AIAA JOURNAL
Vol. 34, No. 9, September 1996

Three-Dimensional Hyperbolic Grid Generation with Inherent Dissipation and Laplacian Smoothing

C. H. Tai,* D. C. Chiang,† and Y. P. Su†
Chung Cheng Institute of Technology, Taiwan 33509, Republic of China

Based on upwind differencing of the governing equations, inherent dissipation terms are derived for three-dimensional hyperbolic grid generation. These terms, taking only a negligible amount of mathematic operations, can effectively eliminate grid oscillation that is often encountered in hyperbolic grid generation. In addition to these dissipation terms, a Laplacian-type smoothing is incorporated to increase smoothness. Both implicit and explicit schemes for three-dimensional hyperbolic grid generation are investigated, and a comparison of these two schemes is made. Several different geometries are used to demonstrate the robustness of the new methods. Finally, a wing–body configuration, with severe concave and convex corners, is used to evaluate the versatility of the present methods.

I. Introduction

FOR generating structured grids, one very popular approach is solving a set of partial differential equations. Three types of partial differential equations have been incorporated successfully, namely, elliptic, parabolic, and hyperbolic. Each type has advantages for different applications. Among them, the hyperbolic equations have the merits of good orthogonality, ease of control of clustering, and efficiency in computation time. But traditionally this method is considered as less robust and only suitable for external flow computations.

Recently, more and more researchers use the Reynolds-averaged Navier–Stokes equations for flow computations. For this type of computation, the orthogonality and proper control of grid clustering are essential requirements for grid generation. Therefore, the merits of hyperbolic grid generation are more significant. For other types of computations the users could also benefit from using hyperbolic grid generation; e.g., the orthogonality at boundary can highly simplify the specification of boundary conditions.

Many researchers have worked on this subject. Instead of an exhaustive reference, only those that have direct relation to this investigation are referred to here. The hyperbolic grid generation can be traced back to McNally[1] and Graves.[2] But Steger and Chaussee[3] made the first systematic analysis of this method for two-dimensional applications configurations. Then it was extended to three dimensions by Steger and Rizk.[4] Further enhancements were made by Chan and Steger.[5]

In hyperbolic grid generation, any discontinuity at the initial boundary will propagate into the grid field by the marching process; eventually oscillations or even overlapping of grid lines may result. This shortcoming can be overcome by adding artificial dissipation term(s) to the discretized governing equations.[3–5] Since the problem is caused by the geometric discontinuity, it is obvious that a constant coefficient for the dissipation term is insufficient. In Ref. 5, the spacing and intersection angle of the neighboring points were used to adjust the dissipation coefficients. Grids for very complex configurations were obtained, but the procedure is rather complex and time consuming.

In the work of Tai and Yin[6] and Tai et al.[7] another approach was proposed. By upwind differencing of the governing equations, an inherent adaptive dissipation term was obtained and successfully used in generating grids for two-dimensional configurations. Recently, Jeng et al.[8] adopted this concept and developed a predictor- corrector scheme with additional smoothness.

Received March 31, 1995; revision received Oct. 20, 1995; accepted for publication Nov. 10, 1995. Copyright © 1996 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Associate Professor, Department of Mechanical Engineering, Ta-Hsi, Taoyuan.

†Ph.D. Student, Department of System Engineering, Ta-Hsi, Taoyuan.

In this study, the use of inherent dissipation terms is extended to three dimensions. Implicit and explicit schemes are developed. A number of smoothing techniques are imposed for better smoothness and robustness. The explicit scheme has been found to be less robust.[7] However, while considering the future deployment of a massive parallel processing computer, it could have much higher efficiency than an implicit scheme,[9] and so it is reexamined. The application to internal flow problems is not included in this paper; interested readers may refer to Ref. 8.

In Sec. II, the governing equations are briefly reviewed. The derivation of inherent adaptive dissipation terms and the discretization for both implicit and explicit schemes shall also be introduced. In Sec. III, specifications of cell volume are discussed, and in Sec. IV, boundary conditions. Techniques for smoothing are described in Sec. V, with the application to different configurations. Results of the application to a wing–body configuration are shown in Sec. VI. Finally, conclusions are made in Sec. VII.

II. Mathematic Model and Numerical Procedure

The governing equations, derived from orthogonality relations and a finite volume constraint, were proposed by Steger and Rizk.[4] The equations are repeated here for completeness:

$$x_\xi x_\zeta + y_\xi y_\zeta + z_\xi z_\zeta = 0 \tag{1}$$

$$x_\eta x_\zeta + y_\eta y_\zeta + z_\eta z_\zeta = 0 \tag{2}$$

$$x_\xi y_\eta z_\zeta + x_\zeta y_\xi z_\eta + x_\eta y_\zeta z_\xi - x_\xi y_\zeta z_\eta - x_\eta y_\xi z_\zeta - x_\zeta y_\eta z_\xi = \Delta V \tag{3}$$

$$\Delta V = J^{-1} = \left| \frac{\partial(x, y, z)}{\partial(\xi, \eta, \zeta)} \right| \tag{4}$$

where x, y, and z are the Cartesian coordinates, and ξ, η, and ζ are the transformed generalized coordinates. In this paper ζ is chosen as the marching direction; $\zeta = 0$ coincides with body surface, where the distributions of $\xi = \text{const}$ and $\eta = \text{const}$ are user specified.

