MCAT Institute
Progress Report
92-009.

P - 42

Numerical Methods for the Simulation of Complex Multi-Body Flows with Applications for the Integrated Space Shuttle Vehicle

William M. Chan

April 1992

NCC2-654

MCAT Institute
3933 Blue Gum Drive
San Jose, CA 95127

Numerical Methods for the Simulation of Complex Multi-Body Flows with Applications for the Integrated Space Shuttle Vehicle

William M. Chan

Introduction

This project forms part of the long term computational effort to simulate the time dependent flow over the integrated Space Shuttle vehicle (orbiter, solid rocket boosters (SRBs), external tank (ET) and attach hardware) during its ascent mode for various nominal and abort flight conditions. Due to the limitations of experimental data such as wind tunnel wall effects and the difficulty of safely obtaining valid flight data, numerical simulations are undertaken to supplement the existing data base. This data can then be used to predict the aerodynamic behavior over a wide range of flight conditions. Existing computational results [1] show relatively good overall comparison with experiments but further refinement is required to reduce numerical errors and to obtain finer agreements over a larger parameter space.

One of the important goals of this project is to obtain better comparisons between numerical simulations and experiments. In the simulations performed so far, the geometry has been simplified in various ways to reduce the complexity so that useful results can be obtained in a reasonable time frame due to limitations in computer resources. In this project, the finer details of the major components of the Shuttle will be better modelled by including more complexity in the geometry definition. Smaller components not included in early Shuttle simulations will now be modelled and gridded.

A major obstacle in the computational effort is the difficulty in generating satisfactory three-dimensional grids to represent the complex geometries of the integrated Space Shuttle vehicle. The body surface grid is usually supplied from algebraic or elliptic grid generation packages. The 3D volume grid is most efficiently obtained from a hyperbolic grid generation code. In the past two years, the hyperbolic grid generation code has been modified significantly to achieve a more modular structure. The algorithm has also been improved to provide robustness for a wide range of different geometries.

An important tool for numerical simulation is the flow solver. The F3D code uses an implicit approximately factored scheme for the 3D thin-layer compressible Navier Stokes equations with the Baldwin-Lomax algebraic eddy viscosity model. Each time step requires the inversion of two implicit block factors which can be quite an expensive procedure when only a steady state solution is required. In order to speed up the solution process, the diagonal algorithm and dissipation model in ARC3D have been implemented into the new flow solver OVERFLOW [2].

Algorithm and Code Development

Three-Dimensional Hyperbolic Grid Generation - HYPGEN

The purpose of the HYPGEN code is to generate a 3D volume grid over a user-supplied single-block surface grid. This is accomplished by solving the 3D hyperbolic grid generation equations (two orthogonality relations and one cell volume constraint). A 2D grid can also be generated by specifying appropriate boundary conditions.

The code is evolved from an old version (HYG3D) by Steger and Rizk [3]. In previous versions of the code, special care and parameter adjustments were frequently necessary in order to cope with a new geometry that is significantly different from ones treated before. The current version of the code is much more modular and many new techniques have been implemented to improve the robustness of the code (see Appendix A for details of the enhanced algorithms). Since a marching scheme is used in the grid generation process, the hyperbolic scheme in the code is about one to two orders of magnitude faster than typical elliptic methods (a volume grid with a hundred thousand grid points can be generated by HYPGEN in about one CPU second on the CRAY-YMP).

Constant user support has been provided for the code since its first release (Version 1.0) in January 1991. A user guide and a set of examples are provided with the release. Over the last year or so, the code has been used by various groups at Ames (e.g., Powered Lift, High Alpha, Incompressible Navier-Stokes groups in RFA) and it has also been distributed to several groups in industry and university (e.g., Boeing, Rockwell International, Sandia National Labs, Gulf Stream, University of California at Davis, Penn State University).

Hyperbolic Surface Grid Generation - SURGRD

Various tools have been developed for surface grid generation and communication used in the Chimera overset grid method. The hyperbolic surface grid generator (SURGRD) was modified extensively and various new capabilities have been added to the code including a set of general floating boundary conditions which allows good control of side boundaries. This code has mainly been used in the generation of collar grids for intersecting geometric components (see Appendix B). A grid projection tool (PROGRD) was also developed to perform appropriate projections between overlapped surface grids such that proper inter-grid communication is maintained in the overlapped regions.

Flow Solver - OVERFLOW

The diagonal algorithm of Pulliam and Chaussee [4] used in the Ames code ARC3D was implemented into the Chimera overlapped grid Navier Stokes flow solver OVERFLOW [2]. The artificial dissipation model in ARC3D was also incorporated. Some optimizations were performed to improve the efficiency of the code. The resulting scheme achieved a three-fold increase in speed over the old F3D algorithm.

Other major code development tasks for OVERFLOW included the implementation of a forces and moments subroutine, an investigation on the symmetry error caused by the diagonal algorithm and the coding for the fortified Navier Stokes scheme [5]. An unofficial copy of the code with the fortified Navier Stokes scheme has been used by the RFA Rotorcraft CFD group to simulate an actuator disk. A boundary layer option for the scheme was also implemented. It was expected to increase the convergence of the code but initial studies did not show much improvement.

Applications

The diagonal scheme option in the flow solver OVERFLOW was tested on a Space Shuttle vehicle launch configuration (9 grids consisting of External Tank, Solid Rocket Boosters, Orbiter and attach hardware). The flow simulation was performed at a Mach number of 1.25, angle of attack of -5.1 degrees and a Reynolds number based on wind tunnel conditions. Very good comparisons with wind tunnel data were obtained. After convergence was attained, the simulation was continued further in time accurate mode to capture the behavior through one cycle of oscillation in the lift coefficient. Other flow simulations using the diagonal scheme in OVERFLOW are described in Appendix C.

As an application for the hyperbolic surface grid generator SURGRD, collar grids for intersecting geometric components were used as test cases. In particular, the collar grid for the orbiter-vertical tail junction was generated using some of the new features in boundary conditions implemented in SURGRD (see Appendix B).

The geometry definition and surface grid generation for the Space Shuttle Orbiter has been made a much less formidable task by the availability of the CAD/grid geneneration software package ICEM. More complex geometric features can now be incorporated into the definition with the CAD capability. Surface grids can then be generated over the new improved geometry.

The surface grids for the back region of the Orbiter were generated using the ICEM software. First, a blocking strategy was drafted and then grids were generated. Several iterations of modifying the blocking strategy were made to eliminate certain deficiencies in the resulting grids. The volume grids were then generated using HYPGEN. The resulting grid system includes 12 new grids for the back region: body flap, body flap side wall, right main engine, symmetry grid for right main engine, out-board side grid for right main engine, top main engine, OMS extension and its collar grid, RCS jet block and its collar grid, and exit caps grids for the two engine grids. The inter-grid communication for the Chimera overlapped grid scheme was then arranged for an Orbiter alone case in an 18-grid system; consisting of the 12 new grids and 6 other grids for the rest of the Orbiter flow field. The PEGASUS code was run in increasingly complex configurations until all 18 grids

were communicating with each other. A flow simulation for an Orbiter alone case with the new grid system is in progress.

Concluding Remarks

The hyperbolic grid generation algorithm was enhanced to increase its robustness and efficiency. A resulting code HYPGEN was developed and has been distributed to various CFD groups. Future plans for the code include more automatic selection of certain grid quality parameters such that less user knowledge and input is required. Grid tools were also enhanced and developed for collar grid generation (SURGRD) and communication for overlapped surface grids (PROGRD).

The diagonal scheme and dissipation model of ARC3D has been implemented into the overset-grid flow solver OVERFLOW. Due to its speed, the diagonal scheme option has been used for all recent steady-state calculations of the integrated Space Shuttle vehicle flow field, both within the Multiple-Body Aerodynamics group in RFA and the Shuttle group at NASA Johnson.

Improvements over the existing geometric complexity used in flow simulations for the Orbiter has started by the generation of 12 new surface grids for the back region. Future plans will include better modelling of the entire Orbiter geometry using the ICEM CAD and grid generation package.

References

1. Buning, P.G., Chiu, I.T., Obayashi, S., Rizk, Y.M. and Steger, J.L., "Numerical simulation of the integrated space shuttle vehicle in ascent," AIAA Paper 88-4359-CP, Aug., 1988.

2. Buning, P.G., Chan, W.M., Renze, K.J., Sondak, D., Chiu, I.T. and Slotnick, J.P., *OVERFLOW User's Manual, Version 1.6*, NASA Ames Research Center, Moffett Field, CA, 1991.

3. Steger, J.L. and Rizk, Y.M., "Generation of Three-Dimensional Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations," NASA TM 86753, 1985.

4. Pulliam, T.H. and Chaussee, D.S., "A Diagonal Form of an Implicit Approximate-Factorization Algorithm," *J. Comp.Phys.*, **39**, p347–363, 1981.

5. Van Dalsem, W.R. and Steger, J.L., "Efficient Simulation of Separated Three-Dimensional Viscous Flows Using the Boundary-Layer Equations," *AIAA J.*, **25**, p395–400, 1987.

APPENDIX A

AIAA-91-1588

A Generalized Scheme for Three-Dimensional Hyperbolic Grid Generation

W. Chan
MCAT Institute
Moffett Field, CA

J. Steger
University of California
Davis, CA

AIAA 10th Computational Fluid Dynamics Conference
June 24-26, 1991 / Honolulu, HI

A GENERALIZED SCHEME FOR THREE-DIMENSIONAL HYPERBOLIC GRID GENERATION

William M. Chan*
MCAT Institute, NASA-Ames Research Center
Moffett Field, California 94035

Joseph L. Steger**
Department of Mechanical, Aeronautical and Materials Engineering
University of California at Davis
Davis, California 95616

Abstract

A hyperbolic grid generation scheme formulated from grid orthogonality and cell volume specification is generalized such that high quality three-dimensional grids can be obtained for a wide variety of geometries. The speed of the scheme is one to two orders of magnitude faster than typical elliptic grid generation methods. The robustness of the scheme is significantly enhanced by several new techniques. By using a spatially variable smoothing coefficient, orthogonality and smoothness of the grid are maintained around complex concave and convex surface topologies. A metric correction procedure is employed to guarantee that the grid marches out of a corner by bisecting the angles subtended by the neighboring points. Extra robustness at severe convex corners is achieved by special local treatments. Different extrapolation methods are used to provide smoothness at floating edges and axis regions. The versatility of the new hyperbolic grid generation scheme is demonstrated by three-dimensional grids generated for external components of the integrated Space Shuttle vehicle and the SOFIA telescope.

1. Introduction

One of the most popular approaches for generating structured grids is by the solution of a set partial differential equations. The governing equations can be classified into three types: elliptic, parabolic and hyperbolic. The most widely used grid generation methods require the solution of a set of elliptic equations[1-4]; however, parabolic and hyperbolic equations have also been successfully employed[5-10] and are advantageous for certain applications.

Since the solution of the elliptic equations satisfies the maximum principle, the grids generated are typically smooth. Moreover, the formulation of the elliptic equations allow exact specifications of all boundary point locations. However, grid orthogonality cannot be maintained with conventional elliptic grid generation methods and boundary surface refinement can be difficult to impose. The user input required to set up boundary distributions can also be time consuming. With hyperbolic grid generation, nearly orthogonal grids with excellent clustering control can be produced with one to two orders of magnitude savings in computer time over elliptic grid generation. However, hyperbolic grid generation methods are less robust, tend to propagate input discontinuities, and the outer boundary location cannot be precisely specified. Hence, they are usually restricted to the generation of grids for external flows or for chimera overset-grid schemes[13] where the exact location of the outer boundary is not constrained. This later application has become more important, however, and this coupled with the efficiency and often superior grid quality obtained with hyperbolic grid generation schemes motivates investigation into improving their robustness.

In the last few years, the hyperbolic grid generation algorithm described in Ref. 7 has evolved to include some significant enhancements. A wider range of boundary conditions can now be treated and some feedback features have been added so that dissipative terms (which give the equations a somewhat parabolic nature and smoothness) are now automatically adjusted depending on grid evolution or character. The resulting hyperbolic grid generation scheme is significantly more robust, produces higher quality grids, and can treat a wider variety of topologies.

The governing equations for three-dimensional hyperbolic grid generation are presented in §2. The numerical marching scheme employed to solve these equations is described in §3. Three factors important in controlling grid quality, the boundary conditions, cell volume specification and added smoothing are discussed in §4, 5 and 6 respectively. Discontinuities due to body shape or initial grid point distribution can present special problems to a hyperbolic solver that dissipation alone cannot satisfactorily cure. Corner points are especially difficult and special remedies are described in §7 and 8. A metric correction procedure

* Research Scientist.
** Professor, Mechanical, Aeronautical and Materials Engineering, Associate Fellow AIAA.

which is essential in providing smooth grids at corners with uneven grid spacings is described in §7. Extra robustness at sharp convex corners can be achieved by switching from solving the hyperbolic equations to some other equations by marching at the convex corner points. These schemes are described in §8. In §9, examples from grids produced with the new hyperbolic grid generator for various external components of the Space Shuttle launch vehicle and the SOFIA telescope are presented. Finally, conclusions are given in §10.

2. Governing Equations

Generalized coordinates $\xi(x,y,z)$, $\eta(x,y,z)$, $\zeta(x,y,z)$ are sought where the body surface is chosen to coincide with $\zeta(x,y,z)=0$ and the surface distributions of $\xi=$ const and $\eta=$ const are user-specified. With external aerodynamic applications in mind, the location of the outer boundary $\zeta(x,y,z)=\zeta_{max}$ is not specified. The governing equations are derived from orthogonality relations between ξ and ζ, between η and ζ, and a cell volume or finite Jacobian J constraint[7]:

$$x_\xi x_\zeta + y_\xi y_\zeta + z_\xi z_\zeta = 0, \tag{2.1a}$$

$$x_\eta x_\zeta + y_\eta y_\zeta + z_\eta z_\zeta = 0, \tag{2.1b}$$

$$x_\xi y_\eta z_\zeta + x_\zeta y_\xi z_\eta + x_\eta y_\zeta z_\xi - x_\xi y_\zeta z_\eta - x_\eta y_\xi z_\zeta - x_\zeta y_\eta z_\xi = \Delta V, \tag{2.1c}$$

or, with $\vec{r}$ defined as $(x,y,z)^T$

$$\vec{r}_\xi \cdot \vec{r}_\zeta = 0, \tag{2.2a}$$

$$\vec{r}_\eta \cdot \vec{r}_\zeta = 0, \tag{2.2b}$$

$$\left|\frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)}\right| = J^{-1} = \Delta V. \tag{2.2c}$$

Equations (2.1) comprise a system of nonlinear partial differential equations in which x, y, and z are specified as initial data at $\zeta=0$. Local linearization of Eqs. (2.1) about the state 0 results in the system of grid generation equations

$$A_0(\vec{r}-\vec{r}_0)_\xi + B_0(\vec{r}-\vec{r}_0)_\eta + C_0(\vec{r}-\vec{r}_0)_\zeta = \vec{f} \tag{2.3}$$

with

$$A = \begin{pmatrix} x_\zeta & y_\zeta & z_\zeta \\ 0 & 0 & 0 \\ (y_\eta z_\zeta - y_\zeta z_\eta) & (x_\zeta z_\eta - x_\eta z_\zeta) & (x_\eta y_\zeta - x_\zeta y_\eta) \end{pmatrix}, \tag{2.4a}$$

$$B = \begin{pmatrix} 0 & 0 & 0 \\ x_\zeta & y_\zeta & z_\zeta \\ (y_\zeta z_\xi - y_\xi z_\zeta) & (x_\xi z_\zeta - x_\zeta z_\xi) & (x_\zeta y_\xi - x_\xi y_\zeta) \end{pmatrix}, \tag{2.4b}$$

$$C = \begin{pmatrix} x_\xi & y_\xi & z_\xi \\ x_\eta & y_\eta & z_\eta \\ (y_\xi z_\eta - y_\eta z_\xi) & (x_\eta z_\xi - x_\xi z_\eta) & (x_\xi y_\eta - x_\eta y_\xi) \end{pmatrix}, \tag{2.4c}$$

and

$$\vec{f} = \begin{pmatrix} -(\frac{\partial\vec{r}}{\partial\xi}\cdot\frac{\partial\vec{r}}{\partial\zeta})_0 \\ -(\frac{\partial\vec{r}}{\partial\eta}\cdot\frac{\partial\vec{r}}{\partial\zeta})_0 \\ \Delta V - \Delta V_0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \Delta V - \Delta V_0 \end{pmatrix}. \tag{2.4d}$$

Equation (2.3) can be rewritten as

$$A_0\vec{r}_\xi + B_0\vec{r}_\eta + C_0\vec{r}_\zeta = \vec{e} \tag{2.5}$$

with $\vec{e} = (0,0,\Delta V + 2\Delta V_0)^T$. Now C_0^{-1} exists unless $(\Delta V_0) \to 0$, so Eq. (2.5) can be rewritten as

$$C_0^{-1}A_0\vec{r}_\xi + C_0^{-1}B_0\vec{r}_\eta + \vec{r}_\zeta = C_0^{-1}\vec{e}. \tag{2.6}$$

Although the algebraic verification is not trivial, $C_0^{-1}A_0$ and $C_0^{-1}B_0$ are symmetric matrices[7]. This implies that the linearized system Eq. (2.6) is hyperbolic and can be marched with ζ serving as the "time-like" direction.

3. Numerical Marching Scheme

The system of grid generation equations given by Eq. (2.5) are solved with a non-iterative implicit finite difference scheme. An unconditionally stable implicit scheme has the advantage that the marching step size in ζ can be arbitrarily selected based only on considerations of accurately generating the grid. Linearization is performed about the previous marching step in ζ.

Let $\Delta\xi = \Delta\eta = \Delta\zeta = 1$ such that $\xi = j-1$, $\eta = k-1$, and $\zeta = l-1$. Central spatial differencing of Eq. (2.5) in ξ and η with two-point backward implicit differencing in ζ leads to

$$A_l\delta_\xi(\vec{r}_{l+1}-\vec{r}_l) + B_l\delta_\eta(\vec{r}_{l+1}-\vec{r}_l) + C_l\nabla_\zeta\vec{r}_{l+1} = \vec{g}_{l+1} \tag{3.1}$$

where

$$\vec{g}_{l+1} = \begin{pmatrix} 0 \\ 0 \\ \Delta V_{l+1} \end{pmatrix}$$

and

$$\delta_\xi\vec{r}_j = \frac{\vec{r}_{j+1}-\vec{r}_{j-1}}{2}, \quad \delta_\eta\vec{r}_k = \frac{\vec{r}_{k+1}-\vec{r}_{k-1}}{2},$$

$$\nabla_\zeta\vec{r}_{l+1} = \vec{r}_{l+1} - \vec{r}_l.$$

Throughout, only those indices that change are indicated, thus $r_{l+1} \Rightarrow r_{j,k,l+1}$ and $r_{j+1} \Rightarrow r_{j+1,k,l}$, etc.

Multiplying through by C_l^{-1} and approximately factoring gives

$$(I + C_l^{-1}B_l\delta_\eta)(I + C_l^{-1}A_l\delta_\xi)(\vec{r}_{l+1} - \vec{r}_l) = C_l^{-1}\vec{g}_{l+1} \tag{3.2}$$

where I is the identity matrix. The problem is now reduced to solving a sequence of block tridiagonal systems.

