 35th Aerospace Sciences Meeting & exhibit Reno, NV, January 1997.
5. Yen, G. W and Baysal, O., "Effects of Efficiency Techniques or Accuracy of Dynamic Overlapped Grids for Unsteady Flows," *Journal of Fluids Engineering*, Vol. 119, No. 23, pp. 577-583.
6. Takayama, M. S., Timfeev, E. V., and Voinovich, P.A., "A Moving Boundary Technique for 2-D Locally Adaptive Unstructured Euler Solve," Lecture note.
7. Mavriplis, D., and Jameson, A. "Multigrid Solution of The Euler Equations on Unstructured and Adaptive Meshes," ICASE Report, No. 87-53, NASA Langley Research Center, Hampton, VA, 1987.

8. Anderson, W. K, Rausch, R. D., and Bonhaus, D, "Implicit/Multigrid Algorithms for Incompressible Turbulent Flows on Unstructured Grids," *Journal of Computational Physics*, Vol. 128, No. 0219, pp. 391-408, 1996.

9. Pirzadeh, S. "Structured Background Grids for Generation of Unstructured Grids by Advancing-Front Method," *AIAA Journal*, Vol. 31, No.2, pp. 257-265, 1993.

10. Wood, W.A. and Kleb, W. L., "Diffusion Characteristics of Upwind Schemes on Unstructured Triangulations", AIAA 98-2443, 29th AIAA Fluid Dynamics Conference, Albuquerque, NM, June 1998.

11. Mavriplis, D. J., "Adaptive Meshing Techniques for Viscous Flow Calculations onMixed Element Unstructured Meshes," ICASE Report No. 97-20, NASA Langley Research Center, Hampton, VA, 1997.

12. Peraire, J., Peiro, J., and Morgan, K., "Multigrid Solutions of the 3D Compressible Euler Equations on Unstructured Tetrahedral Grids," *International Journal for Numerical Methods in Engineering*, Vol. 36, pp.1029-1044, 1993.

13. Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," AIAA-91-0725, January 1991.

14. Luo, X., and Baysal, O., "Dynamic CFD Simulation of Hypersonic Vehicle Separation From its Booster", AIAA-99-4807, 9th International Space Planes and Hypersonic Systems and Technologies Conference, Norfolk, VA, November 1999.

15. Baysal, O. and X. Luo, "Staging of a Hypersonic Vehicle: Numerical Simulations Using Dynamic Unstructured Meshes", AIAA-98-2412, 16th AIAA Applied Aerodynamics Conference, Albuquerque, NM, June 1998.

16. Ollivier, C. F., "On Mesh to Mesh Transfer Operators for Unstructured Multigrid Solvers," 13th Computational Fluid Dynamics Conference, Snow-mass Village, CO, June/July 1997.

17. Cvrljie, T., "Unsteady Separation of a Two Stage Hypersonic Vehicle", 30th AIAA Fluid Dynamics Conference, Norfolk, VA, June/July 1999.

18. Batina, J. T., "A Fast Implicit Upwind Solution Algorithm for Three-Dimensional Unstructured Dynamic Meshes," 30th Aerospace Sciences Meeting & Exhibit, Reno, NV, January 1992.

19. Noack, P. W., and Bishop, D. G., "A 3-D Delaunay Grid Generator and Flow Solver for Bodies in Relative Motion," 11th AIAA CFD Conference, Orlando, FL.

20. Löhner, R., Yang, C., Baum, J. D, "Rigid and Flexible Store Separation Simulations Using Dynamic Adaptive Unstructured Grid Technology," First AFOSR Conference on Dynamic Motion CFD, Rutgers, NJ, June 1996.

21. Löhner, R., "Three-Dimensional Fluid-Structure Interaction Using a Finite Element Solve and Adaptive Remeshing," *Computer Systems in Engineering* 1, 2-4, pp. 254-272, 1990.