Equations (1–3) can be cast into a compact form as

$$\boldsymbol{A}\boldsymbol{W}_\xi + \boldsymbol{B}\boldsymbol{W}_\eta + \boldsymbol{C}\boldsymbol{W}_\zeta = \boldsymbol{f} \tag{5}$$

where

$$\boldsymbol{A} = \begin{bmatrix} x_\zeta^0 & y_\zeta^0 & z_\zeta^0 \\ 0 & 0 & 0 \\ \left(y_\eta^0 z_\zeta^0 - y_\zeta^0 z_\eta^0\right) & \left(x_\zeta^0 z_\eta^0 - x_\eta^0 z_\zeta^0\right) & \left(x_\eta^0 y_\zeta^0 - x_\zeta^0 y_\eta^0\right) \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0 & 0 \\ x_\zeta^0 & y_\zeta^0 & z_\zeta^0 \\ \left(y_\zeta^0 z_\xi^0 - y_\xi^0 z_\zeta^0\right) & \left(x_\xi^0 z_\zeta^0 - x_\zeta^0 z_\xi^0\right) & \left(x_\zeta^0 y_\xi^0 - x_\xi^0 y_\zeta^0\right) \end{bmatrix}$$

$$C = \begin{bmatrix} x_\xi^0 & y_\xi^0 & z_\xi^0 \\ x_\eta^0 & y_\eta^0 & z_\eta^0 \\ \left(y_\xi^0 z_\eta^0 - y_\eta^0 z_\xi^0\right) & \left(x_\eta^0 z_\xi^0 - x_\xi^0 z_\eta^0\right) & \left(x_\xi^0 y_\eta^0 - x_\eta^0 y_\xi^0\right) \end{bmatrix}$$

$$W = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \qquad f = \begin{bmatrix} 0 \\ 0 \\ \Delta V + 2\Delta V^0 \end{bmatrix}$$

where the superscript 0 denotes the known state. For finite cell volume C^{-1} exists, and so Eq. (5) can be written as

$$W_\zeta + DW_\xi + EW_\eta = C^{-1}f \tag{6}$$

where $D \equiv C^{-1}A$ and $E \equiv C^{-1}B$ are symmetric matrices.

To discretize Eq. (6), we first rewrite it in conservation form:

$$W_\zeta + F(W)_\xi + G(W)_\eta = C^{-1}f \tag{7}$$

Let $\Delta\xi = \Delta\eta = \Delta\zeta = 1$ and $\xi = i-1, \eta = j-1$, and $\zeta = k-1$. By the Roe upwind scheme,[10] the fluxes are

$$F_\xi = \tfrac{1}{2}\left[(DW_{i+1} + DW_i) - |D|_{i+\frac{1}{2}}(W_{i+1} - W_i)\right]$$
$$-\tfrac{1}{2}\left[(DW_{i-1} + DW_i) - |D|_{i-\frac{1}{2}}(W_i - W_{i-1})\right] \tag{8a}$$

$$G_\eta = \tfrac{1}{2}\left[(EW_{j+1} + EW_j) - |E|_{j+\frac{1}{2}}(W_{j+1} - W_j)\right]$$
$$-\tfrac{1}{2}\left[EW_{j-1} + EW_j) - |E|_{j-\frac{1}{2}}(W_j - W_{j-1})\right] \tag{8b}$$

where only those indices that change are indicated, and so W_{i+1} represents $W_{i+1,j,k}$, etc. More details of derivation were given in Ref. 7. In Eqs. (8a) and (8b) $|D|$ and $|E|$ are two diagonalized matrices composed of positive eigenvalues of D and E. For a symmetric matrix, all eigenvalues are real. To find the eigenvalues we denote the matrix as

$$D = \begin{bmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{bmatrix}; \qquad E = \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix}$$

for simplicity.

Thus, three eigenvalues of D are found to be 0, $\pm\lambda$, where

$$\lambda = \sqrt{-(d_{11}d_{22} - d_{12}d_{21} + d_{11}d_{33} - d_{13}d_{31} + d_{22}d_{33} - d_{23}d_{32})} \tag{9a}$$

Similarly, the eigenvalues for E are 0, $\pm\beta$, where

$$\beta = \sqrt{-(e_{11}e_{22} - e_{12}e_{21} + e_{11}e_{33} - e_{13}e_{31} + e_{22}e_{33} - e_{23}d_{32})} \tag{9b}$$

Then $|D|$ and $|E|$ are expressed as

$$|D| = R\begin{bmatrix} \lambda & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \lambda \end{bmatrix}R^{-1} = \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix}$$
$$-R\begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & 0 \end{bmatrix}R^{-1} \tag{10a}$$

$$|E| = P\begin{bmatrix} \beta & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \beta \end{bmatrix}P^{-1} = \begin{bmatrix} \beta & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & \beta \end{bmatrix}$$
$$-P\begin{bmatrix} 0 & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 0 \end{bmatrix}P^{-1} \tag{10b}$$

where R and P are matrices composed of eigenvecters corresponding to λ, 0, λ and β, 0, and β, respectively. Unfortunately, the determination of R, P, and their inverse is nontrivial, and many operations are needed. By numerical experiments, we find no significant difference in dropping the last terms in Eqs. (10a) and (10b). Therefore, $|D| = \lambda I$ and $|E| = \beta I$ are used in this paper, where I is the identity matrix. As noted in Ref. 7, there are three ways to calculate the interface value of λ and β; among them, $\lambda_{i\pm1/2} = \sqrt{(\lambda_i\lambda_{i\pm1})}$ and $\beta_{j\pm1/2} = \sqrt{(\beta_j\beta_{j\pm1})}$ are found to be most robust. Therefore, they are used for all cases shown in this paper.