Since all ξ- and η-derivatives are approximated by central differencing, numerical dissipation terms are added in these directions. For simplicity, only second differences are used which are explicitly and implicitly included in the basic algorithm as

$$[I + C_l^{-1}B_l\delta_\eta - \epsilon_{i_\eta}(\Delta\nabla)_\eta][I + C_l^{-1}A_l\delta_\xi - \epsilon_{i_\xi}(\Delta\nabla)_\xi] \times(\vec{r}_{l+1} - \vec{r}_l) = C_l^{-1}\vec{g}_{l+1} - [\epsilon_{e_\xi}(\Delta\nabla)_\xi + \epsilon_{e_\eta}(\Delta\nabla)_\eta]\vec{r}_l \tag{3.3}$$

where, for example,

$$(\Delta\nabla)_\eta\vec{r} = \vec{r}_{k+1} - 2\vec{r}_k + \vec{r}_{k-1},$$

and with $\epsilon_{i_\xi} \approx 2\epsilon_{e_\xi}$ and $\epsilon_{i_\eta} \approx 2\epsilon_{e_\eta}$. Additional smoothing and implicitness are put into the algorithm[11] by differencing $\nabla_\zeta\vec{r} = \vec{F}$ as $\vec{r}_{l+1} - \vec{r}_l = (1+\theta)\vec{F}_{l+1} - \theta\vec{F}_l$, with $\theta \geq 0$. This differencing in ζ is incorporated into Eq. (3.3) as

$$[I + (1+\theta_\eta)C_l^{-1}B_l\delta_\eta - \epsilon_{i_\eta}(\Delta\nabla)_\eta] \times [I + (1+\theta_\xi)C_l^{-1}A_l\delta_\xi - \epsilon_{i_\xi}(\Delta\nabla)_\xi] \times (\vec{r}_{l+1} - \vec{r}_l) = C_l^{-1}\vec{g}_{l+1} - [\epsilon_{e_\xi}(\Delta\nabla)_\xi + \epsilon_{e_\eta}(\Delta\nabla)_\eta]\vec{r}_l \tag{3.4}$$

The values of θ_ξ and θ_η are kept at zero unless the body contains concave profiles in ξ or η. Suppose the body has a concave profile in the ξ-direction, then values of θ_ξ of 1 to 4 are effective in preventing grid lines from crossing. The same concept also applies in the η direction.

The coefficient matrices A_l, B_l and C_l contain derivatives in ξ, η and ζ. The derivatives in ξ and η are obtained by central differencing while the derivatives in ζ are obtained from Eqs. (2.1) as a linear combination of ξ- and η-derivatives as follows

$$\begin{pmatrix} x_\zeta \\ y_\zeta \\ z_\zeta \end{pmatrix} = \frac{\Delta V}{Det(C)} \begin{pmatrix} y_\xi z_\eta - y_\eta z_\xi \\ x_\eta z_\xi - x_\xi z_\eta \\ x_\xi y_\eta - x_\eta y_\xi \end{pmatrix} = C^{-1}\vec{g} \tag{3.5}$$

with

$$Det(C) = (y_\xi z_\eta - y_\eta z_\xi)^2 + (x_\eta z_\xi - x_\xi z_\eta)^2 + (x_\xi y_\eta - x_\eta y_\xi)^2.$$

A discussion on a more appropriate way to compute these ζ-derivatives is given in §7 .

4. Boundary Conditions

Five types of implicit boundary conditions have been implemented in the grid generation code at the ξ and η boundaries (except for the axis condition which is only implemented in ξ). In the following, coordinate increments $\Delta\vec{r} = \vec{r}_{l+1} - \vec{r}_l$ are represented by $(\Delta x, \Delta y, \Delta z)^T$.

(a) *Periodicity* - All derivatives at the end points in the periodic direction are evaluated by 'wrapping around'. A periodic block-tridiagonal solver is used for the inversion of the appropriate factor in the left-hand side of Eq. (3.4).

(b) *Constant Cartesian plane* - If a ξ or η boundary is restricted to an x =const, y =const or z =const plane, then that value is enforced and the other variables are 'floated'. For example, for an x =const plane at the $j = 1$ boundary, x is held constant and y and z are floated using

$$\begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix}_{j=1} = \begin{pmatrix} 0 \\ \Delta y \\ \Delta z \end{pmatrix}_{j=2} . \tag{4.1}$$

(c) *Symmetry plane* - Conventional reflection planes are used to impose symmetry about any $x = 0$, $y = 0$ or $z = 0$ plane and values are updated implicitly. For example, to update a reflected plane at $j = 1$ for a symmetry condition about $x = 0$ corresponding to $j = 2$, x reflects odd and y and z reflect even as (in delta form):

$$\begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix}_{j=1} = \begin{pmatrix} -\Delta x \\ \Delta y \\ \Delta z \end{pmatrix}_{j=3} . \tag{4.2}$$

Since the $j = 2$ plane in this example may deviate very slightly from $x = 0$ away from the body surface due to round-off error, post-processing is done to set the x coordinate exactly to zero at $j = 2$ at each incremental level in l. Otherwise slight asymmetry may be experienced in a flow solver.

(d) *Floating edge* - Much as in the case of a constant Cartesian plane, an entire ξ or η boundary can be floated using the simple hyperbolic equation $\vec{r}_{\xi\zeta} = 0$ or $\vec{r}_{\eta\zeta} = 0$ to update a boundary plane. This is essentially a zeroth-order extrapolation of $\Delta\vec{r}$ from the adjacent interior value, and it often works well. Occasionally, however, the floating boundary plane itself may tend to 'roll in' or 'kink' while its neighboring planes in the interior remain smooth. Since the interior points are fine, this problem has been remedied by the addition of a fictitious line of points on the body surface next to the floating edge by linear extrapolation from the interior. The dimension of the surface grid in a particular direction is

temporarily increased by one or two depending if one or two floating edges are present in that direction. The 3D grid is then generated over the extended surface grid. After the 3D grid is generated, the fictitious plane or planes of points is removed. The addition and removal of these fictitious points are carried out internal to the code and is not a concern for the user.

When using the chimera overset-grid scheme, it is particularly desirable to have the floating edges splay outwards, i.e., in the direction away from the interior of the grid; thus providing better overlap between neighboring grids. Although a free floating edge using zeroth-order extrapolation may bend inwards or outwards, it is found that using a mixed zeroth and first-order extrapolation scheme tends to bend the edge outwards. For example, at the $j = 1$ boundary, we have

$$\begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix}_{j=1} = \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix}_{j=2} + \epsilon_{ex} \left[\begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix}_{j=2} - \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix}_{j=3} \right] \quad (4.3)$$

where $0 \leq \epsilon_{ex} \leq 1$. Zeroth-order and first-order extrapolation schemes are recovered at the two limits of ϵ_{ex} respectively. Typically, a value of $\epsilon_{ex} \approx 0.05 - 0.2$ is used. An example using the floating edge condition with $\epsilon_{ex} = 0.2$ at the boundaries of a flat plate is shown in Figure 1.

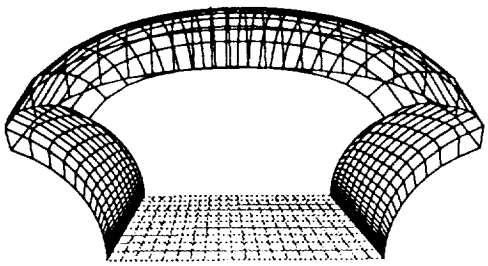


Figure 1. Outward-splaying edges of a flat plate.

(e) *Axis* - When the axis logic is used in the j-direction, it is assumed that the boundary condition in the k-direction is either periodic or that of symmetry or constant planes at both ends. For example, one may generate a grid for a complete ellipsoid (periodic in k) or just one half of it (symmetry or constant planes at end points in k).

The treatment of the axis requires special attention in order to produce smooth results. Two methods are described below. The first method requires the user to adjust certain input parameters and is able to produce smooth results for all cases encountered so far. The second method does not require the user to adjust any parameters but is only able to produce smooth results for a smaller number of cases.

The first method involves using a mixed zeroth and first-order extrapolation and volume scaling. The axis point is updated implicitly by imposing that $\Delta\vec{r}$ at the axis is extrapolated by a mixed zeroth and first-order scheme similar to that given by Eq. (4.3). The resulting k_{max} predicted values, where k_{max} is the number of points around the axis, are averaged to produce a unique value at the axis. The methods of volume specification described in §5 below usually produce volumes that are too large near the axis. Hence, the volumes in the ring of points around the axis are scaled by a reduction factor in the range 0.1 to 1.0. Typical values of the extrapolation factor and volume scaling factor are 0.4 and 0.3 respectively. Figure 2 shows the symmetry plane of an external tank grid with an axis coming off a pointed nose and an axis coming off a flat back. We see that the axis logic described above is able to provide smoothness at both the front and the back regions.

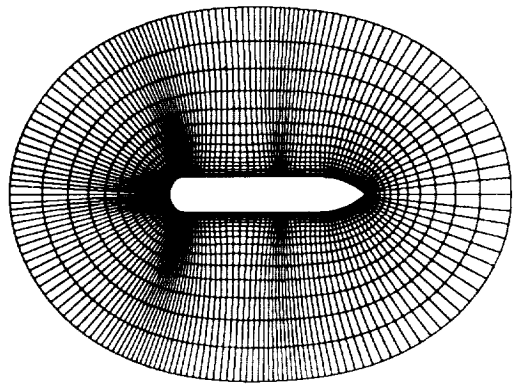


Figure 2. Symmetry plane of external tank grid with a pointed axis in front and a flat axis at the back.

The second method requires an explicit update of the axis points. Axis values are first updated using a special local coordinate system $(\tilde{\xi}_i, \tilde{\eta}_i)$ that bridges across the axis. The grid generation equations are used on the local coordinates to update x, y and z using an explicit differencing procedure and central space differencing that avoids

using the center point (a similar approach has been used for unstructured data in a $\zeta = const$ plane, see Ref. 12). There are $k_{\max}$ such coordinates introduced so that data around the axis can be sampled, and all values explicitly predicted from the differenced grid generation equations are averaged since all axis values of x, y and z must be the same. These values are then filtered or smoothed with the interior grid values once they are updated in ζ. (Note that the numerical dissipation coefficients defined with Eq. (6.2) can become too large near an axis for this method. For example, near an $\xi = const$ axis ϵ_{e_η} is reduced.) A corrector step has not yet been imposed and the currently implemented unstable one step explicit scheme is only satisfactory for rapidly expanding axis regions, for example, the nose of the external tank in Figure 2.

5. Cell Volume Specification

With the hyperbolic grid generation method, one of the means of controlling the grid is by specification of the cell volumes, $\Delta V_{j,k,l}$. Through the cell volumes, the extent and clustering of the grid can be modified. Since the cell volume at each point must be given, it is clear that the user must devise a simple global method for specifying volumes.

Two methods of volume specification are described below. Both of them require a radial point distribution function $s_{j,k,l}$ which prescribes the arc length between points in the direction normal to the body surface. In the most general case, s is a function of j, k and l since each point on the body surface may be stretched to a different outer boundary location (For example, in the case of a body at a positive angle of attack in hypersonic flow, one may wish to have the outer boundary of the grid further away on the top surface than on the lower surface). Typically, the points are stretched away from the body exponentially. If grid point spacing control is required at the outer boundary as well as at the body surface, or if more uniform grid point spacing is required away from the inner body surface, hyperbolic tangent stretching can be used. The ability to control the grid spacing at each end of the domain is useful when multiple zones consisting of stretched points or uniformly-spaced points are desired. Also, the presence of more points in the far field of a component grid helps to improve grid-overlap regions for the chimera overset-grid scheme[13].

In the first method for specifying cell volumes, the specified volume at each point is set equal to the computed surface area element times a user specified arc length. Specifically,

$$\Delta V_{j,k,l} = \Delta s_{j,k,l} \Delta A_{j,k,l} \tag{5.1}$$

where $\Delta s_{j,k,l} = s_{j,k,l+1} - s_{j,k,l}$ is the user specified arc length for marching and $\Delta A_{j,k,l}$ is the surface area element. In this kind of volume control specification, if an initial distribution of points is highly clustered in ξ or η, then these points tend to remain highly clustered even far away from the body. In order to obtain a more uniform far-field distribution, the volumes specified from Eq. (5.1) are averaged in ξ and η with each step taken in ζ. For example, the averaged volume $\Delta \bar{V}_{j,k,l}$ can be computed as

$$\Delta \bar{V}_{j,k,l} = (1 - \nu_a)\Delta V_{j,k,l} + \frac{\nu_a}{4}(\Delta V_{j+1,k,l} + \Delta V_{j-1,k,l} + \Delta V_{j,k+1,l} + \Delta V_{j,k-1,l}) \tag{5.2}$$

where this is applied one or more times with each step in ζ. A typical value of ν_a that has been employed is 0.16.

The second method of generating mesh cell volumes is to form a grid about a 'similar' but simple reference body for which the grid can be generated analytically, and to use the cell volumes from this reference grid for the more complex problem. In order to mesh a wing with a spherical-like grid, for example, cell volumes can be specified by analytically building a grid for a simple body with the same surface area. An obvious choice is a spherical grid that has uniform angle spacing and a radial distribution prescribed by $s_{j,k,l}$. In this special case, the control cell volumes are analytically known. The grid cell volumes of this spherical reference grid are then used to specify the cell volumes of the wing grid. However, the wing will not have the same kind of surface area distribution as a sphere with equal angle distribution. So an adjustment is made of the form

$$\Delta V_{j,k,l} = \left[(1 - \nu) + \frac{(\Delta A_{j,k})_{wing}}{(\Delta A_{j,k})_{sphere}} \nu\right] (\Delta V_{j,k,l})_{sphere} \tag{5.3}$$

where $\nu \to 1$ for small l and $\nu \to 0$ for large l. That is, the volumes are adjusted initially to the local boundary surface increments. But as the solution is marched outwards, the uniform spherical volumes gradually become specified. When such an approach is used, the far field portion of the grid tends to be more uniformly spherical.

6. Smoothing

Since central differencing is used in the numerical scheme, artificial dissipation terms are added to control oscillations arising from the odd-even uncoupling of grid points. An equally important function of the added dissipation is to control the smoothness of the resulting grid. The form and magnitude of the

added dissipation are extremely important in shaping the grid quality.

The parameters θ_ξ and θ_η on the left-hand side of the grid generation equations (3.4) can be thought of as a type of smoothing in the ζ marching direction. In the ξ and η directions, it is adequate for the purpose of controlling the smoothness of the grid to use just second-order smoothing.

For simple surface topologies, such as an ellipsoid which has a convex profile in all directions, a constant dissipation coefficient is sufficient to provide good grid quality. However, body surfaces encountered in aerodynamic applications are frequently much more complex with combinations of sharp convex and concave corners. Hence, the dissipation, which tends to reduce grid curvature and orthogonality, must be applied more selectively. The reduction of grid curvature can be advantageous in concave regions and detrimental in convex regions. In order to prevent grid lines from converging and crossing, the dissipation has to be relatively high in concave regions (e.g., in the region between the orbiter fuselage and the wing root in Figure 7). Conversely, the dissipation must be kept small near the body surface and sharp convex corners (such as the convex corners of the IEA box in Figure 6), otherwise the resulting grid spacing in the ζ direction would become too reduced or even negative. A spatially uniform dissipation coefficient is unable to satisfy all of the above requirements. Spatially-varying dissipation coefficients have been used previously, but they have only accounted for variations of mesh size, specifically, $C^{-1}A$ and $C^{-1}B$ variations. A spatially-varying form of the dissipation coefficient which has worked very well for many cases is described below.

The explicit second-order dissipation D_e added to the right hand side of the equations is given by

$$D_e = -[\epsilon_{e\xi}(\Delta\nabla)_\xi + \epsilon_{e\eta}(\Delta\nabla)_\eta]\vec{r}_l, \qquad (6.1)$$

with

$$\epsilon_{e\xi} = \epsilon_c R_\xi N_\xi, \qquad \epsilon_{e\eta} = \epsilon_c R_\eta N_\eta, \qquad (6.2)$$

where ϵ_c is a user-supplied constant of $O(1)$, N_ξ and N_η are approximations to the matrix norms $||C^{-1}A||$ and $||C^{-1}B||$ respectively, given by

$$N_\xi = \sqrt{\frac{x_\zeta^2 + y_\zeta^2 + z_\zeta^2}{x_\xi^2 + y_\xi^2 + z_\xi^2}}, \qquad N_\eta = \sqrt{\frac{x_\zeta^2 + y_\zeta^2 + z_\zeta^2}{x_\eta^2 + y_\eta^2 + z_\eta^2}}, \qquad (6.3)$$

and R_ξ, R_η are the dissipation coefficients given by

$$R_\xi = S_l\, \tilde{d}^\xi_{j,k,l}\, a^\xi_{j,k,l}, \qquad R_\eta = S_l\, \tilde{d}^\eta_{j,k,l}\, a^\eta_{j,k,l}. \qquad (6.4)$$

The dissipation coefficients consist of three functions which provide different ways to automatically adjust the local dissipation appropriately depending on the local grid topology:

(1) A scaling function S_l which varies with normal distance from the body surface.

(2) A grid point distribution sensor function, $\tilde{d}^\xi_{j,k,l}$ or $\tilde{d}^\eta_{j,k,l}$ depending on the direction, which senses mesh convergence based on the distances between neighboring grid points.

(3) A grid angle function, $a^\xi_{j,k,l}$ or $a^\eta_{j,k,l}$ depending on the direction, which senses mesh convergence based on the angles between neighboring grid points.

The form of the scaling function S_l is given by

$$S_l = \begin{cases} \sqrt{\frac{(l-1)}{(l_{\max}-1)}} & 2 \le l \le l_{\text{trans}}\ , \\ \sqrt{\frac{(l_{\text{trans}}-1)}{(l_{\max}-1)}} & l_{\text{trans}}+1 \le l \le l_{\max}\ , \end{cases} \qquad (6.5)$$

where $l_{\max}$ is the number of points in the l direction and l_{trans} is restricted to the range $[\frac{3}{4}, 1] \times l_{\max}$. With $d^\xi_{j,k,l}$ and $d^\eta_{j,k,l}$ defined by Eqs. (6.8a,b), l_{trans} is set to l when one or both of the following is true:

$$\max_{j,k} d^\xi_{j,k,l} - \max_{j,k} d^\xi_{j,k,l-1} < 0, \qquad (6.6a)$$

$$\max_{j,k} d^\eta_{j,k,l} - \max_{j,k} d^\eta_{j,k,l-1} < 0. \qquad (6.6b)$$

Once l_{trans} is located; the above tests are not performed for $l > l_{\text{trans}}$.

The purpose of the scaling function S_l is to guarantee small dissipation, and hence grid orthogonality, near the body surface. As one moves away from the body surface, dissipation is increased since grid lines may begin to converge in concave regions and some loss of grid curvature in convex regions is no longer a problem. Away from the body surface, the grid point distribution sensor function and the grid angle function alone are sufficient to provide the appropriate amount of dissipation. Hence, the influence of S_l is removed by setting it to a constant at some location l_{trans} away from the body. It is found that a good criterion for locating l_{trans} is when the convergence of local grid lines is slowing down in some sense given by Eqs. (6.6a,b). For all the cases encountered so far, it is sufficient to test for l_{trans} for $l \ge \frac{3}{4} l_{\max}$. For some cases, l_{trans} may be less than $l_{\max}$ which reduces loss of orthogonality near the outer boundary; while for other cases, l_{trans} may have to be equal to $l_{\max}$ when increasing values of dissipation are needed all the way to the outer boundary to prevent grid lines from converging.

