AIAA JOURNAL
Vol. 41, No. 1, January 2003

# Curvature-Based Wall Boundary Condition for the Euler Equations on Unstructured Grids

Z. J. Wang* and Yuzhi Sun†
*Michigan State University, East Lansing, Michigan 48824*

**A curvature-corrected symmetry technique developed by Dadone and Grossman (Dadone, A., and Grossman, B., "Surface Boundary Conditions for Compressible Flows," *AIAA Journal*, Vol. 32, No. 2, 1994, pp. 285–293) for a structured grid Euler solver has been extended to unstructured grids and to arbitrary curved boundaries. The local curvature is estimated numerically with a robust procedure and is used successfully in the boundary treatment. Numerical results for two-dimensional inviscid flows around circular and elliptic cylinders and the NACA0012 airfoil with the new boundary condition showed dramatic improvements over those with a conventional noncurvature-corrected approach. In all cases, spurious entropy productions with the new boundary treatment are significantly reduced, sometimes by several orders of magnitude.**

## Nomenclature

| | | |
|---|---|---|
| $E$ | = | total energy |
| $\boldsymbol{F}$ | = | vector of inviscid flux |
| $\boldsymbol{F}_f$ | = | vector of numerical inviscid flux through face $f$ |
| $I_{xx}$, $I_{xy}$, $I_{yy}$ | = | reconstruction coefficients |
| $M$ | = | matrix of reconstruction coefficients |
| $N$ | = | number of faces of a control volume |
| $\boldsymbol{n}$ | = | unit face normal, $(n_x, n_y)$ |
| $p$ | = | pressure |
| $\boldsymbol{Q}$ | = | vector of conserved variables |
| $\bar{\boldsymbol{Q}}$ | = | vector of cell-averaged conserved variables |
| $q$ | = | any of the primitive variables (density, pressure, and velocity components) |
| $R$ | = | local radius of curvature |
| $\boldsymbol{r}$ | = | position vector |
| $S_f$ | = | face area of face $f$ |
| $u$ | = | velocity component in $x$ direction |
| $V_i$ | = | volume of control volume $i$ |
| $v$ | = | velocity component in $y$ direction |
| $v_n$ | = | normal velocity to a face |
| $\gamma$ | = | ratio of specific heats |
| $\Delta n$ | = | distance between the ghost cell centroid and the interior cell centroid |
| $\rho$ | = | density |

*Subscripts*

| | | |
|---|---|---|
| $c$ | = | quantity at the cell centroid |
| $g$ | = | value at the ghost cell |
| $i$ | = | value at the cell $i$ |
| $L$ | = | value at the left side of a face |
| $R$ | = | value at the right side of a face |
| $w$ | = | value at a wall |

## I. Introduction

PROPER boundary treatment is critical in computational fluid dynamics (CFD). Its importance is obvious because it is the boundary conditions that determine the flow characteristics once the governing equations are given, at least for steady flow problems. Since the early days in CFD development, boundary conditions have been investigated intensively by many researchers.[1–7] For external aerodynamic problems, two types of boundary conditions play very important roles. One type is the nonreflective boundary condition and is used at the truncated far-field boundary. The other type is the nonpenetrating boundary condition, which is used at solid wall boundaries. Both boundary conditions have received much attention in the literature over the years. In many of the different treatments of boundary conditions, the characteristic analysis has been used to guide the development of physical and numerical boundary conditions. For example, the wall boundary condition developed by Chakravarthy[6] was implemented using both characteristic variables and physical variables. Another treatment of solid wall boundary conditions[7] used Taylor expansions and accuracy analysis. Most of the investigations on boundary conditions have focused on the finite difference method for structured grids.

More recently, the finite volume method has gained popularity because of its property of conservation and its easy extensibility to unstructured grids. Furthermore, the solid wall boundary condition in a finite volume framework is significantly simpler than that in a finite difference framework. This is because on a solid wall, which is part of a control volume, the mass flux and energy flux vanish. The only nonzero term is the pressure contribution to the momentum fluxes. Even the simplest treatment of a zeroth-order pressure extrapolation from the centroid of a wall-touching cell to a solid wall boundary gives acceptable results. However, such a treatment can seriously degrade solution accuracy and generate significant spurious entropy, as demonstrated by Dadone and Grossman.[8] They, therefore, developed a new boundary condition called the curvature-corrected symmetry technique (CCST) for solid walls with a nonzero curvature. In CCST, the pressure at a ghost cell is determined based on the local momentum equation, taking into account the curvature effects. Then the density and tangential velocity are computed using the condition of constant entropy and total enthalpy. Numerical results for flow around a circular cylinder with CCST showed dramatic improvements over those with the zeroth- and first-order pressure extrapolation techniques or a noncurvature corrected symmetry technique. CCST has since been successfully extended to three dimensions[9] and to noncircular geometries.[10] More recently, CCST has been implemented for Cartesian grids for rapid aerodynamic designs using the Euler equations.[11]