Substituting Eq. (8) and the expressions for $|D|$ and $|E|$ into Eq. (7), the semidiscretized equation is

$$W_\zeta + \tfrac{1}{2}\left(D - \lambda_{i+\frac{1}{2}}I\right)W_{i+1} + \tfrac{1}{2}\left(\lambda_{i+\frac{1}{2}} + \lambda_{i-\frac{1}{2}}\right)W_i$$
$$-\tfrac{1}{2}\left(D - \lambda_{i-\frac{1}{2}}\right)W_{i-1} + \tfrac{1}{2}\left(E - \beta_{j+\frac{1}{2}}I\right)W_{j+1} + \tfrac{1}{2}$$
$$\times\left(\beta_{j+\frac{1}{2}} + \beta_{j-\frac{1}{2}}\right)W_j - \tfrac{1}{2}\left(E - \beta_{j-\frac{1}{2}}\right)W_{j-1} = C^{-1}f \tag{11}$$

By approximate factorization, the implicit formulation is

$$\left[I + E\delta_\eta - \tfrac{1}{2}\varepsilon_\eta(\Delta\nabla)_\eta\right]\left[I + D\delta_\xi - \tfrac{1}{2}\varepsilon_\xi(\Delta\nabla)_\xi\right]$$
$$\times(W_{k+1} - W_k) = C^{-1}g_{k+1} \tag{12}$$

where $g = [0, 0, \Delta V]^T$; δ_η and δ_ξ are central difference operators with respect to η and ξ. For an arbitrary variable ϕ, they are defined as

$$\delta_\xi\phi = \frac{\phi_{i+1} - \phi_{i-1}}{2}, \qquad \delta_\eta\phi = \frac{\phi_{j+1} - \phi_{j-1}}{2} \tag{13}$$

Note that Eq. (12) is just like the one incorporated in Ref. 5, except that it contains only the implicit dissipation terms. For an arbitrary variable ϕ, the dissipation terms are

$$\varepsilon_\eta(\Delta\nabla)_\eta\phi_j = \beta_{j+\frac{1}{2}}\phi_{j+1}$$
$$-\left(\beta_{j+\frac{1}{2}} + \beta_{j-\frac{1}{2}}\right)\phi_j + \beta_{j-\frac{1}{2}}\phi_{j-1} \tag{14a}$$

$$\varepsilon_\xi(\Delta\nabla)_\xi\phi_i = \lambda_{i+\frac{1}{2}}\phi_{i+1}$$
$$-\left(\lambda_{i+\frac{1}{2}} + \lambda_{i-\frac{1}{2}}\right)\phi_i + \lambda_{i-\frac{1}{2}}\phi_{i-1} \tag{14b}$$

Since these terms resulted from the discretization procedure rather than being artificially added, they are called inherent dissipation terms.

For the explicit scheme, Eq. (7) can be expressed as

$$\begin{aligned} W_\zeta &= C^{-1}f - F_\xi - G_\eta \equiv \mathrm{Res}(W_k) \\ W_{k+1} &= W_k + \mathrm{Res}(W_k) \end{aligned} \tag{15}$$

with F_ξ and G_η defined in Eqs. (8a) and (8b); Res stands for residue.

III. Specification of Cell Volume

Through the specification of the cell volume distribution, the user may control the clustering or stretching of the grid. One simple and popular way to specify the cell volume is

$$\Delta V_{i,j,k} = \Delta s_{i,j,k}\Delta A_{i,j,k} \tag{16}$$

where Δs is a user-specified marching distance and ΔA is the calculated element surface area. For different purposes, different functions can be used to specify Δs. In this paper, only a simple exponential function is used to control the stretching in the ζ direction. The elemental surface area should be the area surrounding the nodal point, as illustrated by the shaded area in Fig. 1a. However, exact calculation of this area is too complicated and not worth the effort. Actually, it is the area that is bounded by four adjacent points that is calculated, and a quarter of this area contributes to each point. Obviously, this procedure cannot be applied to the boundary points. For calculation of the areas at the boundaries, pseudopoints are added outside the boundary (which will be discussed in the next section) to make the calculation algorithm no different than that for the internal points. At an axis point, as shown in Fig. 1b, the cell is a triangle formed by the adjacent points instead of a rectangle. Thus, three-eights of the area contributes to the points next to the axis point.

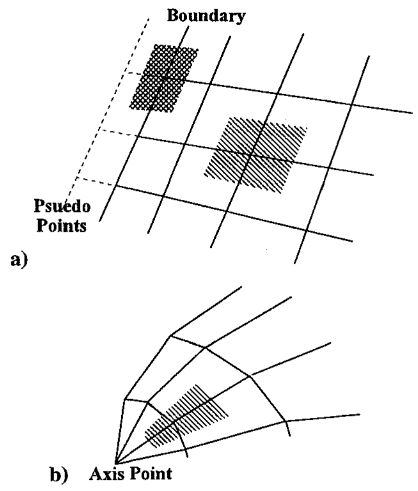


Fig. 1 Cell area for a) normal point and b) axis point.