22. Liu, M. S, "An Extended Lagrangian Method," AIAA 93-3305 CP, July 1993.

23. Baysal, O., and Yen, G. W., "Kinematic Domain Decomposition to Simulate Flows Past Moving Bodies," AIAA 91-0725, 29th Aerospace Science Meeting, Reno, NV, January 1991,

24. Yen, G. W, and Baysal, O., "Computing Unsteady High Speed Flows Past an Ocillating Cylinder Near a Vertical Wall," *Journal of Spacecraft and Rockets,* Vol. 31, No. 4, pp. 630-635, July/August 1994.

25. Meakin, R. L., "Computations of the Unsteady Flow about a Generic Wing/Plyon. Finned Store Configuration," AIAA 92-4569 CP, Applied Aerodynamics Conference, Hilton Head, SC, August 1992.

26. Goswam, A., "Grid Restructuring for Moving Boundaries," AIAA-91-158 CP, 1991.

27. Batina, J. T., "Implicit Flux-Split Euler Scheme for Unsteady Aerodynamic Analysis Involving Unstructured Dynamic Meshes," *AIAA Journal*, Vol. 29, No.53, pp. 1836-1843, 1991.

28. Palmerio, B., "An Attraction Repulsion Mesh Adaption Model for Flow Solution on Unstructured Grids," *Journal of Computers and Fluids*, pp.487-506, 1994.

29. Kennon, S. R., Meyering, J.M., Berry, C.W., Oden, J.T., "Geometry Based Delaunay Tetrahedralization and Mesh Movement Strategies for Multi-Body CFD," AIAA Paper 92-4575, AIAA Atmospheric Flight Mechanics Conference, Hilton Island, SC, August 1992.

30. Connel, S. D. and Holmes, D.G, "A 3D Unstructured Adaptive Multigrid Scheme for Euler Equations, " AIAA Journal, Vol.32, pp.956-963, 1994.

31. Baysal, O., and Luo., X., "Computational Aeromechanics Method with Multigrid Accelerated Dual-Time Stepping on Unstructured Meshes," ASME, Paper FEDSM 98-4939, Fluids Engineering Division Summer Meeting, Washington, D.C, June 1998.

32. Guillar, H., "Node Nested Multigrid with Delaunay Coarsening," INRIA Report, No.1898, 1993.

33. Venkatakrishnan, V., and Mavriplis, D. J., "Agglomeration Multigrid for the Three Dimensional Euler Equations," AIAA Paper 94-0069, 1994.

34. Venkatakrishnan, V., and Jameson, A., "Computation of Unsteady Transonic Flows the Solution of Euler Equations," *AIAA Journal*, Vol. 26, pp. 974-981, 1988.

35. Yen, G. W., and Baysal, O., "Accuracy of Dynamic Overlapped Grids for Unsteady Flows", Proceedings of Unsteady Flows, ASME, Hilton Head, SC, August 1995.

36. Gaitonde, A. L., Fidds, S. P., "A Three Dimensional Moving Mesh System for the Calculation of Unsteady Transonic Flow," *Aeronautical Journal*, Vol. 99, No. 984, 1995.

37. Steger, J. L., "Implicit Finite-Difference Simulation of Flow About Arbitrary Two Dimensional Geometry," *AIAA Journal*, Vol. 16, No. 7, pp. 679-686, 1978.

38. Rumsey, C. R., Sanetrik, M. D., Biedron, R. T., "Efficiency and Accuracy of Time Accurate Turbulent Navier-Stokes Computations," AIAA 95-1835-CP.

39. Frink, N. T., "Recent Progress Toward a Three Dimensional Unstructured Navier-Stokes Flow Solver," AIAA-94-0061, 32nd Aerospace Sciences Meeting, Reno, NV, January 1994.

40. Venkatakrishnan, V., and Mavriplis, D. J., "Implicit Solvers for Unstructure Meshes," ICASE Report, No. 91-40, May 1991.