In the current paper, CCST is further extended to unstructured grids for arbitrary curved geometries with a numerically estimated local radius of curvature. The past one and one-half decades have seen a surge of activities in the area of CFD solution methodologies based on unstructured grids.[12–20] Unstructured grids provide considerable flexibility in tackling complex geometries and for adapting the computational grids according to flow features. Types of unstructured grids include classical triangular or tetrahedral grids, quadrilateral or hexahedral grids, prismatic grids, or mixed grids. Based

Received 27 October 2001; revision received 12 August 2002; accepted for publication 12 August 2002.

*Associate Professor, Department of Mechanical Engineering, 2555 Engineering Building; zjw@egr.msu.edu. Senior Member AIAA.

†Ph.D. Candidate, Department of Mechanical Engineering.

on experiences gained in the past decade, many CFD researchers have come to the conclusion that mixed grids (or hybrid grids)[16] are the way to go. In the pursuit of the ultimate flow solver, we have developed a finite volume flow solver that is capable of handling arbitrary grids,[19,20] including hybrid adaptive Cartesian/prism and viscous Cartesian grids. The CCST boundary condition is implemented in this flow solver and tested for general curved geometries.

The paper is, therefore, arranged in the following manner. In the next section, the finite volume flow solver supporting arbitrary unstructured grids is briefly described. Then CCST is extended to arbitrary unstructured grids. To handle general curved boundaries, a technique to estimate the local curvature numerically is developed. After that, the general CCST treatment is tested for several geometries, including a circular and an elliptic cylinder and the NACA0012 airfoil. Finally, several concluding remarks based on the current study are included.

## II. Finite Volume Solver on Arbitrary Grids

The Euler equations governing inviscid flow can be written in the following integral form:

$$\int_V \frac{\partial \boldsymbol{Q}}{\partial t}\,\mathrm{d}V + \int_S \boldsymbol{F}\,\mathrm{d}S = 0 \tag{1}$$

where $\boldsymbol{Q}$ and $\boldsymbol{F}$ are given by

$$\boldsymbol{Q} = \{\rho, \rho u, \rho v, E\}^T$$

$$\boldsymbol{F} = \{\rho v_n, \rho u v_n + p n_x, \rho v v_n + p n_y, (p+E) v_n\}^T \tag{2}$$

The total energy is computed from

$$E = p/(\gamma - 1) + \tfrac{1}{2}\rho(u^2 + v^2) \tag{3}$$

with $\gamma = 1.4$ for air. The integration of Eq. (1) in an arbitrary control volume $V_i$ with $N$ faces gives

$$\frac{\mathrm{d}\bar{\boldsymbol{Q}}_i}{\mathrm{d}t} V_i + \sum_{f=1}^{N} \boldsymbol{F}_f S_f = 0 \tag{4}$$

The overbar in $\bar{\boldsymbol{Q}}_i$ will be dropped from here on, and $\boldsymbol{Q}_i$ is interpreted to represent the vector of conserved variables at the cell centroid of $V_i$ without loss of (second-order) accuracy. The focus of the numerical approach is then the computation of the numerical flux $\boldsymbol{F}_f$, which is computed with a Godunov-type[21] approach. There are two key components in a Godunov scheme.[21] One is data reconstruction, and the other is the Riemann solver. The original Godunov scheme[21] employed a piecewise constant data reconstruction, and the resultant scheme was only first-order accurate. Van Leer extended the first-order Godunov scheme to second-order[22] by using a piecewise linear data reconstruction. The second-order Godunov scheme is adopted in the flow solver. The two major ingredients of the flow solver, data reconstruction and Riemann solver, are briefly described in the following subsections.