If the initial surface grid is not equally spaced or an abrupt change of geometry is encountered, the cell volume calculated by the preceding method will not be smoothly distributed. Weighting of the cell volumes for a smoother distribution is suggested in Refs. 4, 5, and 8. Weighting to the average value of neighboring cells is proposed in Ref. 5, whereas the average of the whole level of cell volumes is used in Ref. 8. In Ref. 4, a similar but simple reference body, for which the grid can be generated analytically, is used for weighting of the cell volume. In this study, the average value of the whole level is used to adjust the cell volume. The weighting is defined as

$$\Delta\tilde{V}_{i,j,k} = \alpha\Delta V_{i,j,k} + \tfrac{1}{2}(1-\alpha)(\Delta V_{i,j,k} + \Delta\bar{V})$$

$$\Delta\bar{V} = \frac{1}{(im \times jm)}\Sigma_{i,j}\Delta V_{i,j,k}; \qquad i = 1, im, \quad j = 1, jm \tag{17}$$

$$\alpha = (1-\alpha_0)^{(k-2)}$$

where $\Delta\tilde{V}$ is the weighted cell volume, ΔV is the area calculated by Eq. (16), and α_0 is a user-specified constant.

IV. Boundary Conditions

Four types of boundary conditions are incorporated for the cases shown in this paper. For generality, pseudoboundary lines are used for all types of boundary conditions. The grids on the initial surface are defined in $i = 1$ to im and $j = 1$ to jm, where im and jm are the maximum indices for i and j. A pseudopoint at each end is added to increase the range of i and j so that $i = 0$ to $im + 1$ and $j = 0$ to $jm+1$. These are used to march the grid to the new level in the ζ direction for $i = 1$ to im and $j = 1$ to jm. Then new pseudopoints are calculated for next marching step. Four different ways to calculate these pseudopoints are described next and illustrated in Fig. 2.

Periodic

For a closed surface, use of a periodic boundary condition in one direction would be the simplest way. If the periodic condition is applied in the η direction, $W_{j=jm+1}$ is set equal to $W_{j=1}$ and $W_{j=0} = W_{j=jm}$; also $\beta_{jm+1} = \beta_1$ and $\beta_0 = \beta_{jm}$. A periodic block tridiagonal solver is used for the implicit scheme.

Constant Cartesian Plane

If the grid is restricted to x = const in ξ direction at $i = 1$, x_0 is extrapolated as $x_0 = 2x_1 - x_2$, y_0, and z_0 are set equal to y_1 and z_1; λ_0 is set equal to λ_1. For the implicit scheme, the implementation is the same as that proposed in Ref. 5. For the explicit scheme, no further treatment is needed. The same method can be applied to both ends of ξ and η directions for x = const or y = const.

Symmetry Plane

Reflection relation is used for a plane of symmetry about $x = 0$, $y = 0$, or $z = 0$ plane. For example, in the η direction, if $j = 1$ is the symmetry plane at $z = 0$, then $z_0 = -z_2$, $x_0 = x_2$, and $y_0 = y_2$; β_0 is set equal to β_2.

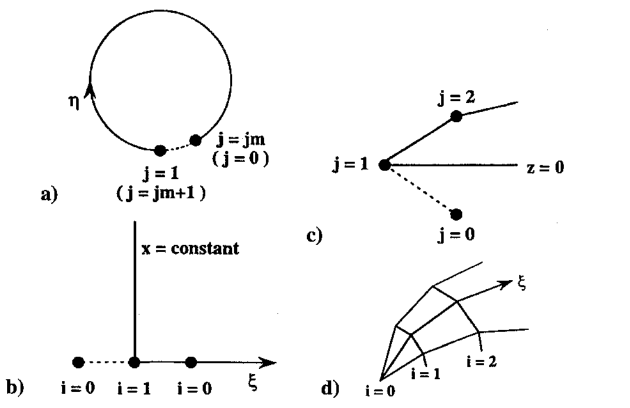


Fig. 2 Illustration of psuedopoints for a) periodic, b) constant plane, c) symmetry, and d) axis boundary conditions.

Axis

The axis condition may be the most difficult boundary condition for hyperbolic grid generation. At the axis point, a whole line coincides into one point; this makes the determinant of $\boldsymbol{C}$ equal zero, and the governing equations are no longer well posed. For this reason, the axis points cannot be included in a normal calculation procedure and must be treated as pseudoboundary points. At the initial surface, these points must be userspecified. For other level of grids, these points are determined by a mixing type extrapolation, as proposed in Ref. 5. The extrapolation can be represented as

$$\phi_0 = \phi_1 + \alpha(\phi_1 - \phi_2) \tag{18}$$

where the subscript 0 represents axis point and α is an extrapolation factor. In general, a value of 1 is used; however, it can be adjusted between 0 to 1. This is shown in the next section. For the implicit scheme, $\Delta\boldsymbol{W}$ is extrapolated. For the explicit scheme, Res is extrapolated. The new axis points are determined from the extrapolated values of $\Delta\boldsymbol{W}$ or Res. These points may not coincide on the axis, especially for a nonaxisymmetric configuration. The marching distances of these points are averaged and then are used to relocate these points on the axis.

V. Smoothing

In Ref. 5, spatially variable dissipation coefficients were introduced and very good smoothing properties were achieved. Although effective, they are rather complicated. These coefficients are composed of matrices norms, distribution functions, angle functions, scaling function, and a user-specified constant. On the contrary, the dissipation coefficients proposed in this paper, as defined in Eqs. (9a) and (9b), are much simpler to implement and require little computational effort.