The forms of the grid point distribution sensor functions $\tilde{d}^\xi_{j,k,l}$, $\tilde{d}^\eta_{j,k,l}$ are given by

$$\begin{aligned} \tilde{d}^\xi_{j,k,l} &= \max\left[(d^\xi_{j,k,l})^{2/S_l},\ 0.1\right], \\ \tilde{d}^\eta_{j,k,l} &= \max\left[(d^\eta_{j,k,l})^{2/S_l},\ 0.1\right], \end{aligned} \qquad (6.7)$$

where

$$d^{\xi}_{j,k,l} = \frac{|\vec{r}_{j+1,k,l-1} - \vec{r}_{j,k,l-1}| + |\vec{r}_{j-1,k,l-1} - \vec{r}_{j,k,l-1}|}{|\vec{r}_{j+1,k,l} - \vec{r}_{j,k,l}| + |\vec{r}_{j-1,k,l} - \vec{r}_{j,k,l}|}, \tag{6.8a}$$

$$d^{\eta}_{j,k,l} = \frac{|\vec{r}_{j,k+1,l-1} - \vec{r}_{j,k,l-1}| + |\vec{r}_{j,k-1,l-1} - \vec{r}_{j,k,l-1}|}{|\vec{r}_{j,k+1,l} - \vec{r}_{j,k,l}| + |\vec{r}_{j,k-1,l} - \vec{r}_{j,k,l}|}. \tag{6.8b}$$

The distribution of grid points in the ξ and η directions are monitored by the functions $d^{\xi}_{j,k,l}$ and $d^{\eta}_{j,k,l}$ respectively. The quantity $d^{\xi}_{j,k,l}$ is the ratio of the distances between a grid point and its neighbors in the ξ direction at level $(l-1)$ to that at level l. This ratio is high in concave regions and hence more dissipation is provided here. It is of order one or smaller in flat or convex regions where less dissipation is needed. Similarly, $d^{\eta}_{j,k,l}$ represents the corresponding quantity in the η direction. The quantities $\tilde{d}^{\xi}_{j,k,l}$ and $\tilde{d}^{\eta}_{j,k,l}$ are constructed from $d^{\xi}_{j,k,l}$ and $d^{\eta}_{j,k,l}$ respectively which are raised to the power $2/S_l$ in order to counteract the small value of S_l close to the body surface. Also, the values of $\tilde{d}^{\xi}_{j,k,l}$ and $\tilde{d}^{\eta}_{j,k,l}$ are limited from becoming too low by a limiter of 0.1. Note that a grid point distribution sensor function based on cell area ratios is not as effective since the grid lines could be converging in one direction but not the other and the cell areas would not decrease very much.

The grid angle functions $a^{\xi}_{j,k,l}, a^{\eta}_{j,k,l}$ are more conveniently defined in terms of the following unit vectors. Let the vectors pointing in the plus and minus ξ directions at grid point (j,k,l) be represented by $\vec{r}^{\,+}_j$ and $\vec{r}^{\,-}_j$ respectively, where

$$\vec{r}^{\,+}_j = \vec{r}_{j+1,k,l} - \vec{r}_{j,k,l} \quad \vec{r}^{\,-}_j = \vec{r}_{j-1,k,l} - \vec{r}_{j,k,l}, \tag{6.9}$$

and let $\hat{r}^{+}_j$ and $\hat{r}^{-}_j$ be the respective unit vectors for $\vec{r}^{\,+}_j$ and $\vec{r}^{\,-}_j$. Similar expressions are defined for $\vec{r}^{\,+}_k$, $\vec{r}^{\,-}_k$ and $\hat{r}^{+}_k$, $\hat{r}^{-}_k$ for vectors and unit vectors pointing in the plus and minus η directions respectively at grid point (j,k,l). The local unit normal $\hat{n}_{j,k,l}$ based on the cross product of the above unit vectors is given by

$$\hat{n}_{j,k,l} = \frac{(\hat{r}^{+}_j - \hat{r}^{-}_j) \times (\hat{r}^{+}_k - \hat{r}^{-}_k)}{|(\hat{r}^{+}_j - \hat{r}^{-}_j) \times (\hat{r}^{+}_k - \hat{r}^{-}_k)|}. \tag{6.10}$$

The cosine of the local half angle $\alpha_{j,k,l}$ in the ξ direction is then given by

$$\cos\alpha_{j,k,l} = \hat{n}_{j,k,l} \cdot \hat{r}^{+}_j = \hat{n}_{j,k,l} \cdot \hat{r}^{-}_j. \tag{6.11}$$

The grid angle function $a^{\xi}_{j,k,l}$ is then defined as

$$a^{\xi}_{j,k,l} = \begin{cases} 1/(1-\cos^2\alpha_{j,k,l}) & \text{if } 0 \le \alpha_{j,k,l} \le \frac{\pi}{2}, \\ 1 & \text{if } \frac{\pi}{2} < \alpha_{j,k,l} \le \pi. \end{cases} \tag{6.12}$$

A similar expression is defined for the grid angle function $a^{\eta}_{j,k,l}$ in terms of the cosine of the local half angle $\beta_{j,k,l}$ in the η direction.

The smoothing provided by the grid point distribution functions in Eq. (6.7) has to be modified locally at grid points located at very sharp concave corners. These corners are detected by computing the half angles α and β subtended by neighboring grid points in the ξ and η directions respectively (see Eq. 6.11). At a severe concave corner point, extra dissipation is required to prevent crossing of grid lines. The functions $a^{\xi}_{j,k,l}$ and $a^{\eta}_{j,k,l}$ serve to provide the appropriate local modifications to the dissipation required at these corner points. For a typical mesh, the values of $a^{\xi}_{j,k,l}$ and $a^{\eta}_{j,k,l}$ are close to unity at most grid points. Figure 3 shows the grid marching out of a concave angle of 20 degrees. Note that orthogonality is still maintained for the first point off the surface. Grids for concave angles down to 5 degrees have been obtained with the above algorithm.

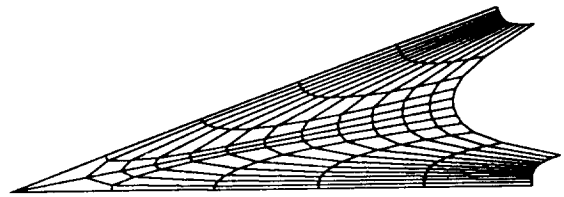


Figure 3. A concave corner at 20 degrees.

A minor local refinement at $l = 2$ of the above dissipation scheme for sharp corners is given in the Appendix.

7. Metric Correction

In order to provide satisfactory resolution of the flow around convex or concave corners, it is important that the grid spacing on each side of the corner be approximately equal as the corner is approached. However, if the surface grid does possess corners with uneven grid spacings for some reason or another, using the spatially-varying dissipation described in §6 alone is not sufficient to provide smoothness at these corners. The situation can be remedied by a metric correction procedure.

With the way the ζ-derivatives are defined in Eq. (3.5), the direction in which the grid will emanate from the corner is such that it is perpendicular to the line joining the two neighbor points of the corner. For a corner with unequally spaced points, this is obviously

not a desirable direction to take (see Figure 4a). In order to guide the grid out in a direction that bisects the angles at a point subtended by its neighbors in both the ξ and η directions, the derivatives $x_\zeta, y_\zeta, z_\zeta$ at the point have been modified to $x'_\zeta, y'_\zeta, z'_\zeta$ as follows:

$$\begin{pmatrix} x'_\zeta \\ y'_\zeta \\ z'_\zeta \end{pmatrix} = \frac{\Delta V}{Det(C')} \begin{pmatrix} y'_\xi z'_\eta - y'_\eta z'_\xi \\ x'_\eta z'_\xi - x'_\xi z'_\eta \\ x'_\xi y'_\eta - x'_\eta y'_\xi \end{pmatrix}, \qquad (7.1)$$

with

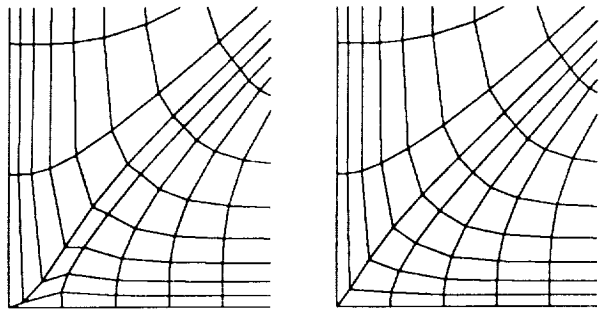
$$Det(C') = (y'_\xi z'_\eta - y'_\eta z'_\xi)^2 + (x'_\eta z'_\xi - x'_\xi z'_\eta)^2 + (x'_\xi y'_\eta - x'_\eta y'_\xi)^2,$$

where

$$(x'_\xi, y'_\xi, z'_\xi)^T = \frac{1}{2}(|\vec{r}_j^{\,+}| + |\vec{r}_j^{\,-}|) \left(\frac{\vec{r}_j^{\,+}}{|\vec{r}_j^{\,+}|} - \frac{\vec{r}_j^{\,-}}{|\vec{r}_j^{\,-}|} \right), \qquad (7.2a)$$

and

$$(x'_\eta, y'_\eta, z'_\eta)^T = \frac{1}{2}(|\vec{r}_k^{\,+}| + |\vec{r}_k^{\,-}|) \left(\frac{\vec{r}_k^{\,+}}{|\vec{r}_k^{\,+}|} - \frac{\vec{r}_k^{\,-}}{|\vec{r}_k^{\,-}|} \right). \qquad (7.2b)$$



(a) (b)

Figure 4. Comparison of treatment of concave corner with unequal grid spacings. (a) Without metric correction, (b) with metric correction .

While the ζ-derivatives should be computed by Eq. (7.1) near the body surface, the original method of computing these quantities should be restored away from the body surface. This can be achieved smoothly by

$$(x_\zeta, y_\zeta, z_\zeta)^T = (1 - \nu_l)(x^o_\zeta, y^o_\zeta, z^o_\zeta)^T + \nu_l (x'_\zeta, y'_\zeta, z'_\zeta)^T, \qquad (7.3)$$

where $\nu_l = 2^{2-l}$ and $x^o_\zeta, y^o_\zeta, z^o_\zeta$ are obtained by Eq. (3.5). We see that Eq. (7.1) is of the same form as Eq. (3.5) except that the ξ- and η-derivatives are replaced by the corresponding primed quantities. These modified ξ- and η-derivatives are constructed in such a way that the neighboring points of the corner appear to be of equal distance from the corner. The result of applying this procedure to a concave corner is shown in Figure 4b. Although the examples shown in the figures are for concave corners, the above scheme works equally well for convex corners.

8. Treatment of Convex Corners

Two methods are presented here which provide extra robustness at convex corners. They both involve switching from solving the hyperbolic grid generation equations (3.4) to some other equations at the convex corner point. The first method is an implicit averaging scheme. Instead of solving the hyperbolic equations at a convex corner, the following averaging equation is solved.

$$\Delta\vec{r}_{j,k} = \frac{1}{2}(\mu_\xi + \mu_\eta)\,\Delta\vec{r}_{j,k}, \qquad (8.1)$$

where

$$\mu_\xi \Delta\vec{r}_{j,k} = \frac{1}{2}(\Delta\vec{r}_{j+1,k} + \Delta\vec{r}_{j-1,k}), \qquad (8.2a)$$

$$\mu_\eta \Delta\vec{r}_{j,k} = \frac{1}{2}(\Delta\vec{r}_{j,k+1} + \Delta\vec{r}_{j,k-1}). \qquad (8.2b)$$

In other words, the marching increment at the corner is the average of the marching increment of its four neighboring points. The form of the above scheme can be made compatible with the hyperbolic grid generation scheme given by Eq. (3.4) by approximate factorization. The equation to be solved at convex corner points is then

$$\left(I - \frac{1}{2}\mu_\xi\right)\left(I - \frac{1}{2}\mu_\eta\right)\Delta\vec{r} = 0. \qquad (8.3)$$

At $l = 2$, the switch to solve Eq. (8.3) is performed if there is *either* a convex corner in ξ ($\cos\alpha_{j,k} < -0.5$, say) or there is a convex corner in η ($\cos\beta_{j,k} < -0.5$), i.e., external angles greater than 240 degrees. The resulting equation would not make sense if only one of the factors is altered. As the grid marches out in l, the switch to return to the normal hyperbolic scheme is performed when the minimum of $\cos\alpha_{j,k}$ and $\cos\beta_{j,k}$ becomes larger than say -0.2. Although some factorization error is present, this implicit averaging scheme has worked well for a variety of convex corners including some cases where the normal scheme has failed. One such case is illustrated by the NACA0012 airfoil shown in Figures 5a and 5b. The point of this example is *not* to show that an O-grid should be used for the airfoil but simply to demonstrate the ability of the scheme to produce high quality grids around sharp convex corners.

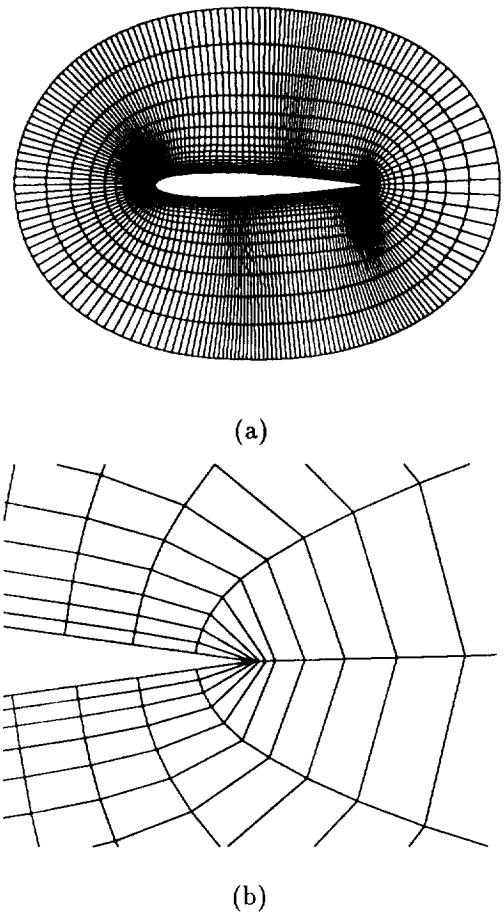


Figure 5. Airfoil with sharp trailing edge and O-grid topology. (a) Far view, (b) close up view of sharp trailing edge region .

The implicit averaging scheme described above possesses two desirable properties. It can be shown geometrically that if the grid marches out orthogonally at the neighbors of the convex corner, then the distance marched out at the corner point is always smaller than the distance marched out at the neighboring points. Moreover, the distance marched out at the corner point becomes smaller as the convex corner becomes sharper. This is very useful in helping to bend the neighboring grid lines towards the sharp corner as the grid is marched out away from the body surface (see Figure 5b). The second property is that the grid marching out from the corner point will bisect the angle at the corner provided the neighboring points march out the same distance and in symmetrical directions to each other. The angle bisecting property still holds even if the grid spacing is unequal from one side to the other side of the convex corner.

An alternative but potentially more robust method than the implicit averaging scheme above is described below. The exact location of the grid point in the next marching step out from a convex corner is predicted in advance. The predicted point is located by marching the grid a distance of $\Delta\tilde{s}_{j,k,l}$ in the direction given by the angle-bisecting unit normal where

$$\Delta\tilde{s}_{j,k,l} = \Delta s_{j,k,l} \times \min(\sin\alpha_{j,k,l}, \sin\beta_{j,k,l}), \quad (8.4)$$

and $\Delta s_{j,k,l}$ is the user specified arc length in the normal direction (see §5). The scaling with the sine of the half angle causes the corner point to march out a smaller distance than its neighbors, thus helping to bend the neighboring grid lines towards the corner in much the same way as described in the last paragraph. The angle-bisecting unit normal is simply the unit vector in the direction of $(x'_\zeta, y'_\zeta, z'_\zeta)^T$ given by Eq. (7.1). Thus, $\Delta\vec{r}$ at the convex corner point can be computed in advance and combined with the grid generation equations (3.4). From numerical trials with different geometries, it was found that this procedure also works well for many types of convex corners.

9. Results and Applications

As the 3D grid is generated by marching out in the ζ-direction, a cell volume check by tetrahedral decomposition and a grid-lines crossing check[15] are performed at each grid cell. If negative volumes or crossings of grid lines are encountered, smoothing parameters can be adjusted appropriately to remove the bad cells.

The vectorized version of the current hyperbolic grid generator runs at 142 megaflops on the CRAY-YMP and requires about 9.7 microseconds of CPU time per grid point. As an example, generation of the largest grid in the Space Shuttle launch vehicle grid system, the orbiter grid for flight Reynolds number ($98 \times 77 \times 57 = 430122$ points), takes 4.17 seconds of CPU time. This is about one to two orders of magnitude faster than typical elliptic grid generators.

The various external components of the integrated Space Shuttle vehicle contain a wide variety of different geometric features that are found in many other applications. Hence, these geometric components provide good tests of robustness for the hyperbolic grid generation scheme described above. Some examples are given below.

The following two examples show how the spatially variable dissipation coefficient works at different types of corners. The first example shows the grid around a sequence of sharp convex and concave corners which appears at the IEA box on the attach ring of the solid rocket booster (see Figure 6). Low dissipation values are needed at the body surface and above the convex corners to maintain orthogonality while high dissipation values are needed in the concave regions to provide grid smoothness. Figure 7 shows the smoothness of the grid in a large concave region - the wing root region at the back of the orbiter.

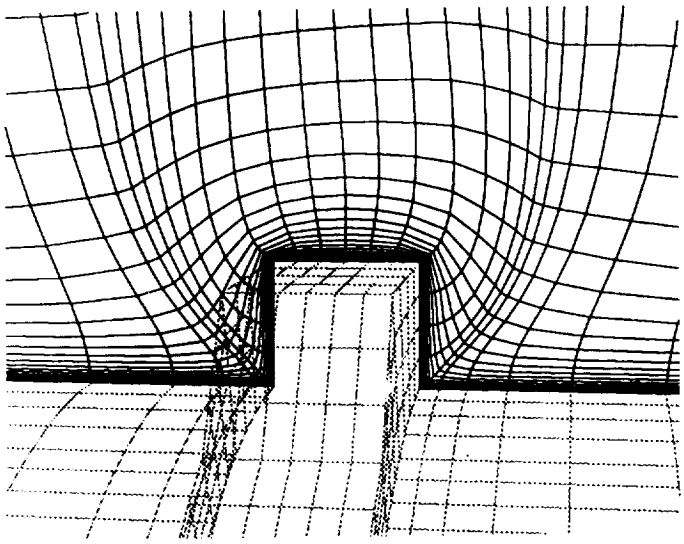


Figure 6. Plane through IEA box on the ring of the solid rocket booster.

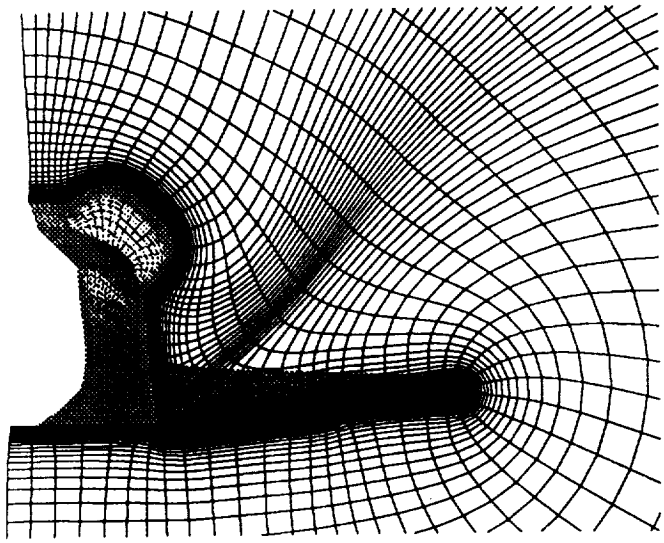


Figure 7. Plane on the back section of the orbiter.

The robustness of the axis logic is tested in the next example where the geometry is non-circular in the circumferential direction around the axis and that the distribution of grid points in this direction is non-uniform. This type of axis condition is present at the tip of the vertical tail section of the orbiter (see Figure 8). The mixed extrapolation scheme with volume scaling is able to produce a smooth grid for this case.

When the chimera overset-grid method is used on two grids whose body surfaces intersect each other, the grid points in the region around the intersection line which are common to both grids are left with no interpolation stencils. In order to remedy this problem, a collar grid can be introduced which covers the region around the intersection line[14]. The most challenging example tested by the grid generator so far is the collar grid which covers the intersection region between the vertical tail and the orbiter (see Figures 9a,b). The methods used to generate the collar grid surface are explained in Ref. 16. The surface of the collar grid is made up of two parts. The top part lies on the surface of the tail down to the intersection line with the orbiter. The lower part has two sections. The first section starts at the intersection line with the tail and then follows the top surface of the orbiter. The second section folds over the back of the orbiter and follows the backward-facing aft-bulkhead of the orbiter. The difficult feature of this geometry is the presence of a region where grid lines are concave in one direction and convex in the other. Slices of the 3D grid viewed from the front and back ends of the collar grid are shown in Figures 10a,b.