### A. Reconstruction

In a cell-centered finite volume method, flow variables are known in a cell-average sense. No indication is given as to the distribution of the solution over the control volume. To evaluate the inviscid flux through a face using the Godunov[21] approach, flow variables are required at both sides of the face. This task is fulfilled through data reconstruction. In this paper, a least-squares reconstruction algorithm capable of preserving a linear function on arbitrary grids is employed. This linear reconstruction also makes the finite volume method second-order accurate in space. The reconstruction problem reads as follows: Given cell-averaged primitive variables (denoted by $q$) for all of the cells of the computational grid, build a linear distribution for each cell, for example, $c$, using data at the cell itself and its neighboring cells sharing a face with $c$. We use that the cell-averaged solutions can be taken to be the point solutions at the cell centroids without sacrificing the second-order accuracy. Therefore, we seek to reconstruct the gradient $(q_x, q_y)$ for cell $c$, which produces the following linear distribution:

$$q(x, y) = q_c + q_x(x - x_c) + q_y(y - y_c) \tag{5}$$

where $(x_c, y_c)$ are the cell-centroid coordinates. The following expressions can be easily derived from a least-squares approach:

$$\begin{bmatrix} q_x \\ q_y \end{bmatrix} = M \begin{bmatrix} \displaystyle\sum_{f=1}^{N} (q_{f,c} - q_c)(x_{f,c} - x_c) \\ \displaystyle\sum_{f=1}^{N} (q_{f,c} - q_c)(y_{f,c} - y_c) \end{bmatrix} \tag{6}$$

where $q_{f,c}$ is the primitive variable at a neighboring cell sharing face $f$ with cell $c$, $x_{f,c}$ and $y_{f,c}$ are the coordinates of the cell centroid, and

$$M = \frac{1}{\Delta}\begin{bmatrix} I_{yy} & -I_{xy} \\ -I_{xy} & I_{xx} \end{bmatrix}$$

with

$$I_{xx} = \sum_{f=1}^{N} (x_{f,c} - x_c)^2, \qquad I_{xy} = \sum_{f=1}^{N} (x_{f,c} - x_c)(y_{f,c} - y_c)$$

$$I_{yy} = \sum_{f=1}^{N} (y_{f,c} - y_c)^2, \qquad \Delta = I_{xx} I_{yy} - I_{xy}^2 \tag{7}$$

Note that matrix $M$ is symmetric and dependent only on the computational grid. If one stores three elements of $M$ for each cell, the reconstruction can be performed efficiently through a loop over all faces.

### B. Riemann Solver

In the original Godunov[21] method, an exact Riemann solver is used to solve the Euler equations with the following initial condition at $t = 0$:

$$\boldsymbol{Q} = \begin{cases} \boldsymbol{Q}_L & \text{if} \quad x < 0 \\ \boldsymbol{Q}_R & \text{otherwise} \end{cases} \tag{8}$$

The numerical flux is then taken to be the exact flux at $x = 0$ when $t > 0$. When the Godunov method is generalized to multiple dimensions, a one-dimensional Riemann solver in the face normal direction is used. Therefore, the flux at a face with normal $\boldsymbol{n}$ can be expressed as

$$\boldsymbol{F}_f = \boldsymbol{F}_{\text{Riem}}(\boldsymbol{Q}_L, \boldsymbol{Q}_R, \boldsymbol{n}) \tag{9}$$

Although the exact Riemann solver is very effective, it is expensive to compute because an iterative Newton solver is necessary to solve the nonlinear equations. As a result, other, more efficient approximate Riemann solvers have been developed to compute the flux. One popular one is Roe's approximate Riemann solver,[23] which is used in the present study.

For time integration, an efficient block lower–upper symmetric Gauss–Seidel (LU-SGS) implicit scheme[24] is used on arbitrary grids. This block LU-SGS scheme takes much less memory than a fully (linearized) implicit scheme, albeit having essentially the same or better convergence rate than a fully implicit scheme because of its nonlinear nature.

## III. Curvature-Based Wall Boundary Condition for Unstructured Grids

The CCST developed by Dadone and Grossman[8] is now extended for treating the wall boundary condition for an unstructured-grid finite volume flow solver. Consider a wall boundary shown in Fig. 1. The unit normal $\boldsymbol{n}$ of the boundary points out of the computational domain. Cell $i$ is the interior cell, and cell $g$ is the ghost cell employed

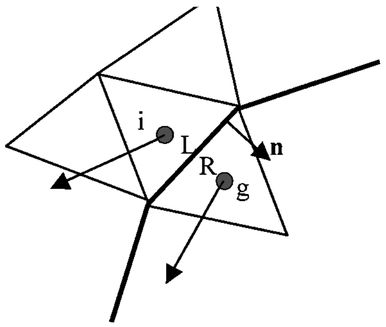


**Fig. 1 Schematic of wall boundary condition.**

to handle the boundary condition. The ghost cell is produced as the mirror image of the interior cell with respect to the boundary face. In the following subsections, both the conventional symmetry technique (ST) and CCST are described.