For a simple configuration without sharp convex or concave corners, the inherent dissipation terms are sufficient for the implicit scheme to generate a grid with satisfaction. But for the explicit scheme, since the spread of information is slower, a Laplacian-type residue smoothing must be incorporated and implemented in the i and j directions separately. For an arbitrary variable ϕ the smoothing operator can be written as

$$\tilde{\phi} = [1 + \alpha(\Delta\nabla)]\phi \tag{19}$$

where ~ identifies the new value after smoothing; $(\Delta\nabla)$ is the Laplace operator [e.g., $(\Delta\nabla)_\xi\phi = \phi_{i+1} - 2\phi_i + \phi_{i-1}$]; and α is a user-specified coefficient. This coefficient should not be larger than 0.25; throughout this paper a value of 0.2 is used. When ϕ is replaced by Res, defined in Eq. (15), it is called residue smoothing. For all cases shown later, residue smoothing is incorporated into the explicit scheme. The smoothing operator can also be cast into an implicit form and better robustness can be achieved, but this extension is not considered in this study.

An ellipsoid with aspect ratio of (16:1:0.5; x:y:z) is used as an example. For both the explicit and implicit schemes, similar results (as shown in Fig. 3) are obtained without any further treatment. In

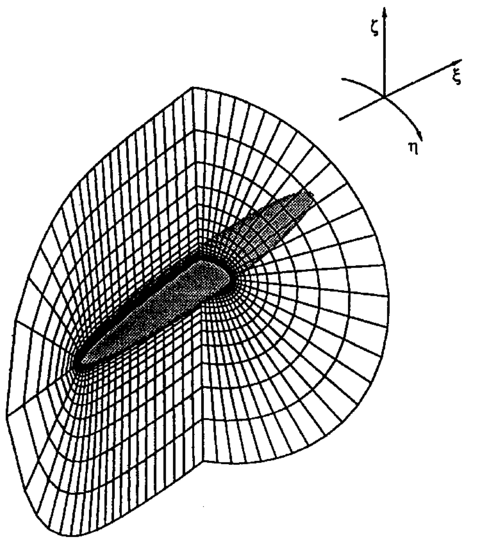


Fig. 3 Grid around an ellipsoid.

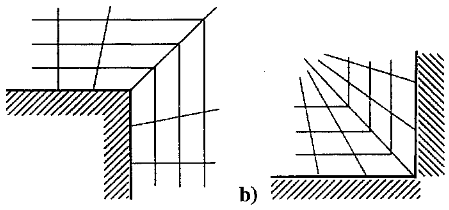


Fig. 4 Illustration of grid near the a) convex corner and b) concave corner.

this case, an axis boundary condition is imposed on both ends of the axial direction and periodic boundary condition for the circumferential direction. Although the configuration is nonaxisymmetric, the axis boundary condition described in the preceding section caused no problems.

For more complicated configurations, further treatments are necessary. Since the orthogonal constraint is imposed in the governing equations, the grid lines at a convex corner will point outward as shown in Fig. 4a. As the grid marches further, the grid lines near the convex corner separate wider apart; and consequently oscillation occurs, whereas the grid lines at a concave corner will cluster into the corner as shown in Fig. 4b. Depending on the degree of the corner and the marching step size, the neighboring grid lines may cross over to each other after several marching steps. One simple solution is to apply an Laplacian operator to the coordinates vector W, after each new level of grid is obtained. Although Eq. (19) can be used for this purpose, a small modification is introduced for better smoothness. Thus, the discrete formulation can be represented as

$$\tilde{W}_i = (1-\alpha)W_i + \alpha\left(r^+ W_{i+1} + r^- W_{i-1}\right)\big/(r^+ + r^-) \qquad (20)$$

where r^+ is the distance between W_i and W_{i+1}, r^- is the distance between W_i and W_{i-1}, and α is again a user-specified coefficient and should be within 0–0.5. In the following cases 0.4 is used as the upper bound. Equation (20) is also applied in the j direction. The idea of Eq. (20) is to have the grid points more uniformly distributed to prevent the neighboring points getting too far away or too close together.

Just like the dissipation coefficients, the smoothing coefficient is also geometry dependent. At the convex corner, the coefficient should become larger as the grid marches further away from the body surface. But, on the contrary, a larger value is needed near the body surface for a concave corner, and so the value of α must be evaluated according to the variation of the geometries.

First consider the case with only the convex corner. The smoothing coefficient should be started from 0 at the body surface and then increased smoothly as the grid marches away. Once the upper bound value is reached, the upper bound value should be used thereafter. For the external flow problems, a nondecreasing function can be used to control the coefficient. In this study, a linear function is used:

$$\alpha = \alpha_0(k-2)/k_s \qquad (21)$$

where α_0 is the upper bound of α; a value of 0.4 is used for the following cases; and k_s can be specified, in general, as the maximum value of the k index. However, for some applications, if only a few levels of grid are needed, k_s should be otherwise specified (e.g., set for $45 \geq k_s \geq 25$). The effect of this smoothing technique can be observed in Fig. 5. Figure 5a is obtained without this smoothing. With the smoothing technique incorporated, Fig. 5b shows that the distribution of the grid cells near the trailing edge and the nose are much smoother than that of the ones in Fig. 5a.