The final example is taken from the telescope grid for the SOFIA vehicle[17]. The SOFIA is a modified Boeing 747 with a telescope mounted inside a cavity on the upper surface of the plane. The telescope is topologically similar to a hollow bowl with a truncated cylinder in the middle of the inside of the bowl. Figure 11a shows the surface geometry (shaded) for half the telescope and slices of the 3D grid. The external surface of the telescope consists of both the outside and inside of the bowl together with the middle cylinder. The symmetry plane of the 3D grid is shown in Figure 11b. We see from these figures that although the surface grid may not possess the sufficient number of points to resolve the detailed flow structures, the grid generator is able to produce a smooth grid over the complex combination of concave and convex corners using the techniques described in §6, 7 and 8. The outer boundary of the telescope grid need not be placed far away from the body surface since the entire telescope grid is surrounded by a larger cavity grid and communication between the two grids is achieved via the chimera overset-grid scheme.

10. Conclusions

A robust three-dimensional hyperbolic grid generation scheme has been presented which is able to produce high quality grids for a wide variety of geometries. The improved robustness and the speed advantage of the scheme have made it extremely attractive for users of chimera flow-solvers. Since the scheme is fast, the user can readily adjust the input parameters to fine-tune the grid quality.

The use of a spatially-varying dissipation coefficient based on distances and angles between neighboring grid points gives the grid generator the ability to cope with geometries that are more complex than before. The grid angle bisecting property is provided by the metric correction procedure near the body surface.

Sharp convex corners are automatically detected and the grid generation equations are altered at these corners to further enhance smoothness and robustness.

Acknowledgments

The authors are grateful to Dr. Pieter Buning and Dr. Yehia Rizk for some very useful discussions. A special thanks is owed to Mr. Chris Atwood for providing the surface grid for the SOFIA telescope.

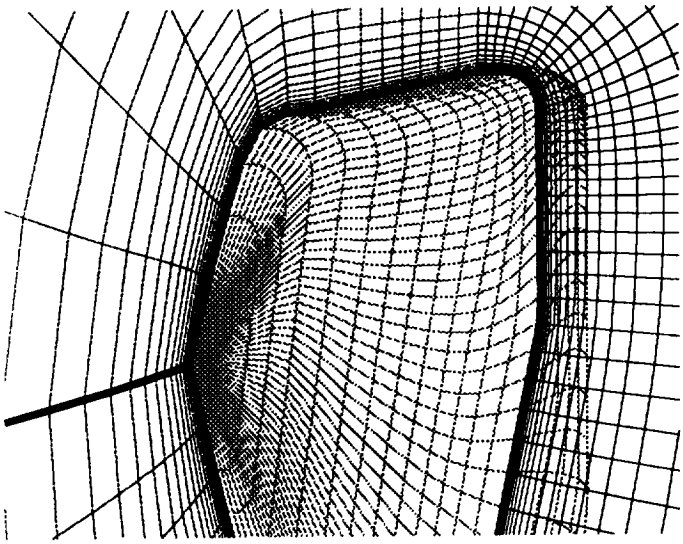


Figure 8. Vertical tail grid showing part of the surface grid near the tip, the 3D grid on the symmetry plane and a section normal to the symmetry plane. The axis is marked by the thick dark line.

Appendix

In order to guarantee orthogonality near the body surface, the dissipation coefficients R_ξ and R_η described in §6 are made to be zero everywhere at $l = 2$ through the scaling function S_l. However, near concave and convex corners, some dissipation has to be restored to maintain smoothness. This can be accomplished by introducing the blanking function $b_{j,k}$ which multiplies R_ξ and R_η at $l = 2$. The blanking function is zero everywhere except near concave ($\alpha_{j,k}$ or $\beta_{j,k} < \pi/3$) and convex ($\alpha_{j,k}$ or $\beta_{j,k} > 2\pi/3$) corners. For a concave or convex corner in the ξ-direction, we set

$$b_{j,k} = b_c, \quad b_{j\pm1,k} = 0.5, \quad b_{j\pm2,k} = 0.25, \qquad (A.1)$$

while for a concave or convex corner in the η-direction, we set

$$b_{j,k} = b_c, \quad b_{j,k\pm1} = 0.5, \quad b_{j,k\pm2} = 0.25, \qquad (A.2)$$

where $b_c = 1$ for a concave corner and $b_c = 0$ for a convex corner. At convex corners, dissipation at the neighboring points to the corner is still needed but the dissipation at the corner point itself should be set to zero in order to produce the desired effects for the special schemes described in §8. In constructing $b_{j,k}$, it is assumed that successive corners in a coordinate direction are separated by at least four points. This is a reasonable assumption if one wishes to provide sufficient resolution for the flow around such corners.

References

1. Winslow, A. M., *Numerical Solution of the Quasilinear Poisson Equation in a Nonuniform Triangle Mesh.* J. Comp. Phys., **2**, 1967, pp. 149-172.

2. Godonov, S. K. and Prokopov, G. P., *The Use of Moving Meshes in Gasdynamical Computations*, USSR Comput. Math. Phys. **12**, 1972, pp. 182-195.

3. Thompson, J. F., Thames, F. C. and Mastin, C. M., *Automatic Numerical Generation of Body-Fitted Curvilinear Coordinate System for Field Containing Any Number of Arbitrary Two-Dimensional Bodies.* J. Comp. Phys. **15**, 1974, pp. 299-319.

4. Thompson, J. F., ed. *Numerical Grid Generation* North-Holland, New York, 1982.

5. Starius, G., *Constructing Orthogonal Curvilinear Meshes by Solving Initial Value Problems*, Numerische Mathematik, **28**, 1977, pp. 25-48.

6. Steger, J. L. and Chaussee, D. S., *Generation of Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations*, SIAM J. Sci. Stat. Comput. **1**, 1980, pp. 431-437.

7. Steger, J. L. and Rizk, Y. M., *Generation of Three-Dimensional Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations*, NASA TM 86753, 1985.

8. Cordova, J. Q. and Barth, T. J., *Grid Generation for General 2-D Regions using Hyperbolic Equations*, AIAA Paper 88-0520, 1988.

9. Nakamura, S. *Marching Grid Generation Using Parabolic Partial Differential Equations*, in *Numerical Grid Generation*, J. F. Thompson ed., North-Holland, New York, 1982. pp. 775-783.

10. Klopfer, G. H., *Solution Adaptive Meshes With a Hyperbolic Grid Generator*, Proceedings of the Second International Conference on Numerical Grid Generation in Computational Fluid Dynamics, Miami, Florida, 1988. pp. 443-453.

11. Kinsey, D. W. and Barth, T. J., *Description of a Hyperbolic Grid Generation Procedure for Arbitrary Two-Dimensional Bodies*, AFWAL TM 84-191-FIMM, 1984.

12. Steger, J. L., *Generation of Three-Dimensional Body-Fitted Grids by Solving Hyperbolic Partial Differential Equations*, in *Finite Element Analysis in Fluids*, T. J. Chung and G. R. Karr, eds., The University of Alabama in Huntsville, UAH Press, 1989.

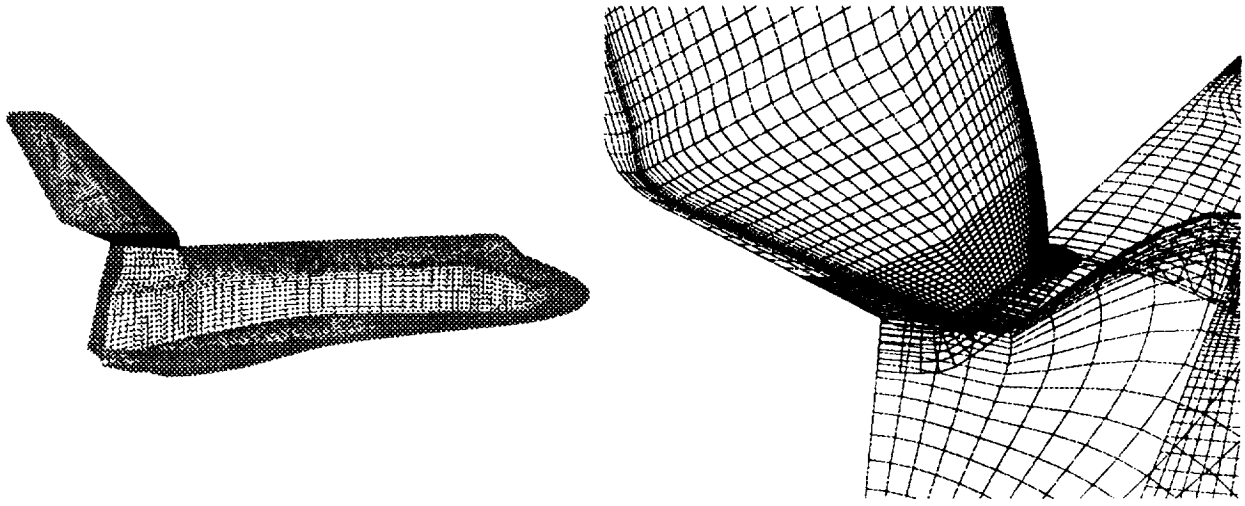
13. Buning, P. G., Chiu, I. T., Obayashi, S., Rizk, Y. M., and Steger, J. L., *Numerical Simulation of the Integrated Space Shuttle Vehicle in Ascent*, AIAA-88-4359-CP, 1988.

14. Steger, J. L., *Notes on Surface Grid Generation using Hyperbolic Partial Differential Equations*, Internal Report TM CFD/UCD 89-101, Department of Mechanical, Aeronautical and Materials Engineering, Univ. of Calif., Davis, 1989.

15. Kordulla, W. and Vinokur, M., *Efficient Computation of Volume in Flow Predictions.* AIAA J., **21**, 1983, pp. 917-918.

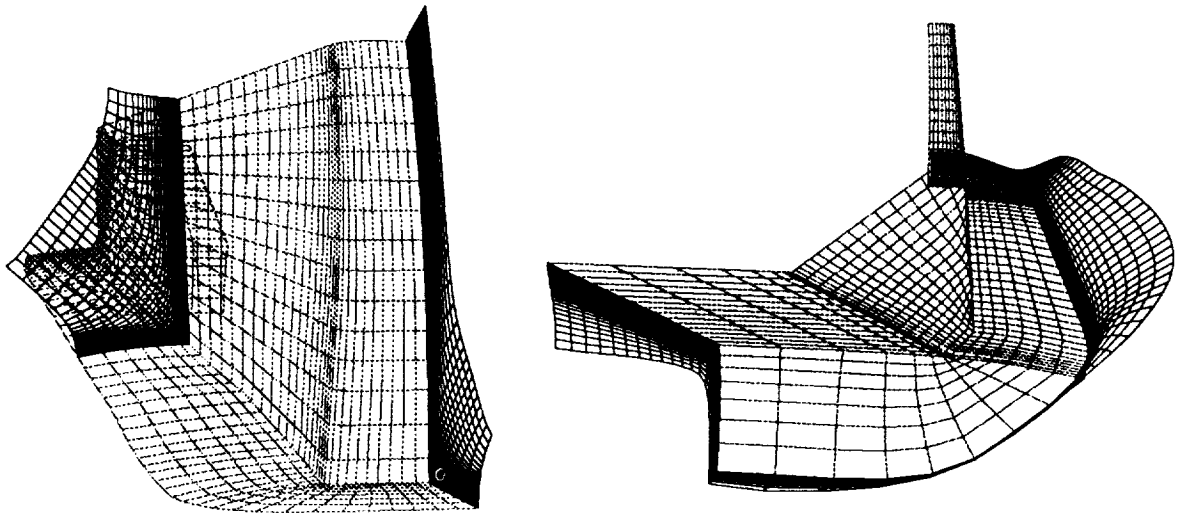
16. Parks, S. J., Buning, P. G., Steger, J. L. and Chan, W. M., *Collar Grids for Intersecting Geometric Components Within The Chimera Overlapped Grid Scheme.* AIAA Paper 91-1587, AIAA 10th Computational Fluid Dynamics Conference, Honolulu, Hawaii, 1991.

17. *SOFIA Technology Progress Briefing*, NASA-Ames Research Center, Dec. 5, 1990.



(a) (b)

Figure 9. Views of the collar grid surface. (a) Position relative to vertical tail and orbiter, (b) close up view relative to vertical tail and orbiter.



(a) (b)

Figure 10. Views of sections of the 3D collar grid joining the vertical tail and the orbiter. (a) Front view, (b) back view.

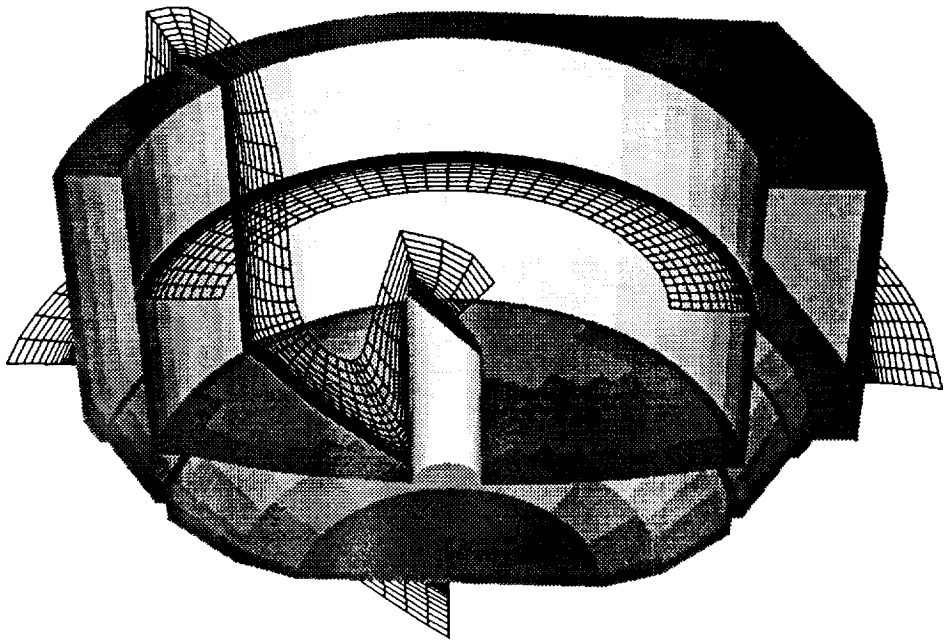


Figure 11a. Surface geometry and slices of the 3D grid for the SOFIA telescope.

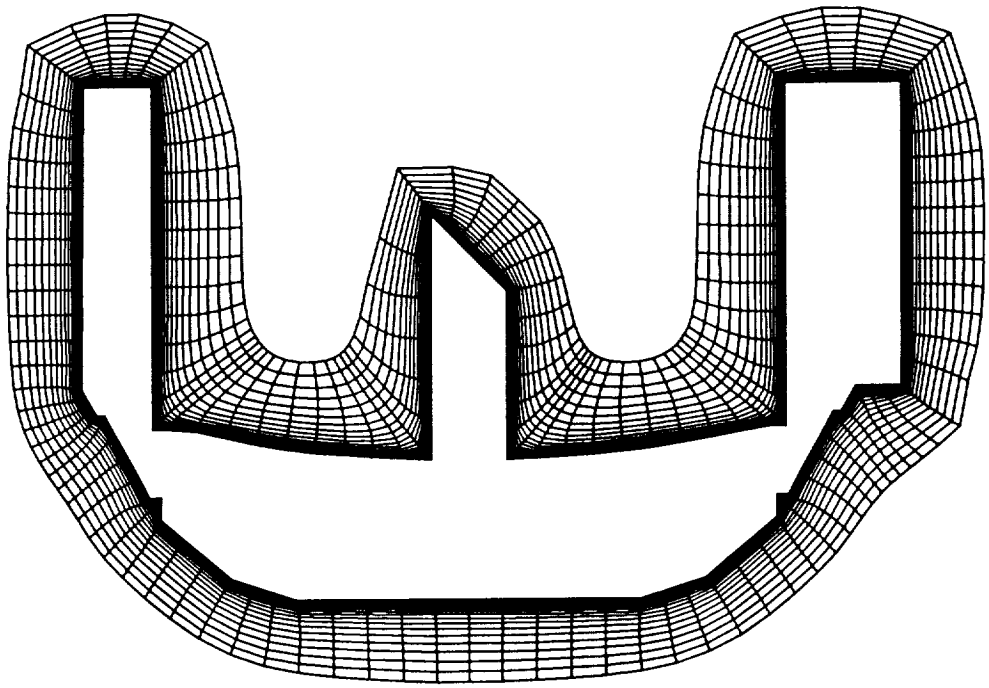


Figure 11b. Symmetry plane of the 3D grid for the SOFIA telescope.

APPENDIX B

AIAA-91-1587

Collar Grids for Intersecting Geometric Components Within the Chimera Overlapped Grid Scheme

S. Parks
Lockheed Engineering & Sciences Company
Houston, TX

P. Buning and W. Chan
NASA-Ames Research Center
Moffett Field, CA
J. Steger
University of California
Davis, CA

AIAA 10th Computational Fluid Dynamics Conference
June 24-26, 1991 / Honolulu, HI

COLLAR GRIDS FOR INTERSECTING GEOMETRIC COMPONENTS WITHIN THE CHIMERA OVERLAPPED GRID SCHEME

Steven J. Parks*
Lockheed Engineering & Sciences Company
Houston, Texas 77258
Pieter G. Buning**
NASA Ames Research Center
Moffett Field, California 94035
Joseph L. Steger†
University of California, Davis
Davis, California 95616
William M. Chan‡
NASA Ames Research Center
Moffett Field, California 94035

Abstract

For computational fluid dynamics simulations of flow about complex geometries, the Chimera overset grid scheme provides a conceptually simple method for domain decomposition. Overlapping grids are generated about individual geometric components, and interpolation is used to communicate boundary information between grids. However, for viscous flow computations about intersecting pieces of geometry (such as a wing and fuselage), generation of suitable grids and interpolation stencils in the intersection region is not straightforward. Problems such as resolution of the intersection region and appropriate definition of the geometric surface with respect to both grids must be addressed.

A method is presented to resolve these issues by the creation of a "collar grid" which resolves the intersection region and effectively connects the related component grids. The surface and 3D grid generation process for the collar grid is described, with the resultant effect on interpolations at the grid and hole boundaries. The method is applied to a simple wing/body geometry, and results are compared with experimental data. Application to the vertical tail of the Space Shuttle Orbiter is also shown.

1. Introduction

In earlier work [1-3], flow about the Space Shuttle launch vehicle has been simulated using the Chimera overset grid approach [4-7] and a thin-layer Navier-Stokes flow solver, F3D [8-9] (Fig. 1). In this work and subsequent studies, the effect of secondary geometrical features such as the Orbiter/External Tank attach hardware has been found to be vital to calculation of accurate surface pressures. A combination of fine grid spacing in the viscous direction normal to the body surface and the use of trilinear interpolation by the AEDC PEGASUS software [10] for grid boundary communication prevented the modeling of attach hardware that actually touched the Orbiter or External Tank (ET). Instead, approximations to the attach hardware and fuel feed line geometries have been used which "float" between the two (Fig. 2).

While this approach has allowed the crucial blockage effect of the attach hardware to be included in the calculations, requirements on the accuracy of Orbiter wing load predictions dictates that a more realistic representation of the attach hardware and other protuberances be included. Indeed, for the Chimera scheme to be able to model even so simple a "complex geometry" as a wing/body, a routine method for modeling intersecting geometry components must be developed.

The Chimera scheme allows for the simple merging of 3D grids by oversetting them onto a main grid. Surface modeling and grid resolution locations can be considered independently of other grids within the system, and 3D grids can be easily generated, for instance with hyperbolic grid generators [11-12]. Because grids are generated somewhat independently, points from one grid can fall within another body boundary. Such points must be excluded from the flow computation. Holes are thus cut in the grids to allow for the body boundaries of the other grids, creating hole boundaries in addition to the outer boundaries of the grids. Flow information is passed back and forth over these boundaries using interpolation stencils.