41. Jameson, A., "Time-Dependent Calculations Using Multigrid with Applications to Unsteady Flows Past Airfoils and Wings", AIAA Paper 91-1596, July 1991.

42. Venkatakrishnan, V. and Mavriplis, D. J, "Implicit Method for the Computation of Unsteady Flows on Unstructured Grids," *Journal of Computational Physics*, Vol.127, No. 0182, pp. 380-397, 1996.

43. Liu, F., and Ji, S., "Unsteady Flow Calculations with a Multigrid Navier-Stokes Method," *AIAA Journal*, Vol. 34, No. 10, October 1996.

44. Frink, N. T., "Assessment of an Unstructured-Grid Method for Predicting 3-D Turbulent Viscous Flows", AIAA-96-0292, 34th Aerospace Sciences Meeting, Reno, NV, January 1996.

45. Campbell, C. H., Caram, J. M, and Berry, J. C, "Overview of X-38 Hypersonic Aeroth-ermodynamic Wind Tunnel Data and Comparison with Numerical Results," 32nd Thermophysics Conference, AIAA 97-2473, Atlanta, Georgia, June 1997.

46. Frendi, A., "On the CFD Support for the Hyper-X Aerodynamic Database," AIAA 99-0885, 37th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1999.

47. Pamadi, B., and Brauckman, G. J., "Aerodynamic Characteristics and Development of the X-34 Reusable Launch Vehicle," International Symposium on Atmospheric Reentry Vehicles and Systems, Arcachon, France, March 1999.

48. Mabey , D. G., Welsh, B. L., and Pyne, C. R., "A summary of Measurements of Steady and Oscillating Pressures on a Rectangular Wing," *Aeronautical Journal*, Vol. 92, No.911, pp. 10-28, January 1988.

49. Barth, T., "Upwind Euler Solver for Unstructured Meshes," AIAA-91-1548 CP.

50. Tamura, Y., and Fujii, K., "Conservation Law for Moving and Transformed Grid", AIAA-93-3365, July 1993.

51. Hackbush, W., "*Multigrid Methods and Applications*", Springer-Verlag, Berlin, Germany, 1985.

52. Stokes, S., Chappell, J. A., and Leatham, M., "Efficient Numerical Store Trajectory Prediction for Complex Aircraft/Store Configurations," 30th AIAA Fluid Dynamics Conference, Norfolk, VA, June 1999.

53. Pirzadeh, S. "Recent Progress in Unstructured Grid Generation," AIAA-92-0445, 30th Aerospace Sciences Meeting & Exhibit, Reno, NV, January 1992.

54. Landon, R., "NACA 0012 Oscillatory and transient Pitching," Compendium of Unsteady Aerodynamic Measurements, AGARD Report No. 702, pp. 3.3-3.25, 1982.

55. Voland, R. T, Rock, K. E, Huebner, L. D, "Hyper-X Engine Design and Ground Test Program," AIAA 8th International Space Planes and Hypersonic Systems and Technologies Conference, Norfolk, VA, April 1998.

56. Reubush, D. E., "Hyper-X Stage Separation: Background and Status," AIAA 99-4818. AIAA 9th International Space Planes and Hypersonic Systems and Technologies Conference, Norfolk, VA, November 1999.

VITA

Xiaobing Luo was born in NeiJiang, People's Republic of China on March 6, 1966. After finishing his high school in 1983, he entered the Aircraft Design and Applied Mechanics Department of Beijing University of Aeronautics and Astronautics (BUAA). He finished his Bachelor's Degree in aerodynamics in 1987 and Master's Degree in computational mechanics in 1990. After leaving BUAA, he entered the Institute of Engineering Development and Design of ChengDu Aircraft Inc. as an applied fluid engineer. In 1996, he joined the PhD program of the Aerospace Engineering Department of Old Dominion University, Norfolk, VA and finished his doctorate in 1999.