Also notice in Fig. 5 that the same axis boundary condition is imposed at both ends of the projectile. Though it is a very sharp convex corner at the nose while a flat plate at the bottom, the same axis condition is applied without any adjustment.

Figures 5a and 5b are obtained by the implicit scheme, and Fig. 6 is calculated by the explicit scheme with the same smoothing technique. One may notice that sharp convex corners are formed at the axis points. If this is not desired, it can be simply cured by changing the extrapolation factor defined in Eq. (18). Figure 7 is obtained

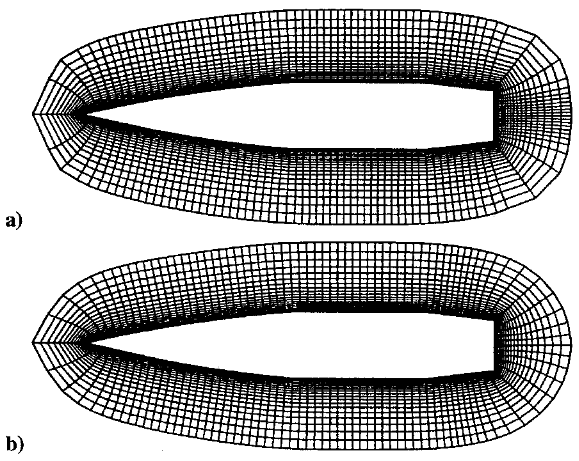


Fig. 5 Grid around a projectile for a) without smoothing and b) with smoothing, by implicit scheme.

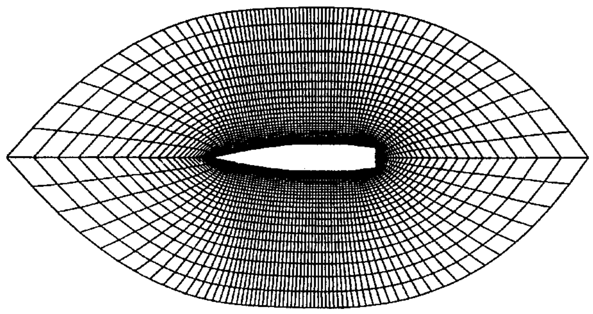


Fig. 6 Grid around a projectile by explicit scheme with extrapolation factor 1.

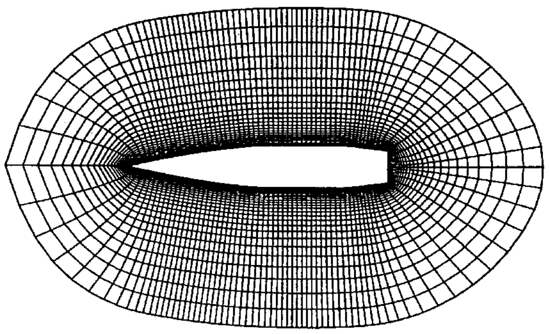


Fig. 7 Grid around a projectile by explicit scheme with extrapolation factor 0.25.

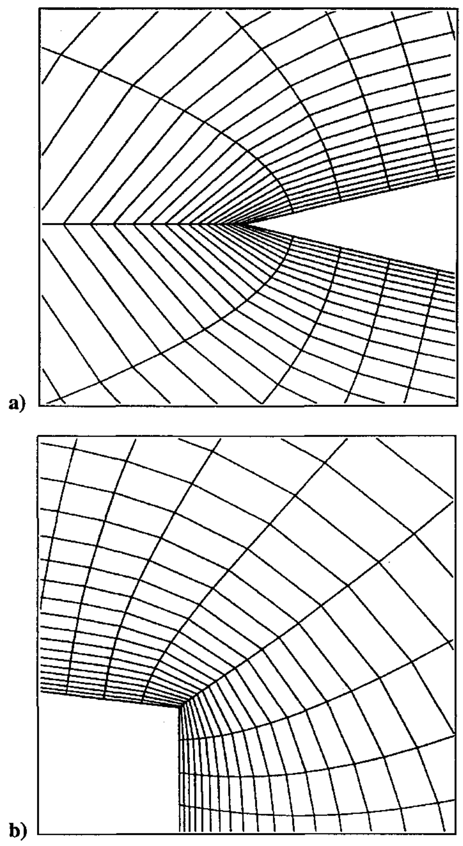


Fig. 8 Enlarged views of projectile grid in a) nose and b) tail regions.

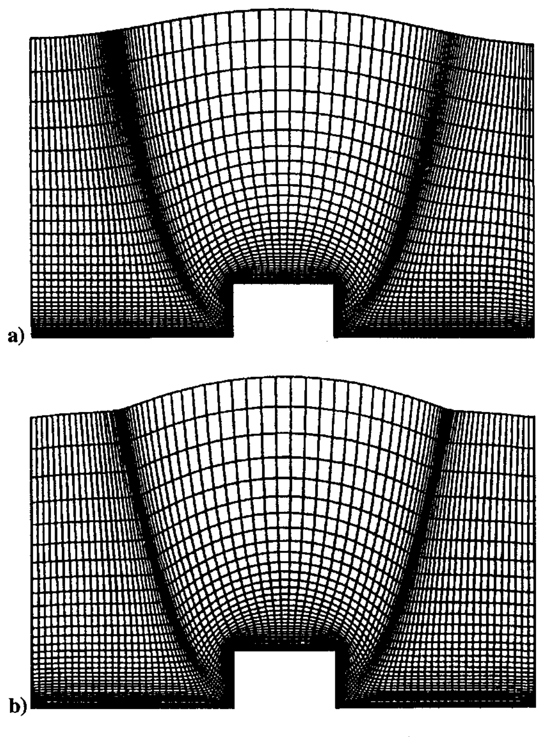


Fig. 9 Grid around a square obstacle by a) implicit scheme and b) explicit scheme.

with a factor of 0.25. Also by the explicit scheme, Figs. 8a and 8b show the blowup views of two sharp corners; smoothness and orthogonality are well maintained near these corners.