When bodies intersect, with each intersecting body having its own individual grid, hole cutting can be particularly difficult. One problem is finding interpolation

*Senior Associate Engineer
**Research Scientist, Member AIAA
†Professor, Mechanical, Aeronautical and Materials Engineering, Associate Fellow AIAA
‡Research Scientist, MCAT Institute

stencils for hole boundaries in the intersection region. The other difficulty, encountered with high Reynolds number viscous flow simulation, is that the grid spacing used to resolve the viscous boundary layer is generally much finer than the accuracy with which the body surface is represented by the discrete computational grid. Consequently, a code such as PEGASUS, which must determine whether points from one grid are inside or outside of a discretely defined body surface, may make the wrong determination. Because PEGASUS uses trilinear interpolation, it will generally be unable to find interpolation stencils for points very close to a concave surface, determining them to be inside the body surface. For a convex surface, interpolation stencils for nearby points will reflect a position farther from the surface than the actual location.

The introduction of a collar grid serves to overcome these limitations. In this approach, a grid is added in the region of the geometry intersection. This grid provides communication between the grids about the intersecting geometry components, and also resolves the intersection region. To avoid problems with viscous grid spacing, edges of the collar surface grid are created based upon the geometry surfaces as defined by trilinear interpolation of the component surface grids. Hole boundary points in the component grids must be moved to the surface as defined by the collar grid as well. In this way, interpolation stencils generated by the PEGASUS grid joining software will correctly reflect the relative locations of the component and collar grids. The component and collar grids can thus be generated independently of each other, with a minimum of modification for the interpolation process. The method requires no changes to the Chimera scheme or to the flow solver.

In following sections, details of this method will be illustrated using a two-dimensional circle/plane example and a three-dimensional cylinder intersecting a curved surface. Computed flow results will be presented for a simple wing/body combination and compared with experiment. Finally, addition of the Orbiter vertical tail is demonstrated for a computation of flow about the Space Shuttle launch vehicle. Comparisons are made with both wind tunnel and flight measured pressures.

2. Flow Solver

The F3D flow solver [8-9] has been used for these calculations. F3D is an implicit finite difference code for solving the thin-layer approximation to the Navier-Stokes equations. Flux vector splitting is used in the ξ-direction, while central differencing is used in η and ζ. The Baldwin-Lomax algebraic turbulence model is used [13].

In the Chimera approach, parts of some grids may be cut out, or eliminated from the computational domain. Such points might lie within another geometry component or simply not resolve the flow as well as another grid present in the same area. This elimination is accomplished in the flow solver by using an array of blanking values $ib = 0$ or 1 which multiply the timestep at each point. Thus where $ib = 0$ there is no change to the flow variables. These points are said to be "blanked out." The value of ib is set to zero within holes and also at hole and outer boundaries where interpolations from other grids will be used to update the solution. At interior points, $ib = 1$.

In this implementation then, at any grid point where $ib = 1$, data on either side is valid, i.e., is either an interior point or a boundary point. Thus any three-point differencing stencil will return valid information. For five-point stencils however, further modification of the flow solver is required. This is true for the second-order accurate flux vector split Euler terms and the fourth-order smoothing. By using the value of ib at neighboring points when forming the differences, accuracy is reduced to first order for the flux split Euler terms, and the fourth-order smoothing reverts to second-order smoothing [1]. This occurs in a manner similar to what happens when approaching the edge of the computational grid.

Boundary conditions for the F3D routine are implemented as calls to modular routines, including viscous wall, axis, symmetry plane, and extrapolated outflow conditions. These conditions are applied explicitly, both before and after each iteration on a grid. Chimera interpolations for hole and outer boundaries are also updated as explicit boundary conditions, and are performed before the standard boundary conditions are applied.

During the solution process, the F3D code cycles through the grids, reading in flow and grid data from the previous iteration and Chimera interpolation pointers for the current grid. At this point, an iteration or iterations may be taken independent of the other grids. Updated flow information is written out and the next grid is processed.

3. Chimera for Intersecting Components

To provide an illustration of the Chimera overset grid scheme as well as an example of the collar grid method, we shall consider the case of a cylinder intersecting a curved surface. This geometry is presented in Fig. 3, which shows parts of the grids for each component. In Fig. 4, grid points which lie inside the other component are blanked out, leaving holes in the cylinder and surface in the region of the intersection. Fig. 5 shows resulting hole boundaries for the curved surface grid (5a) and cylinder grid (5b). These are the points

which receive interpolated flow information from the other grid.

Before delving into the method for filling in the intersection region, we note that this outlines the basic Chimera grid joining strategy. For the results presented in this paper, the process of cutting holes and finding interpolation stencils was performed by the PEGASUS code (version 3.01) from Arvin/Calspan at the Air Force Arnold Engineering Development Center (AEDC) [10]. Several other methods for joining overlapped grids have also been developed (see for example Refs. 14-15).

In order to present the details of the collar grid approach, we now consider a two-dimensional example of a circle and a plane (Fig. 6). Fig. 7 shows the holes cut in both grids. The basic concept of the collar grid is illustrated in Fig. 8 as a grid whose boundary extends from the intersection point onto each component (circle and plane), and which covers the area between the component grids in the region of the intersection. (In this case the surface of the collar grid has been made to conform to the discrete representation of the circle.) In Fig. 9, the outer boundary surface of the collar grid is shown to lie on the discrete definition of the circle. While a wall boundary condition will be applied to the surface point, outer boundary points above must be interpolated from the underlying circle grid. These points are seen here to lie within appropriate cells of the circle grid.

Proper outer boundary communication for the collar grid has thus been established by forcing the last surface point of the collar grid to lie on the surface of the circle, as represented by the surface of the circle grid. The other inter-grid communication that must be established is at the hole boundaries of the component grids in the neighborhood of the intersection. In Fig. 10 we can see that the surface of the circle grid at the hole boundary does not lie exactly on the discrete collar grid representation of the circle. While this offset is generally small, it must be compared to the extremely fine grid spacing normal to the surface necessary to compute high Reynolds number flows. Spacing in the normal direction can easily be 10^5 times smaller than in the other directions. A closeup of the hole boundary of the circle grid is shown in Fig. 11, illustrating that an interpolation stencil for the first point off the surface would come from several grid cells away from the surface in the collar grid. This transfer of information would not accurately represent the flow profile at this point. To correct this, the surface point at the hole boundary is moved onto the surface of the collar grid, and points above are moved a similar amount. (Alternatively, the surface point can be moved and the full grid regenerated.) With this correction, hole boundary points in the circle grid will receive realistic interpolations of the flow data from the collar grid.

This procedure works as long as the outer boundary of the collar grid is at least two points away from the hole boundary in the circle grid, such that motion of the hole boundary point does not affect the position of the surface at the collar outer boundary. The fact that the component grid must be changed is undesirable; a scheme where creation of the collar had no effect on the other grids would be much preferable. However, this approach does have the benefit of not changing the current implementation of the flow solver or the PEGASUS grid joining code.

We note here that some form of tricubic interpolation could be used, allowing a more accurate representation of the surface. In this case, the movement of the hole boundary points would be greatly reduced (but would still be necessary). This would also improve the interpolation quality for the flow information, but would incur the overhead of using a 64-point stencil (in 3D), rather than an 8-point stencil in the flow solver.

4. Collar Grid Generation

To demonstrate the process of generating the collar grid, we return to the three-dimensional example of the cylinder and curved surface (a simplified model of the Shuttle External Tank and liquid hydrogen feed line). Before generating a three-dimensional grid, a surface grid must be defined, and before that, the intersection line. The intersection of the components can be determined by a variety of methods. This is a typical function of a CAD system, but may also be determined from the surface grids of the components. In this case, intersection points along grid lines running the length of the cylinder grid are determined, and form the description of the intersection line. These points may be splined and a new set of points generated if further refinement of the intersection is desired.

For generating the surface grid for the collar, and for other surface grid generation tasks, a hyperbolic surface grid generation program has been created [16-17]. This program generates a single plane (x, y, z) grid by marching out from the initial intersection line onto one of the component surfaces. One of the usual hyperbolic grid generation orthogonality conditions is replaced by a condition of orthogonality to the local surface normal of the component surface grid. After each step, the new row of points is explicitly projected onto the component surface grid to insure adherence to the surface. (The surface itself is currently defined by bilinear interpolation of the component surface grid.) The grid is marched out far enough to cover the hole in the curved surface grid and provide adequate overlap for interpolation. Fig. 12 shows the resulting sur-

face grid on the curved surface. This may be repeated for the cylinder, or a simple stretching of points from the intersection point along cylinder grid lines may be used. When the two grid sections are joined, the result is a surface grid covering the intersection region and adhering to each component surface (Fig. 13).

Once a surface grid has been created, a 3D collar grid is generated using a hyperbolic grid generator [12], where initial wall spacing is specified and the grid is created by marching out until the outer boundary is a sufficient distance from the surface. Slices of the 3D collar grid are shown in Fig. 14.

5. Validation and Results

A wing/body combination was selected to exercise the collar grid approach and validate the method. The particular geometry and experimental data used for comparison are described in Ref. 18, and were chosen because of the presence of measurements in the wing/body junction region. Flow conditions for the experiment were Mach 0.8 and 2° angle-of-attack. The Reynolds number was one million based on the geometric mean chord, tripped at 12.5% chord on the wing and fully turbulent on the body.

The overall geometry is illustrated in Fig. 15, and consists of a simple axisymmetric body with a swept, tapered wing based on an uncambered RAE 101 airfoil. The wing has no dihedral or twist. The same flow conditions as the experiment were used for the computation. The surface grid for the collar, shown in Fig. 16, uses a C-grid topology on the wing, and fans out onto the fuselage. The 3D collar grid is thus "grown" out from this surface using the hyperbolic grid generator. Hole boundaries in the wing and body grids are shown in Fig. 17.

Comparisons of surface pressure coefficients in the neighborhood of the wing/body junction are presented in Fig. 18 for wing section cuts, and in Fig. 19 for fuselage axial cuts. Excellent agreement has been obtained for all areas of the geometry, except near the leading edge of the wing where the turbulence model was turned on abruptly. Modeling of a finite length transition region would be more physically correct, and would reduce the sudden pressure change at this point. Following transition, the computed solution rapidly recovers to again match the experimental data. (The turbulence model was turned on at 8.5% chord, rather than 12.5% where the boundary layer trip was located on the model.)

While work on applying the collar grid scheme to the Shuttle attach hardware is in progress, the method has been used to include the Orbiter vertical tail in the full launch vehicle calculations. Earlier computations had neglected the vertical tail, both because of the difficulty in gridding the geometry and the feeling that at zero sideslip the tail would have minimal effect on Orbiter wing and fuselage surface pressures.

The extent of the collar grid on the Orbiter surface is shown in Fig. 20. At the end of the fuselage, the surface grid folds down onto the aft bulkhead (not shown). One of several computations has been made at a Mach number of 1.1 and angle-of-attack of $-3.9°$ (set to correspond to a flight test data point for wing pressure comparisons). Surface pressure contours are plotted in the region of the vertical tail in Fig. 21. A smooth transition is made between the fuselage and collar grids; a small jog is evident over part of the boundary between the collar and tail grids. This is believed to be due to the somewhat coarser tail grid being allowed to come too close to the fuselage intersection. If this is the case, the remedy would be to increase the size of the hole in the vertical tail grid, and extend the collar grid farther up the tail to maintain the desired grid overlap.

Limited pressure data is available for the vertical tail from the first four Shuttle flights [19]. Wind tunnel data is also available [20], and both are compared with the computed solution in Fig. 22, at a span station of $Z/S = 0.309$. The computation was made at the wind tunnel Reynolds number of 4×10^6 per foot, based on the 3% scale model. Wind tunnel data shown is from $-4°$ angle-of-attack. Comparison between computation and wind tunnel is quite good for the forward portion of the chord. Differences past that point may be attributed to lack of modeling in the computation of the main engine bells and Orbital Maneuvering System (OMS) pod extensions past the aft bulkhead. The flight data includes the effect of main engine and Solid Rocket Booster plumes.

Finally, the effect of the tail on fuselage and wing surface pressures can be seen in Fig. 23. Additional blockage from the tail causes a significant increase in the high pressure region ahead of the OMS pods, and a slight change in inboard wing pressures.

6. Summary

A method has been developed to overcome problems with using the Chimera overset grid scheme in the region of intersecting geometry components. This is accomplished by introducing a "collar grid" which resolves the intersection region and provides communication between the component grids.

Several examples have been presented which illustrate the method. Excellent comparison of computed and experimental data for flow about a wing/body configuration serves as a validationofthe approach. Finally, application of the collar grid scheme to the Orbiter fuselage and vertical tail intersection in a computation of the full Space Shuttle launch vehicle demon-

strates its usefulness for simulation of flow about complex aerospace vehicles.

Acknowledgements

The authors wish to thank Fred Martin, Jr., of the NASA Johnson Space Center for his technical assistance and encouragement. Financial support was provided by the National Space Transportation System Program Office and NASA Johnson Space Center. Computational support was provided by the NASA Johnson Space Center Engineering Computational Facility and the NASA Ames Numerical Aerodynamic Simulation Facility.

References

1. P.G. Buning, I.T. Chiu, S. Obayashi, Y.M. Rizk, and J.L. Steger, "Numerical Simulation of the Integrated Space Shuttle Vehicle in Ascent," AIAA-88-4359-CP, AIAA Atmospheric Flight Mechanics Conference, August 15-17, 1988, Minneapolis, Minnesota.

2. P.G. Buning, I.T. Chiu, F.W. Martin Jr., R.L. Meakin, S. Obayashi, Y.M. Rizk, J.L. Steger, and M. Yarrow, "Flowfield Simulation of the Space Shuttle Vehicle in Ascent," Proceedings of the Fourth International Conference on Supercomputing, Santa Clara, California, April 30-May 5, 1989.

3. F.W. Martin, Jr. and J.P. Slotnick, "Flow Computations for the Space Shuttle in Ascent Mode Using Thin-Layer Navier-Stokes Equations," **Applied Computational Aerodynamics** (Progress in Astronautics and Aeronautics, Vol. 125), P.A. Henne, ed., American Institute of Aeronautics and Astronautics, Washington, D.C., 1990, pp. 863-886.

4. J.L. Steger, F.C. Dougherty, and J.A. Benek, "A Chimera Grid Scheme," **Advances in Grid Generation**, K.N. Ghia and U. Ghia, eds., ASME FED Vol. 5, 1983, pp. 59-69.

5. J.A. Benek, P.G. Buning, and J.L. Steger, "A 3-D Grid Embedding Technique," AIAA-85-1523-CP, July 1985.

6. J.A. Benek, T.L. Donegan, and N.E. Suhs, "Extended Chimera Grid Embedding Scheme with Application to Viscous Flows," AIAA-87-1126-CP, June 1987.

7. J.L. Steger and J.A. Benek, "On the Use of Composite Grid Schemes in Computational Aerodynamics," Computer Methods in Applied Mechanics and Engineering, Vol. 1, No. 4, Dec. 1987, pp. 431-437.

8. S.X. Ying, J.L. Steger, L.B. Schiff, and D. Baganoff, "Numerical Simulation of Unsteady, Viscous, High Angle of Attack Flows Using a Partially Flux-Split Algorithm," AIAA-86-2179, Aug. 1986.

9. J.L. Steger, S.X. Ying, and L.B. Schiff, "A Partially Flux-Split Algorithm for Numerical Simulation of Compressible Inviscid and Viscous Flow," Proceedings of the Workshop on Computational Fluid Dynamics, Institute of Nonlinear Sciences, University of California, Davis, Davis, California, 1986.

10. J.A. Benek, J.L. Steger, F.C. Dougherty, and P.G. Buning, "Chimera: A Grid-Embedding Technique," AEDC-TR-85-64, Arnold Engineering Development Center, Arnold AFS, Tennessee, April 1986.

11. J.L. Steger and Y.M. Rizk, "Generation of Three Dimensional Body Fitted Coordinates Using Hyperbolic Partial Differential Equations," NASA TM 86753, June 1985.

12. W.M. Chan and J.L. Steger, "A Generalized Scheme for Three- Dimensional Hyperbolic Grid Generation, AIAA-91-1586-CP, AIAA 10th Computational Fluid Dynamics Conference, Honolulu, Hawaii, June 24-26, 1991.

13. B.S. Baldwin and H. Lomax, "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA-78-257, Jan. 1978.

14. G. Chesshire and W.D. Henshaw, "Composite Overlapping Meshes for the Solution of Partial Differential Equations," J. Computational Physics, Sept. 1990.

15. R.L. Meakin, "A New Method for Establishing Inter-Grid Communication Among Systems of Overset Grids," AIAA-91-1586-CP, AIAA 10th Computational Fluid Dynamics Conference, Honolulu, Hawaii, June 24-26, 1991.

16. J.L. Steger, "Notes on Surface Grid Generation Using Hyperbolic Partial Differential Equations," Department of Mechanical Engineering, University of California, Davis, Computational Fluid Dynamics Group, Technical Memorandum CFD/UCD 89-101, Nov. 1989.

17. J.L. Steger, "An Introduction to Grid Generation Using Partial Differential Equations," Proceedings of the 3rd Joint Europe/U.S. Short Course in Hypersonics, RWTH, Aachen, FRG, October 1-5, 1990.

18. D.A. Treadgold, A.F. Jones, and K.H. Wilson, "Pressure Distribution Measured in the RAE 8ft x 6ft Transonic Wind Tunnel on RAE Wing 'A' in Combination with an Axi-Symmetric Body at Mach Numbers of 0.4, 0.8 and 0.9," AGARD-AR-138 (**Experimental Data Base for Computer Program Assessment**), 1979, Ref. B4.

19. Rockwell International Internal Letter SAS/AERO/83-094, "STS-5 Postflight Report on Ascent External Aerodynamic Loads," March 1983.

20. R.H. Spangler, "Results of Tests Using a 0.03 Scale Model (47-OTS) of the Space Shuttle Integrated Vehicle in the AEDC 16 Foot Transonic Propulsion Wind Tunnel (IA105A)," NASA CR 160851, Oct. 1981.