Then consider the case of the concave corner. Now the problem is that the neighboring points at the corner will come closer to each other. The ratio of the grid spacing $ar = (r_{k+1}^{+} + r_{k+1}^{-})/(r_k^{+} + r_k^{-})$ is used as an indicator. When ar is smaller than 1, the grid is regarded as clustering, and a value of $(1 - ar)$ is added to the smoothing coefficient calculated from Eq. (21), but the sum should be restricted to the same upper bound.

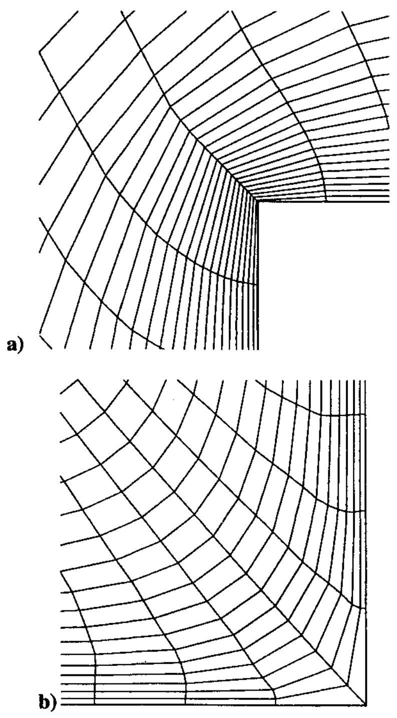


Fig. 10 Enlarged views of Fig. 9 for a) convex corner and b) concave corner.

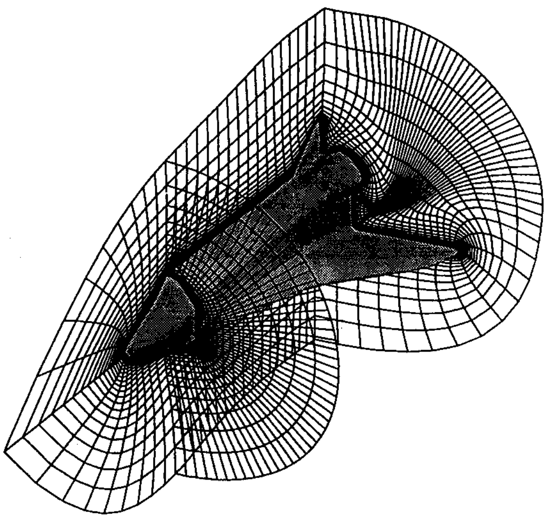


Fig. 11 Grid around a wing–body configuration.

The smoothing procedure can be applied iteratively to the same level of grid. Numerical experiment shows that more iterations with a smaller value of α_0 will be better than fewer iterations with larger value of α_0, and the iteration times can also be increased with respect to the k index.

When severe convex and concave corners are present together, as shown in the next case, the smoothing coefficient needs to be further tuned. For a sharp convex corner, the smoothing coefficient is multiplied by $(k-2)/k_s$. The external angles of every point on the initial surface are calculated by the method described in Ref. 5. The points with angles greater or equal to 270 deg are marked as convex points. For a new level of grid, only the angles of these marked points are recalculated; if the new angle is less than 240 deg, the point is reset to normal. Because of the smoothing property, the convex corner will be flattened as the grid marches outward. Therefore, after several marching steps, there should be no convex point. The increase of CPU time required by this procedure is rather limited because only a few points need be checked.

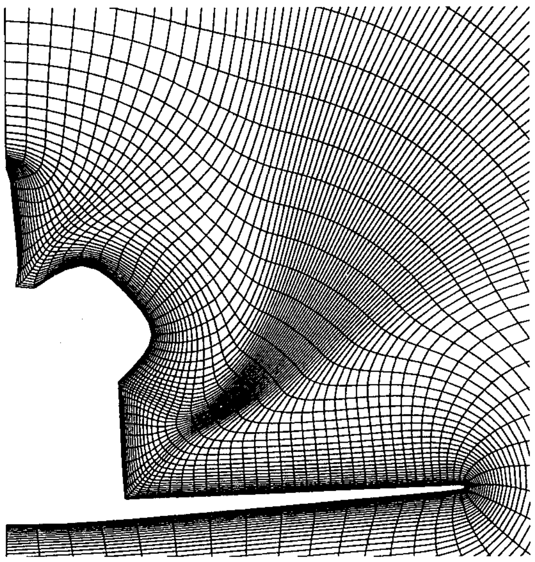
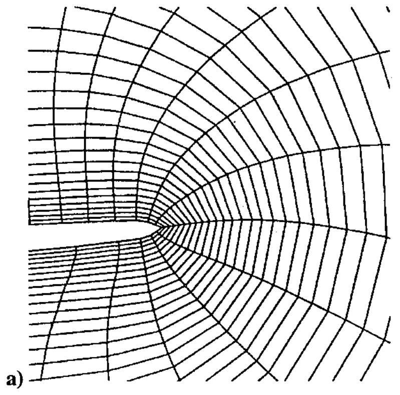


Fig. 12 Grid at rear section of wing–body configuration.



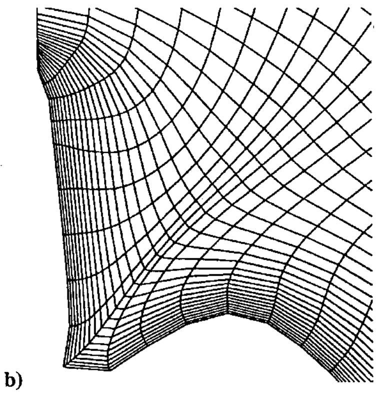


Fig. 13 Enlarged views of Fig. 12 for the a) wing tip and b) root of vertical tail regions.

Figure 9 illustrates the grids for a 90-deg step obtained by both the explicit and implicit schemes. Although the smoothing coefficients are kept the same for both schemes, the grid obtained by the implicit scheme is smoother than the one obtained by the explicit scheme. Constant Cartesian plane boundary conditions are imposed for both the ξ and η directions in this case. As shown in Fig. 9, the straight plane is maintained at both ends; this makes combining with other grid systems much easier. Figures 10a and 10b show the detail near convex and concave corners.

VI. Results and Application

A wing–body configuration with numerous concave and convex corners is used to test the robustness of the present method. The result shown in Fig. 11 is obtained by the implicit scheme. An axis boundary condition is applied at the nose, and a constant plane condition is imposed at the end of the body. A symmetry boundary condition is used at the symmetry plane. Shown in Fig. 12 is a plane at the rear section; blowup views for the wing tip and the root of the vertical tail regions are shown in Figs. 13a and 13b. Special treatment for these corners is not needed. All techniques described are applied in a global way. There is no need to change the procedure by detecting the local grid property or by prescription. This can highly reduce the efforts of coding and required user input.

With the success of the implicit scheme, unfortunately, the explicit scheme failed in generating a grid for such a configuration. The explicit scheme is still less robust than the implicit scheme and the smoothing property is not as good as the one by the implicit scheme. However, while considering which scheme is better, computation efficiency also needs to be taken into account. For the case shown in Fig. 7, calculation of total $(95 \times 30 \times 35)$ points takes 0.71 s of CPU time for the explicit scheme vs 7.7 s for the implicit one, on an HP 755 workstation. The difference is too significant to be ignored, not to mention that, considering the development of massive parallel computer, only the explicit scheme can take full advantage of this type computer.

VII. Conclusion

Methods of three-dimensional hyperbolic grid generation with inherent dissipation terms are developed in this paper for both the explicit and implicit schemes. Capabilities of both schemes have been demonstrated by applying them to different configurations. The explicit scheme, though less robust, is approximately an order of magnitude faster than the implicit scheme. For the flow solver incorporating a dynamic adaptive grid, this time difference will be more significant.

As demonstrated by the wing–body configuration, the method for the implicit scheme introduced in this paper has been proven to be very robust. The smoothing techniques developed in this paper provide good smoothing properties while reducing the complexity of the grid generation code.

Acknowledgment

The authors gratefully acknowledge the financial support of the National Science Council of the Republic of China under Grant NSC-85-2212-E-014-012.

References

[1]McNally, D., "FORTRAN Program for Generating a Two-Dimensional Orthogonal Mesh Between Two Arbitrary Boundaries," NASA TN D-6766, May 1972.

[2]Graves, A., Jr., "Application of a Numerical Orthogonal Coordinates Generator to Axisymmetry Blunt Bodies," NASA TM 80131, Oct. 1979.

[3]Steger, J. L., and Chaussee, D. S., "Generation of Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations," *SIAM Journal on Scientific and Statistical Computing*, Vol. 1, No. 4, 1980, pp. 431–437.

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

[5]Chan, W. M., and Steger, J. L., "Enhancements of a Three-Dimensional Hyperbolic Grid Generation Scheme," *Applied Mathematics and Computation*, Vol. 51, No. 1, 1992, pp. 181–205.

[6]Tai, C. H., and Yin, S. L., "Numerical Grid Generation for Hyperbolic Equation," *Third International Symposium on Explosive Technology and Ballistics, NIXT'91*, National Inst. for Explosive Technology, Pretoria, South Africa, 1991, pp. 671–678.

[7]Tai, C. H., Yin, S. L., and Soong, C. Y., "A Novel Hyperbolic Grid Generation Procedure with Inherent Adaptive Dissipation,"*Journal of Computational Physics*, Vol. 116, Jan. 1995, pp. 173–179.

[8]Jeng, Y. N., Shu, Y. L., and Lin, W. W., "Grid Generation for Internal Flow Problems by Methods Using Hyperbolic Equations," *Numerical Heat Transfer*, Vol. 27, Pt. B, 1995, pp. 43–61.

[9]Baily, F. R., and Simon, H. D., "Future Directions in Computing and CFD," AIAA Paper 92-2734, June 1992.

[10]Van Leer, B., Thomas, J. L., and Roe, P. L., "A Comparison of Numerical Flux Formulas for the Euler and Navier–Stokes Equations," AIAA Paper 87-1104, June 1987.