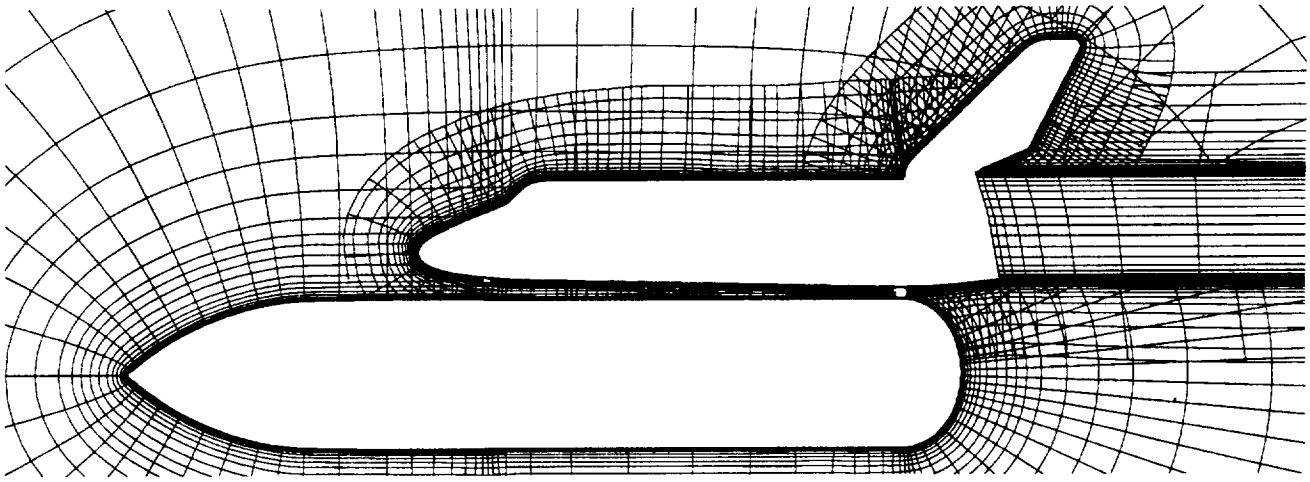
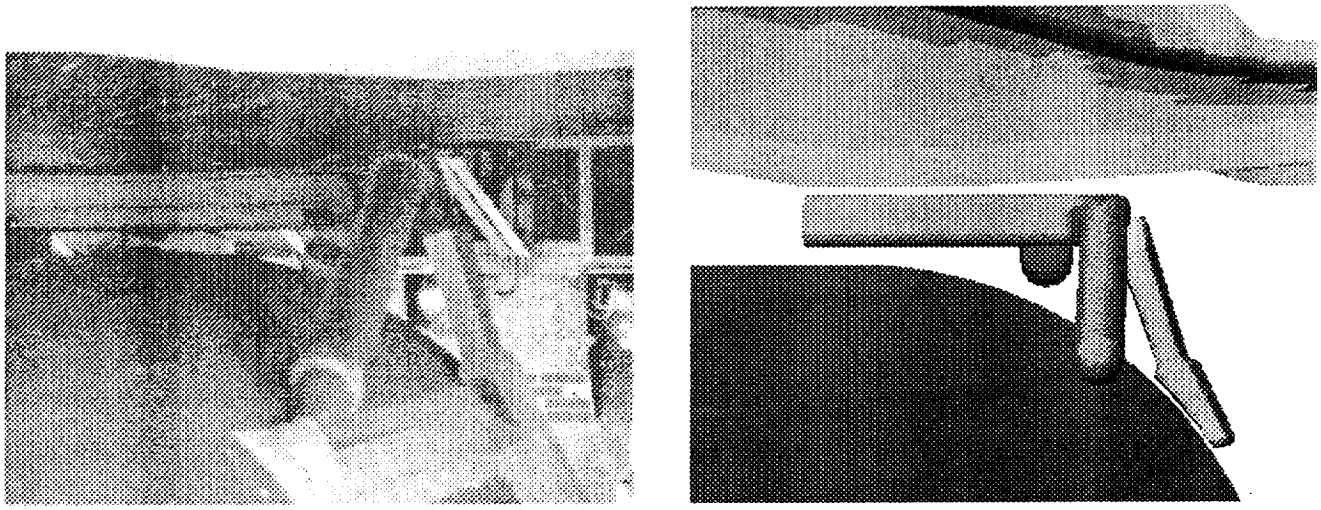


Figure 1. Overlapped grids for the Shuttle launch vehicle configuration.



(a) flight hardware

(b) computational model

Figure 2. Orbiter/ET aft attach hardware.

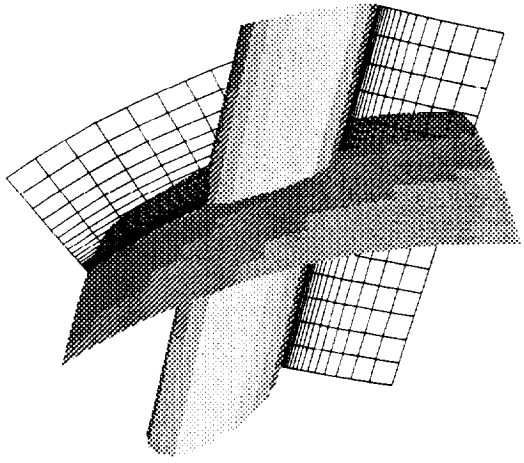


Figure 3. Cylinder and curved surface geometry and grids.

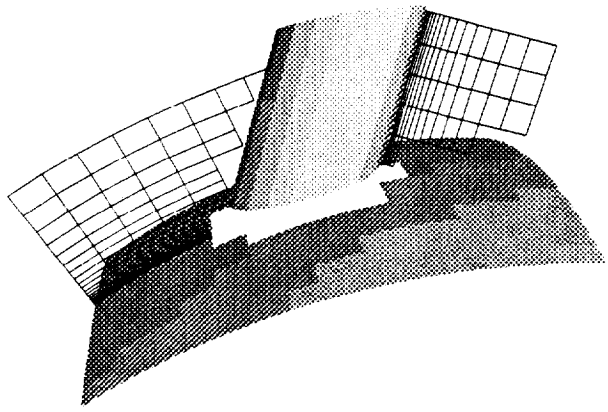


Figure 4. Grids after cutting holes.

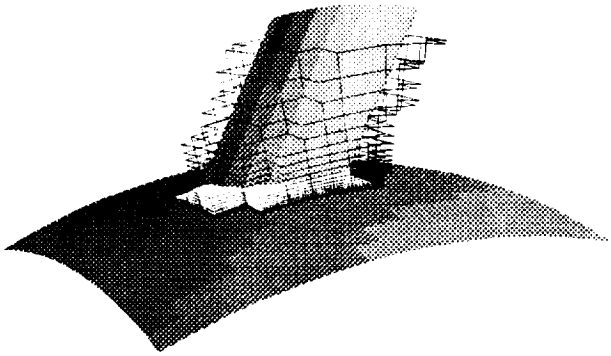


Figure 5(a). Boundary of hole cut in curved surface grid.

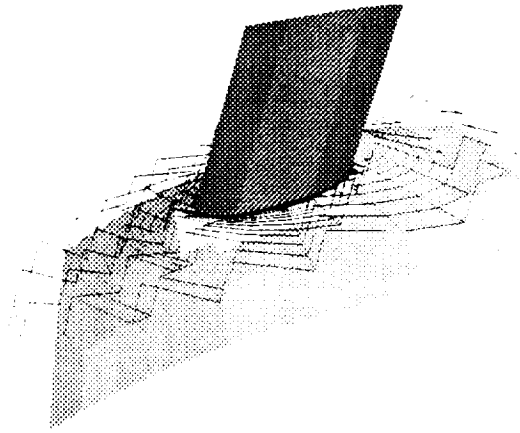


Figure 5(b). Hole boundary for cylinder grid.

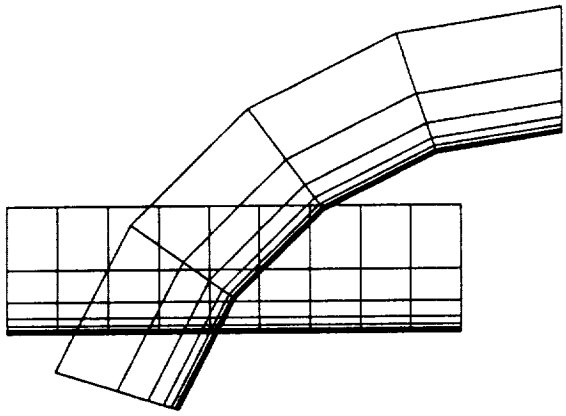


Figure 6. 2D circle and plane grids.

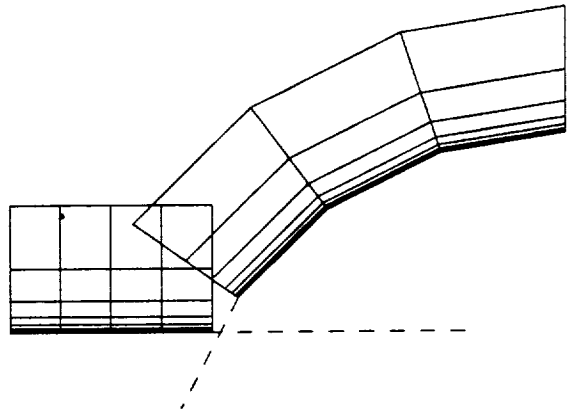


Figure 7. Grids with holes cut.

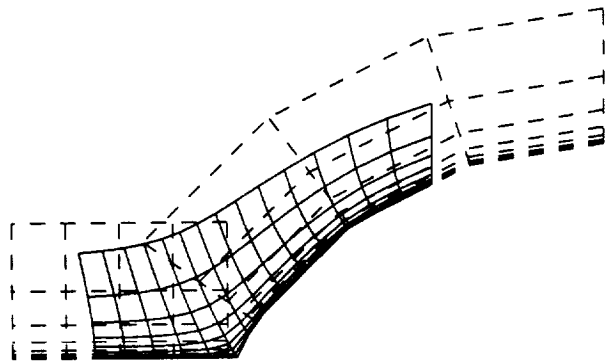


Figure 8. Collar grid filling in intersection region.

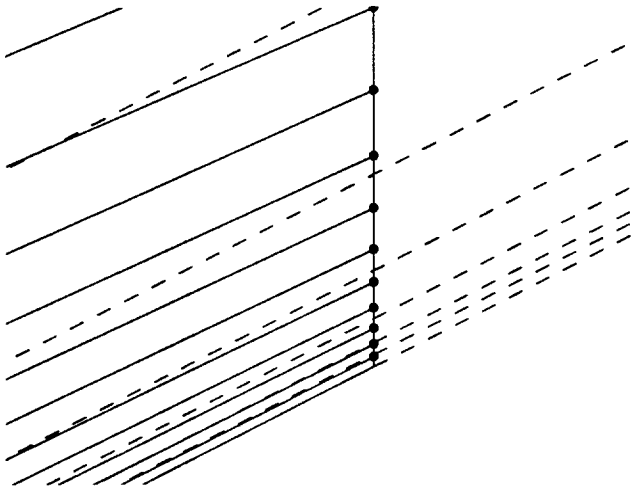


Figure 9. Filled symbols mark outer boundary points of (solid) collar grid which require interpolated flow data from (dashed) circle grid.

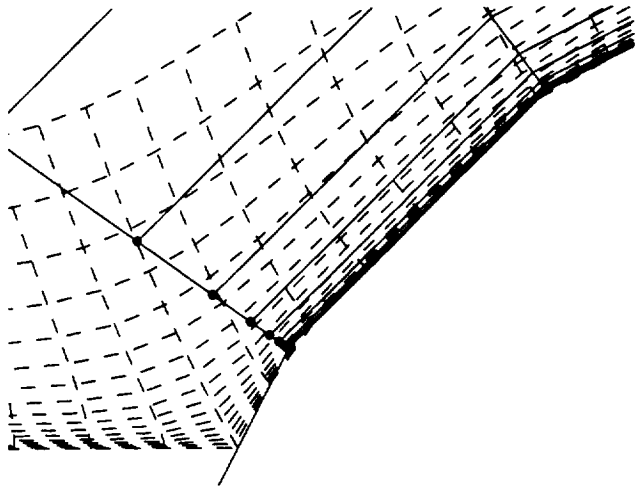


Figure 10. Hole boundary points of (solid) circle grid which require interpolated information from (dashed) collar grid.

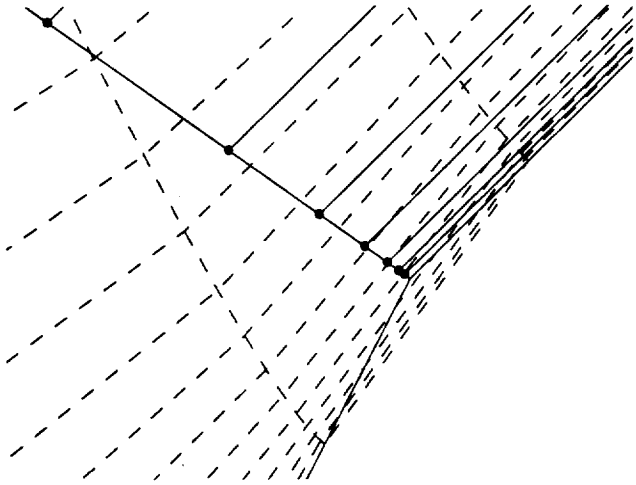


Figure 11. Closeup of correction needed to hole boundary point.

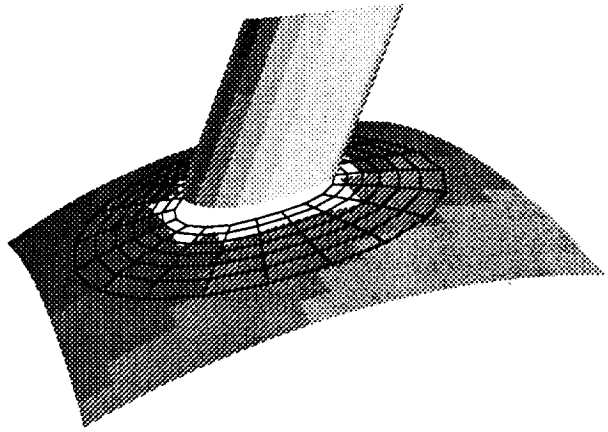


Figure 12. Hyperbolic surface grid generated from the intersection line onto the curved surface.

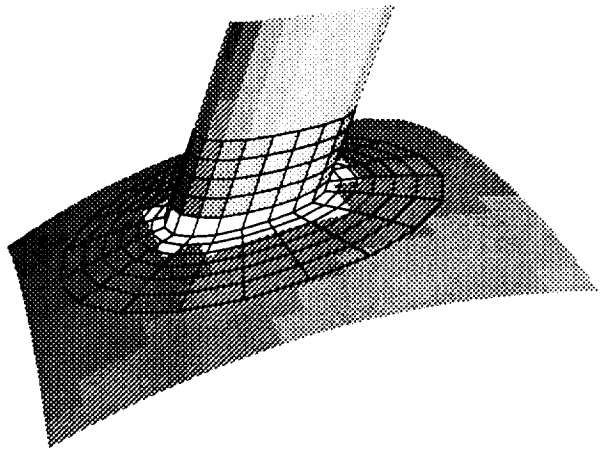


Figure 13. Combined collar surface grid.

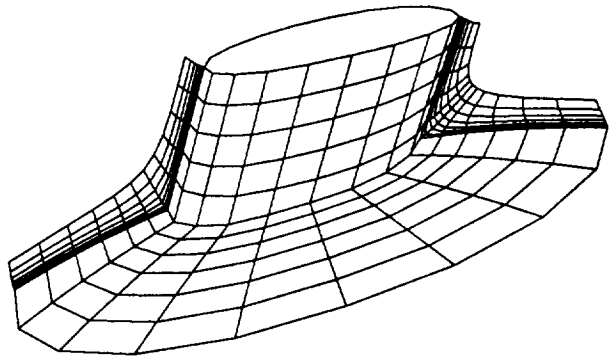


Figure 14. Slices of the completed collar grid.

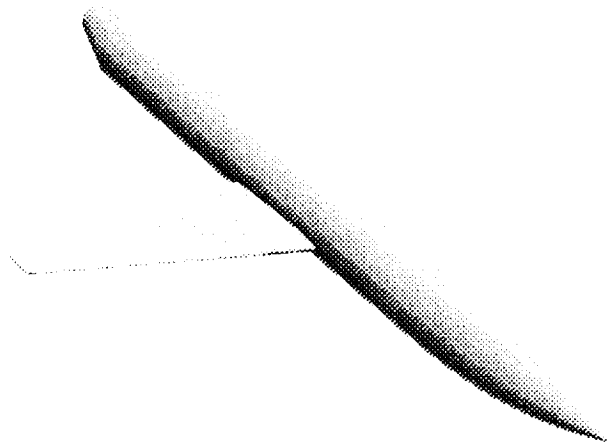


Figure 15. Wing/body geometry.

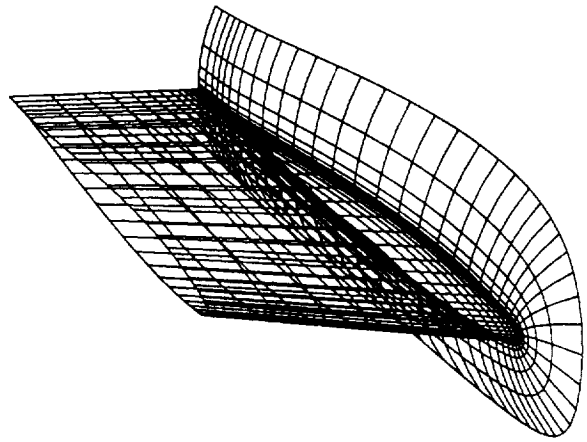


Figure 16. Collar surface grid for wing/body intersection.

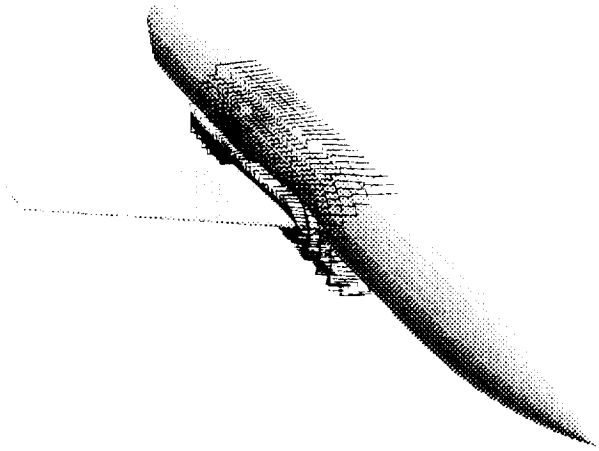


Figure 17(a). Boundary of hole cut in wing grid.

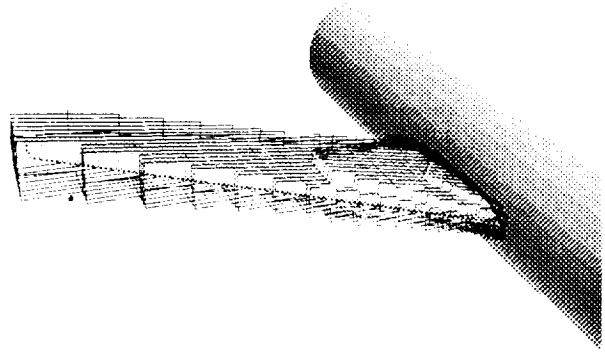


Figure 17(b). Hole boundary of body grid.

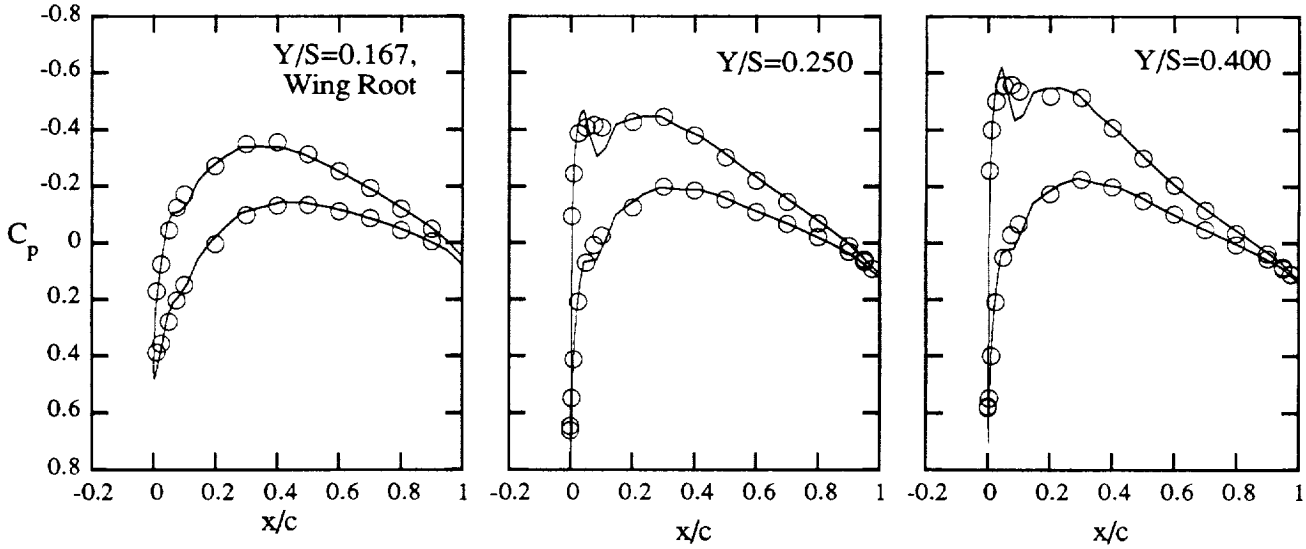


Figure 18. Comparison of surface pressure coefficient on wing section cuts, ○ experimental results (Ref. 18), — present computed results. Turbulence model was turned on abruptly at 8.5% chord.

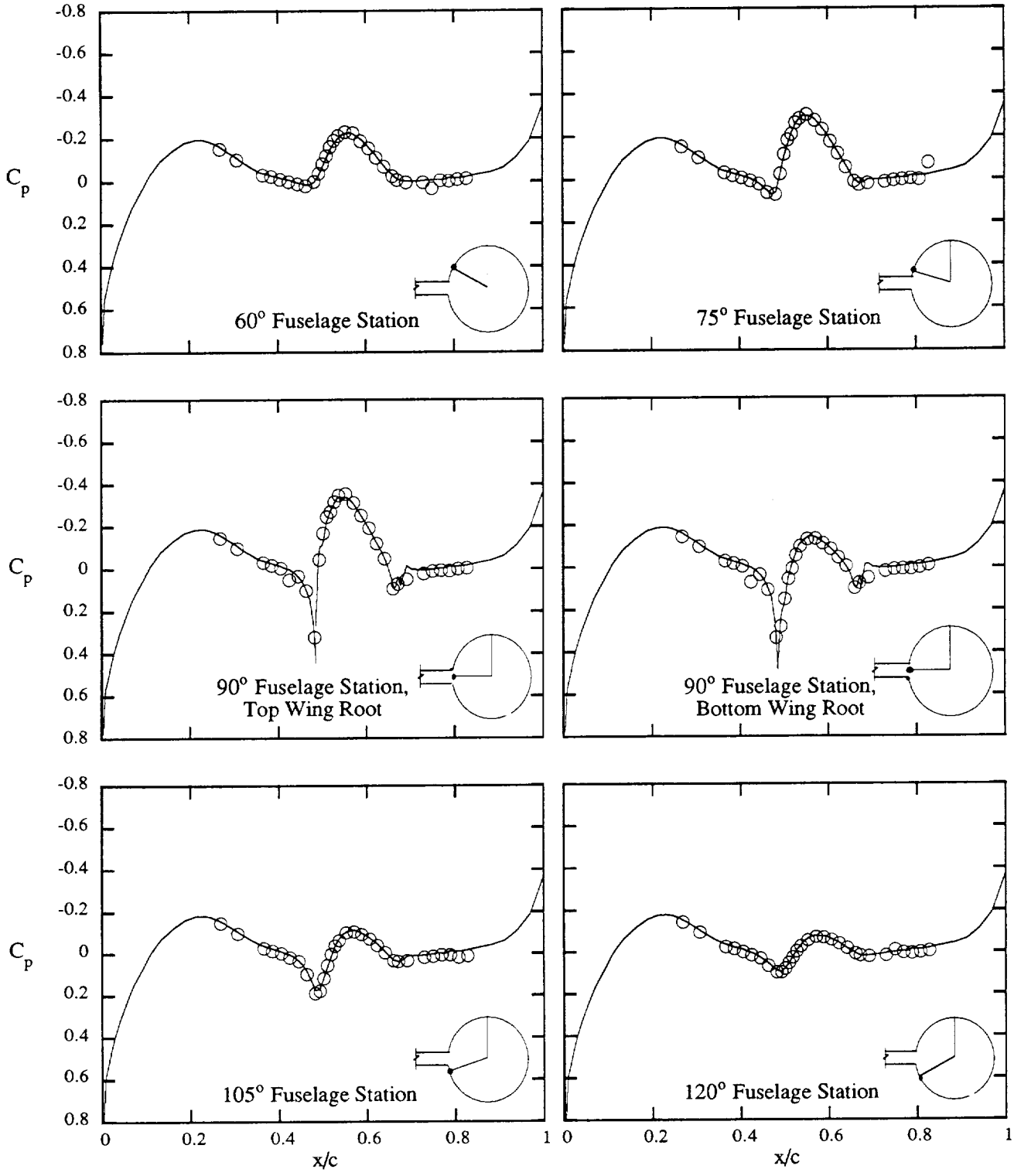


Figure 19. Surface pressure coefficient comparison for several fuselage axial cuts, ○ experimental results (Ref. 18), — present computed results.

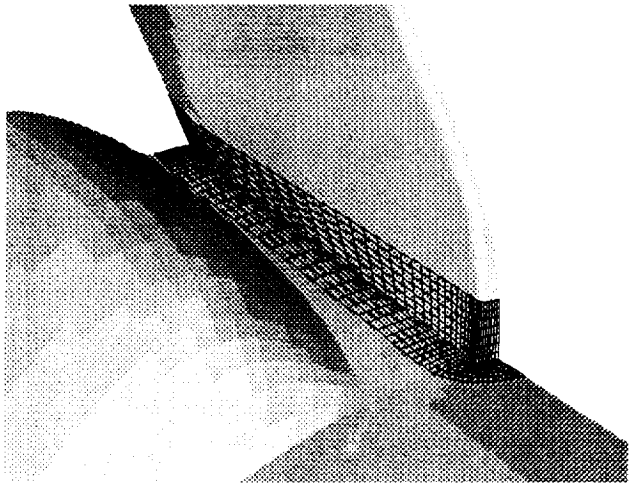


Figure 20. Collar surface grid for the Orbiter vertical tail.

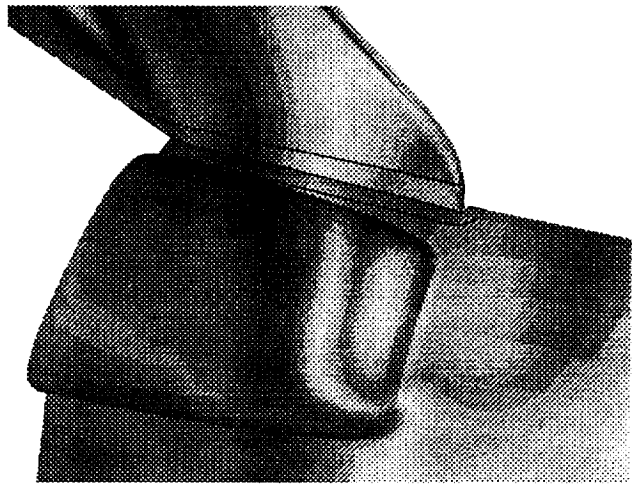


Figure 21. Pressure contours in the Orbiter vertical tail region.

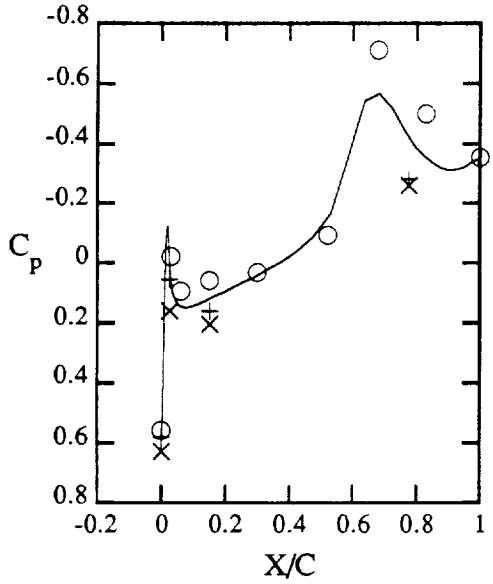


Figure 22. Comparison of pressure coefficient on the vertical tail for a chordwise cut at $Z/S = 0.309$; — present computation, ○ wind tunnel data (Ref. 20), and flight data minimum (+) and maximum (×) (Ref. 19).

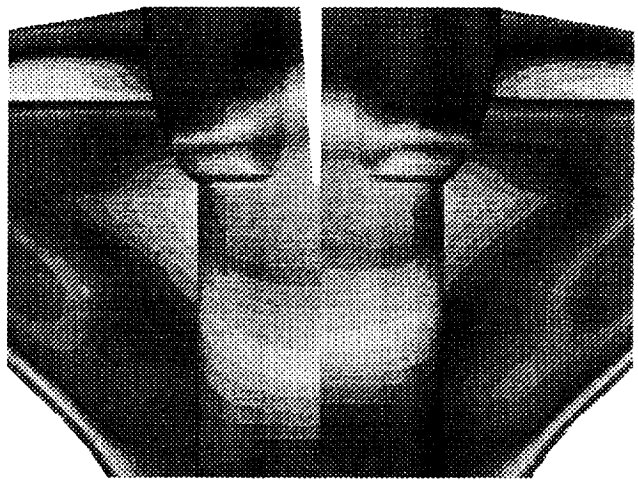


Figure 23. Top view of Orbiter fuselage and inboard wing surface pressure contours, (left) with vertical tail and (right) without.

APPENDIX C

APPLICATION OF THE CHIMERA OVERLAPPED GRID SCHEME TO SIMULATION OF SPACE SHUTTLE ASCENT FLOWS

Pieter G. Buning
NASA Ames Research Center
Moffett Field, California 94035 USA

Steven J. Parks
Lockheed Engineering & Sciences Company
Houston, Texas 77258 USA

William M. Chan
MCAT Institute, NASA Ames Research Center
Moffett Field, California 94035 USA

Kevin J. Renze
Department of Aerospace Engineering, Iowa State University
Iowa 50011 USA

Preprint for the Proceedings of the
4th International Symposium on Computational Fluid Dynamics
September 9-12, 1991
Davis, California USA

APPLICATION OF THE CHIMERA OVERLAPPED GRID SCHEME TO SIMULATION OF SPACE SHUTTLE ASCENT FLOWS

Pieter G. Buning,† Steven J. Parks,‡ William M. Chan,* and Kevin J. Renze**
†NASA Ames Research Center, Moffett Field, California 94035 USA
‡Lockheed Engineering & Sciences Company, Houston, Texas 77258 USA
*MCAT Institute, NASA Ames Research Center, Moffett Field, California 94035 USA
**Department of Aerospace Engineering, Iowa State University, Iowa 50011 USA

ABSTRACT

In this paper, several issues relating to the application of Chimera overlapped grids to complex geometries and flowfields are discussed. These include the addition of geometric components with different grid topologies, gridding for intersecting pieces of geometry, and turbulence modeling in grid overlap regions. Sample results are presented for transonic flow about the Space Shuttle launch vehicle. Comparisons with wind tunnel and flight measured pressures are shown.

INTRODUCTION

During the past four years, simulations of flow about the Space Shuttle launch vehicle have been made using the Chimera overlapped grid scheme [1-4]. Adoption of the Chimera scheme [5-7] allowed discretization of the volume about the complex Shuttle vehicle as a collection of structured grids. The overlapped grid approach was chosen over a patched scheme to reduce the labor time involved in domain decomposition. These simulations have used an implicit, approximately factored finite difference procedure to model the three-dimensional thin-layer Navier-Stokes equations.

The initial geometry simulated included the three major components: the Orbiter, External Tank (ET), and Solid Rocket Boosters (SRBs). Attach hardware between the components, fuel feed lines, and even the Orbiter vertical tail were ignored in the interest of keeping the problem tractable and learning the problems and pitfalls of the Chimera scheme. Simplified forward and aft attach hardware between the Orbiter and ET, and the vertical tail were then added using overlapped grids for each component. Also, SRB and Orbiter stings were filled in with additional grids. Elevon deflections and the SRB attach ring were incorporated by modifying the existing Orbiter and SRB grids. Current grid systems contain about 1.6 million points in 14 grids (Figs. 1-2). Sample wing pressures are shown in Fig. 3, comparing computed results with wind tunnel [8] and flight data [9] for Mach 1.25, $-5.1°$ angle-of-attack.

These CFD computations provide reasonable wing pressures (average differences with wind tunnel c_p are less than 0.1), and total integrated vehicle forces are close to flight-derived values. However, computed wing loads are significantly different from flight, and in the high dynamic pressure portion of the ascent, minimizing Orbiter wing loads is critical for Shuttle performance. Continuing efforts to reduce differences between flight-measured pressures and computations is forcing a reexamination of CFD procedures and assumptions. In addition, development of this capability into an engineering predictive tool is leading to a list of desired improvements. Some of these difficulties and improvements are discussed below.

GRID GENERATION AND JOINING

The overall philosophy behind the use of overlapped grids is that individual component grids can be generated easily if one does not have to match to other grids or pieces of geometry. The process of cutting holes in grids to allow for other components, and finding interpolation stencils to pass flow information between the grids is handled after the grids have been generated. This effectively joins the grids and provides communication between them. Component grid generation can be accomplished using hyperbolic grid generation [10], which is much faster than using an elliptic grid generator. The PEGASUS code from AEDC [11] has been used for the grid joining operation on many of the Shuttle applications, and a modified scheme which is significantly faster is also under development [12].

While this description makes application of the Chimera scheme sound quite simple, several considerations in the grid-joining process make the generation of an adequate grid system considerably more complex.

Mixed Grid Topologies

Overlapped grids are particularly useful for geometries which combine different topologies. One example is flow over a plate with an obstruction above it, shown in Fig. 4. The plate is particularly suited to a cartesian mesh,

while the obstruction is best fit with a cylindrical grid. This type of situation occurs in a number of places on the Shuttle, mostly where pieces of attach hardware approach the Orbiter or External Tank.

In this example, the plate represents the bottom surface of the Orbiter and the obstruction is the crossbeam of the aft attach. As illustrated, downstream spacing for the Orbiter grid, generally picked to resolve geometric features on the Orbiter, is quite large compared to the crossbeam. Again, the philosophy of generating each grid independently does not encourage consideration of other components when choosing grid spacing. An attempt to join these two grids results in a number of regions where there is no overlap between the grids (Fig. 5). Points flagged in the figure indicate "orphans," boundary points for which no interpolation stencil could be found. This example violates the "general rule-of-thumb" which accompanies PEGASUS, which is that grids should be of similar resolution at interface regions. This is easier said than done. When cartesian and cylindrical grids are merged, this discrepancy manifests itself in the regions fore and aft of the obstruction, between the bodies. " Stairsteps" in the hole cut in the cylindrical grid become too large to cover the hole in the cartesian grid. Basically, finer grids are called for.

In Shuttle work to date, significant investments have gone into generating grids for the major components, so that changing the grid resolution is not a trivial activity. Minimum increases to the downstream spacing in the major grids have been made, but the addition of attach hardware grids has still been a delicate operation. In order to ensure adequate resolution in the Orbiter grid, local downstream spacing on the order of the size of the gap between the Orbiter and crossbeam is suggested. A patched refined grid is an economical way to achieve the required tenfold increase in resolution. Figure 6 shows such a grid system with a sample solution at Mach 1.25.

Intersecting Geometric Components

If one accepts the rule-of-thumb that grids are of similar resolution where they are joined, and further that each grid adequately resolves flow gradients found in that grid, then interpolation is a reasonable way of passing boundary information. The PEGASUS code uses trilinear interpolation, the accuracy of which is generally not a problem. In cases where separately gridded geometric components intersect, however, resolution of the intersection region and generation of corresponding interpolation stencils is not a straightforward task. In order to avoid this situation,approximations to the Shuttle geometry have been made. Figure 7 shows how Orbiter/ET attach hardware has been modeled to "float" between the two bodies, without actually connecting to either. The concept of a collar grid has been introduced in Ref. 13, which allows the accurate representation of the surface geometry in light of the trilinear interpolation used at the grid boundaries. In Fig. 8, a simplified model of the Shuttle liquid hydrogen (LH2) feedline and the External Tank is shown, with a collar grid fitted to the intersection line. While this model of the feedline has not been incorporated into the full configuration yet, the Orbiter vertical tail has been included in this manner. The tail collar grid and corresponding surface pressures are shown in Fig. 9.

FLOW SOLVER

Algorithm Change

Improvements to both accuracy and speed of the flow solver affect the use of this simulation capability as an engineering tool on Shuttle-related problems. Acceleration of the code has allowed the timely completion of studies on the effect of SRB geometry modifications. In the Shuttle flow simulations carried out previously [1-4], the F3D thin-layer Navier-Stokes code was used. This code incorporated flux-vector splitting in one coordinate direction and central differencing in the other two. This allowed use of a two-factor implicit factorization of the left-hand side. However, significant computation was required to compute the flux-split terms for both the right- and left-hand sides.

Currently, computations of steady ascent flowfields are being generated using the Pulliam-Chaussee diagonalized algorithm as implemented in ARC3D [14-15]. Similar convergence behavior has been observed for cases run so far, while realizing a factor of three improvement in time per grid point per iteration over the F3D code. (In addition, a recoding of the F3D algorithm has resulted in a 35% reduction in its execution time.)

Turbulence Modeling

Another problem arises when solving for the turbulent flowfield over a multi-body configuration using several grids, such as used for the Shuttle launch configuration. Implementation of the Baldwin-Lomax algebraic turbulence model [16] requires searching out from the body surface for a length scale used in computing the turbulent eddy viscosity. If a grid boundary is encountered before the appropriate length is found, an erroneous eddy viscosity will be computed. Indeed, for overlapped grids, the turbulence model must know the locations of holes cut for other

grids, and should not search too close to another body where boundary layer vorticity might lead to a false peak in the function $F(y) \approx y\omega$. A sample profile of $F(y)$ between the Orbiter and ET is shown in Fig. 10, where each grid finds the location of F_{max} at an inappropriate location, close to the other body. Some method of computing reasonable levels of μ_t must be employed in order to resolve Reynolds number dependence of the flow. One such candidate follows the work of Degani and Schiff [17] by defining a cutoff level in the search for F_{max}, i.e., finalizing the choice of F_{max} when $F(y)$ drops below some fraction of the current maximum $F(y)$ found in the profile. This provides satisfactory results for the present example, but does not handle separation and turbulent wakes. One- and two-equation turbulence models are also being pursued to improve solution quality for such complex multi-body flowfields.

CONCLUSIONS

The Chimera overlapped grid approach to domain decomposition has allowed gridding and computation of flow about the Space Shuttle launch vehicle, without the time required for setting up a block-structured grid system. With improvements to the modeled geometry, accuracy of the computed surface pressures has become quite good from a CFD standpoint. However, accuracy required for wing loading analyses is significantly higher, and effective use of Chimera for geometry changes has proven difficult. Grid joining problems are being overcome by development of the collar grid technique, and by simply increasing grid resolution to a level commensurate with the size of local geometry features. Turbulence modeling can have a significant impact on pressure loads for a multi-body configuration. Difficulties with the Baldwin-Lomax model when using overlapped grids can greatly affect the calculation of turbulent eddy viscosity, and therefore the accuracy of pressure loads. Finally, switching from the F3D code to one based on the diagonalized ARC3D scheme has resulted in a three-fold speedup in solution time, at no penalty in convergence or accuracy.

ACKNOWLEDGEMENTS

The authors wish to acknowledge Prof. Joseph Steger of the University of California, Davis, and formerly of NASA Ames for his counseling and guidance of the Shuttle simulation effort over the years. Financial support for this work was provided by the National Space Transportation System Program Office and NASA Johnson Space Center. Computational support was provided by the NASA Ames Numerical Aerodynamic Simulation Facility and the NASA Johnson Engineering Computational Facility.

REFERENCES

1. P.G. Buning, I.T. Chiu, S. Obayashi, Y.M. Rizk, and J.L. Steger, AIAA-88-4359-CP (Aug. 1988).
2. P.G. Buning, I.T. Chiu, F.W. Martin Jr., R.L. Meakin, S. Obayashi, Y.M. Rizk, J.L. Steger, and M. Yarrow, "Flowfield Simulation of the Space Shuttle Vehicle in Ascent," Proceedings of the Fourth International Conference on Supercomputing, Apr. 30-May 5, 1989, Santa Clara, California.
3. R.L. Meakin and N.E. Suhs, AIAA-89-1996 (June 1989).
4. R.L. Meakin, "Transient Flow Field Responses About the Space Shuttle Vehicle During Ascent and SRB Separation," Store Carriage, Integration and Release Conference, Royal Aeronautical Society, Apr. 4-6, 1990, Bath, UNITED KINGDOM.
5. J.L. Steger, F.C. Dougherty, and J.A. Benek, *Advances in Grid Generation*, K.N. Ghia and U. Ghia, ASME FED-5, 59 (1983).
6. J.A. Benek, P.G. Buning, and J.L. Steger, AIAA-85-1523-CP (July 1985).
7. J.A. Benek, T.L. Donegan, and N.E. Suhs, AIAA-87-1126-CP (June 1987).
8. R.H. Spangler, NASA CR 160851 (Oct. 1981).
9. R.A. Machin, private communication, NASA Johnsons Space Center, Houston, Texas.
10. W.M. Chan and J.L. Steger, AIAA-91-1588-CP (June 1991).
11. J.A. Benek, J.L. Steger, F.C. Dougherty, and P.G. Buning, AEDC-TR-85-64, Arnold Engineering Development Center, Arnold AFS, Tennessee (Apr. 1986).
12. R.L. Meakin, AIAA-91-1586-CP (June 1991).
13. S.J. Parks, P.G. Buning, J.L. Steger, and W.M. Chan, AIAA-91-1587-CP (June 1991).
14. T.H. Pulliam and D.S. Chaussee, J C P **39**, 347 (1981).
15. T.H. Pulliam, "Efficient Solution Methods for the Navier-Stokes Equations," von Karman Institute for Fluid Dynamics Lecture Series: Numerical Techniques for Viscous Flow Computation in Turbomachinery Bladings, Jan. 20-24, 1986, Brussels, BELGIUM.
16. B.S. Baldwin and H. Lomax, AIAA-78-257 (Jan. 1978).
17. D. Degani and L.B. Schiff, AIAA-83-0034 (Jan. 1983).

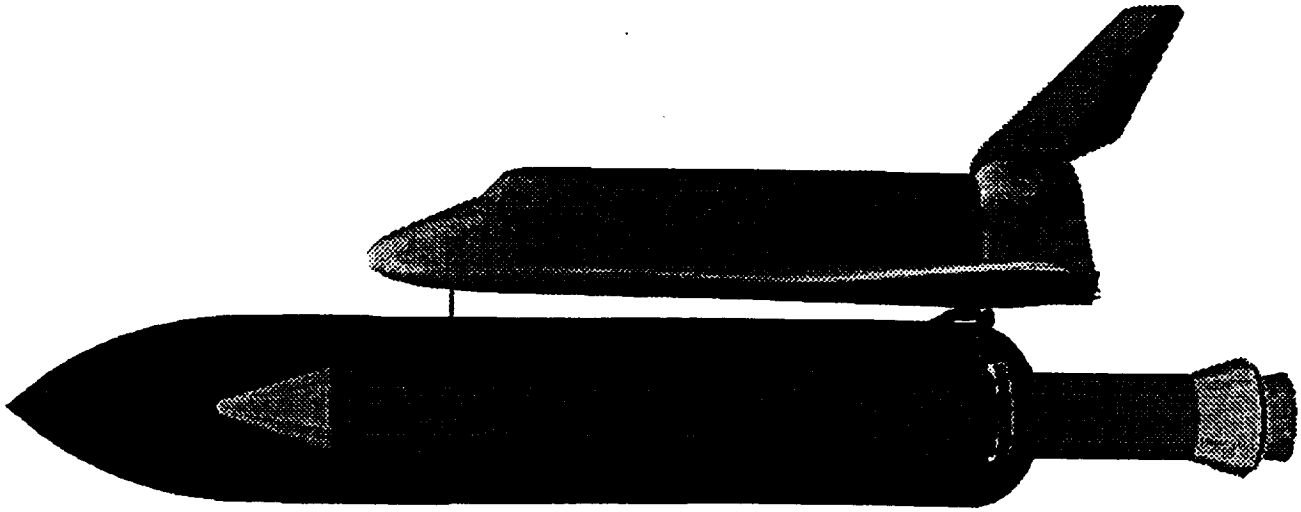


Figure 1. Computational model of the Space Shuttle launch vehicle geometry.

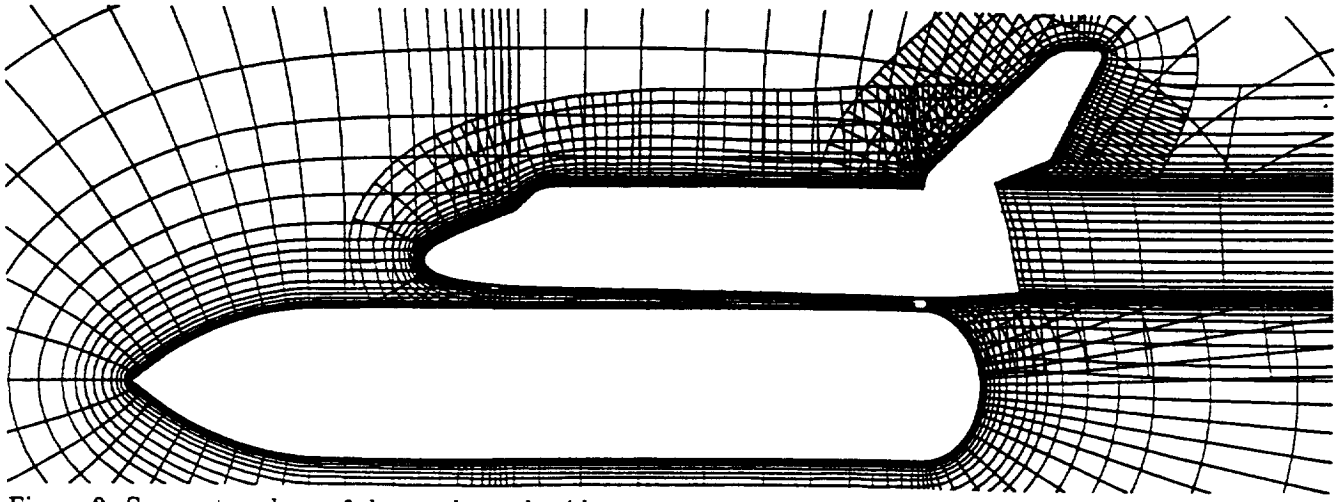
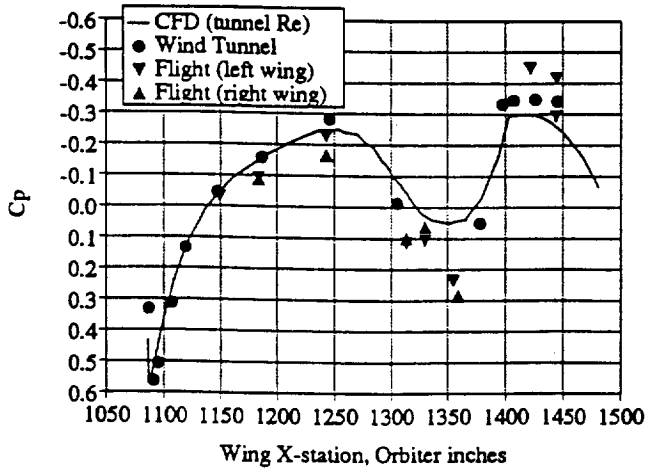
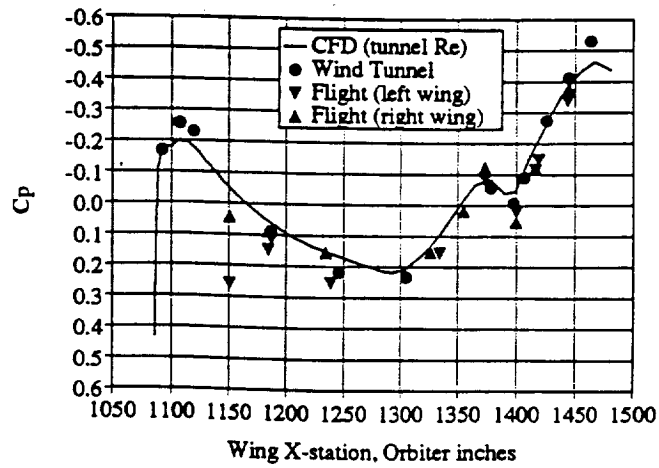


Figure 2. Symmetry plane of the overlapped grid system.



(a) upper surface



(b) lower surface

Figure 3. Orbiter wing pressure coefficient at span location $y = 250$ inches (directly above SRB centerline).

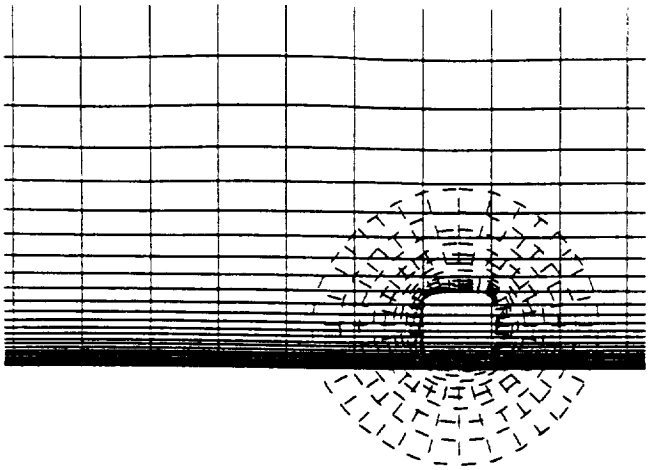


Figure 4. Overlapped grids for a flat plate with an obstruction.

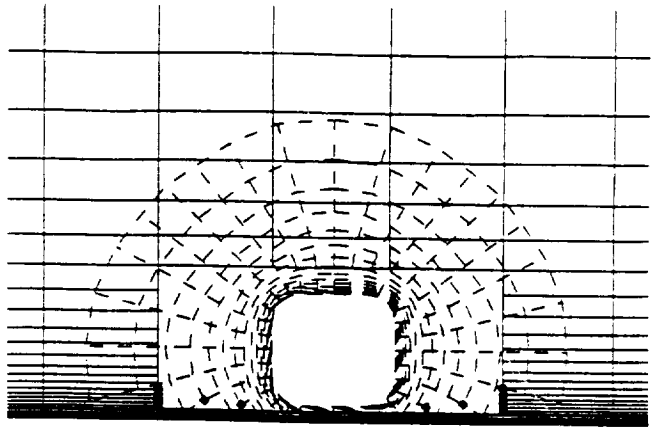
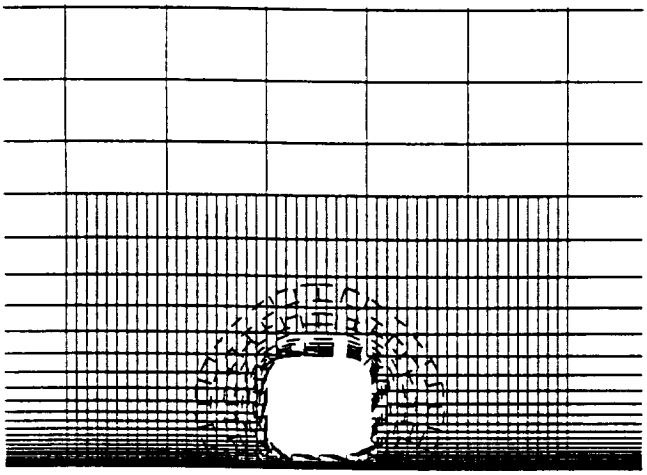
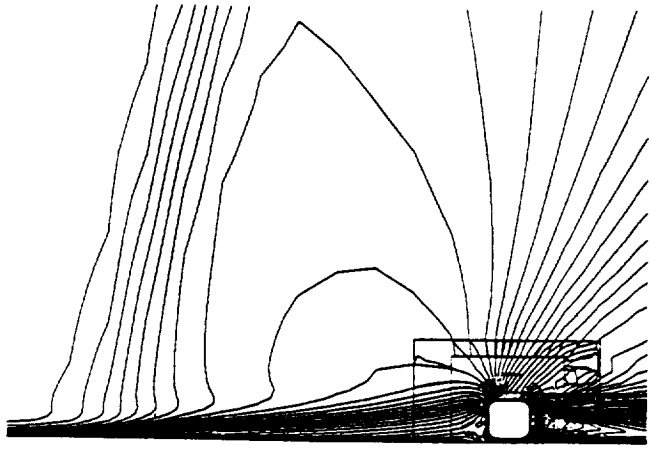


Figure 5. Orphan points left when joining overlapped grids.

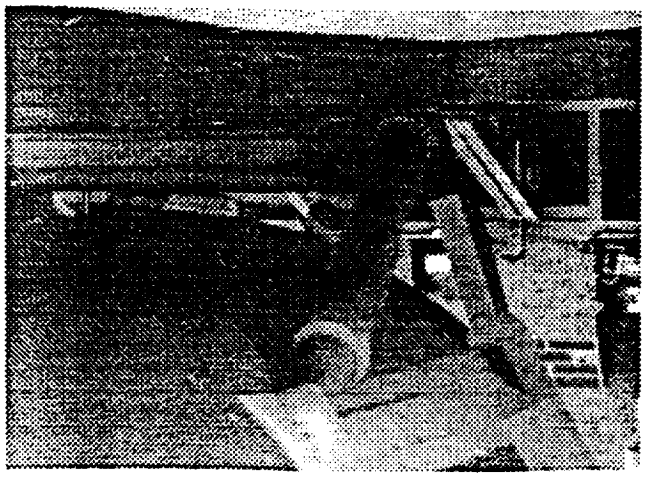


(a) grids

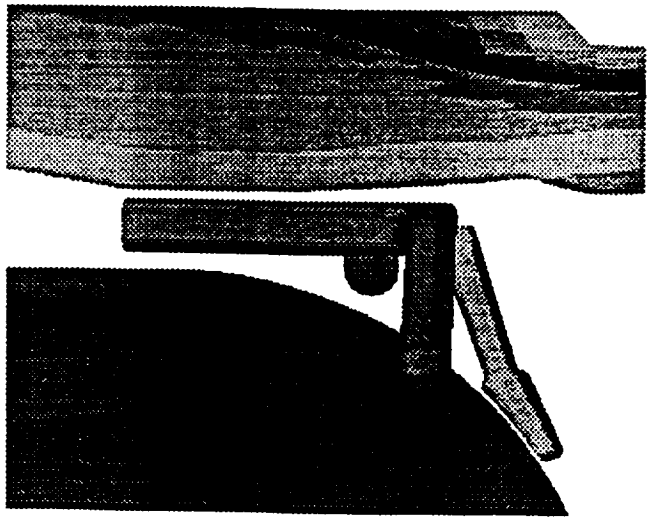


(b) sample solution at Mach 1.25

Figure 6. Local grid refinement to overcome grid joining problems.

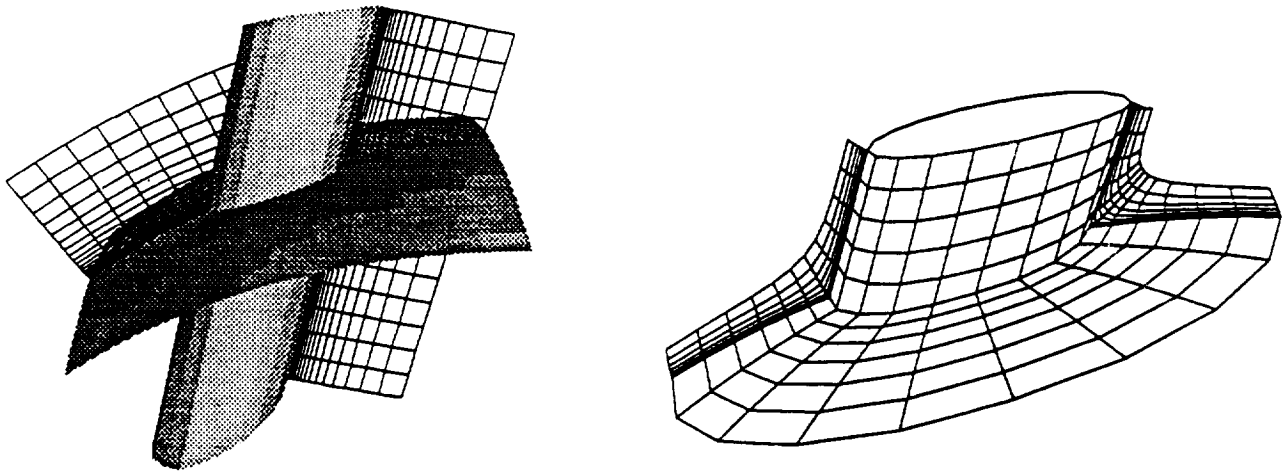


(a) flight hardware



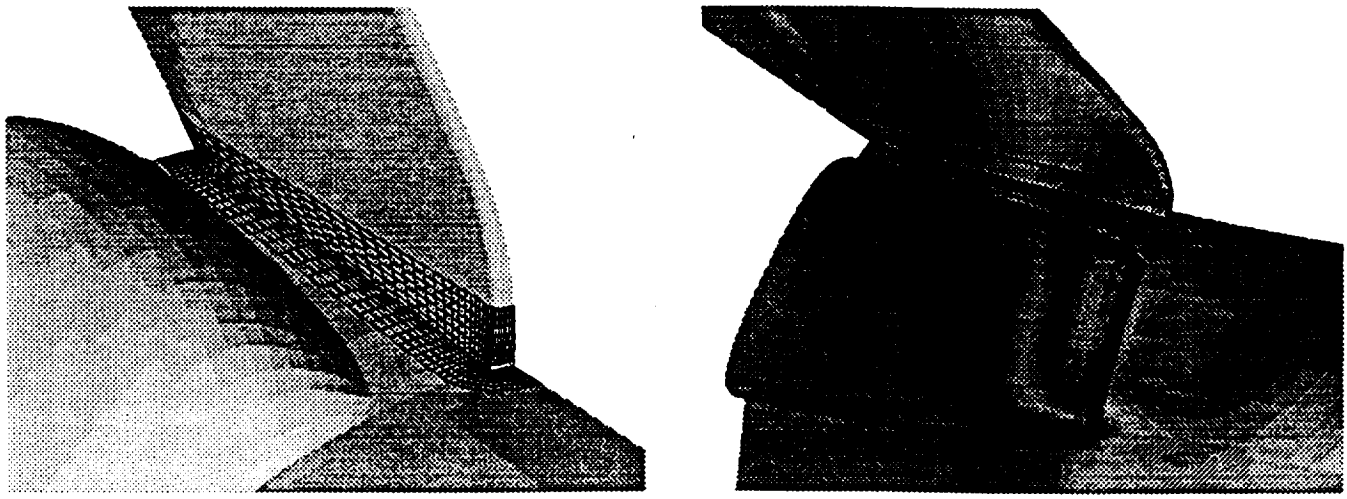
(b) computational model

Figure 7. Orbiter/ET aft attach hardware.



(a) component grids (b) collar grid about intersection region

Figure 8. Simplified model of the External Tank and LH2 feedline.



(a) collar grid (b) surface pressure coefficient

Figure 9. Orbiter vertical tail addition.

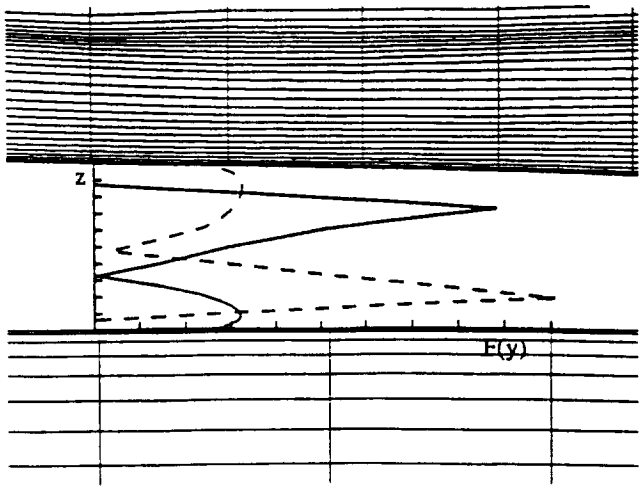


Figure 10. Profile of $F(y)$ vs. z between the Orbiter (top) and ET, from Orbiter grid (- -), and from ET grid (—).