## A. ST

In a noncurvature-correctd boundary treatment, the boundary face is considered flat, and the flow variables at the ghost cell are just mirror images of those at the interior cell. Let the pressure, density, and velocity vector in the interior cell $i$ be $p_i$, $\rho_i$, and $\boldsymbol{v}_i$. Then the flow variables at the ghost cell $g$ are computed with

$$p_g = p_i, \qquad \rho_g = \rho_i, \qquad \boldsymbol{v}_g = \boldsymbol{v}_i - 2(\boldsymbol{v}_i \cdot \boldsymbol{n})\boldsymbol{n} \tag{10}$$

Then the least-squares linear reconstruction approach is used to reconstruct the gradients of the primitive variables for cell $i$. After that, the primitive variables at the boundary face center reconstructed from cell $i$ can be expressed as

$$p_L = p_i + (\boldsymbol{r}_f - \boldsymbol{r}_i)\cdot\nabla p_i, \qquad \rho_L = \rho_i + (\boldsymbol{r}_f - \boldsymbol{r}_i)\cdot\nabla\rho_i$$
$$u_L = u_i + (\boldsymbol{r}_f - \boldsymbol{r}_i)\cdot\nabla u_i, \qquad v_L = v_i + (\boldsymbol{r}_f - \boldsymbol{r}_i)\cdot\nabla v_i \tag{11}$$

where $\boldsymbol{r}_f$ is the position vector of the face center and $\boldsymbol{r}_i$ is the position vector of the cell centroid. The reconstructed normal component of the velocity vector at the face center may not vanish, that is, $u_L n_x + v_L n_y \neq 0$. To force the mass and energy fluxes to vanish at the wall boundary, the primitive variables just to the right of the boundary face are computed, assuming they are again mirror images of the variables just to the left of the face, that is,

$$p_R = p_L, \qquad \rho_R = \rho_L, \qquad \boldsymbol{v}_R = \boldsymbol{v}_L - 2(\boldsymbol{v}_L \cdot \boldsymbol{n})\boldsymbol{n} \tag{12}$$

Then, the final flux vector at the face is computed from an approximate Riemann solver:

$$\boldsymbol{F}_f = \boldsymbol{F}_{\text{Riem}}(\boldsymbol{Q}_L, \boldsymbol{Q}_R, \boldsymbol{n}) \tag{13}$$

Because of the use of linear reconstructions of primitive variables at the interior cell, this boundary condition is compatible with the interior numerical scheme and is second-order accurate, which has been numerically verified.

## B. CCST

The basic idea of CCST is to use the local momentum equation to specify the pressure at the ghost cell. Therefore, the following equation is applied locally:

$$\frac{\partial p}{\partial n} = -\frac{\rho|\boldsymbol{v}|^2}{R} \tag{14}$$

Applying this equation to the ghost cell, we then obtain

$$p_g = p_i - \Delta n \rho_w |\boldsymbol{v}_w|^2 \big/ R \tag{15}$$

In our implementation, $\rho_w$ and $|\boldsymbol{v}_w|$ are chosen to be the density and tangential velocity at cell $i$, that is,

$$\rho_w = \rho_i, \qquad |\boldsymbol{v}_w| = |\boldsymbol{v}_i - (\boldsymbol{v}_i \cdot \boldsymbol{n})\boldsymbol{n}| \tag{16}$$

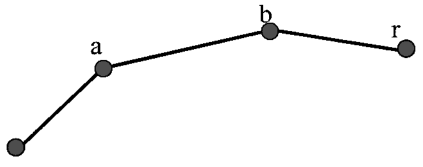


**Fig. 2 Estimation of local curvature.**

The density at the ghost cell is computed assuming that the entropy is the same as that in the interior cell, that is,

$$\rho_g = \rho_i (p_g / p_i)^{1/\gamma} \tag{17}$$

The velocity vector at the ghost cell is still computed according to Eq. (10). Again, Eqs. (11) and (12) are used to compute ($\boldsymbol{Q}_L, \boldsymbol{Q}_R$), which are used in the approximate Riemann solver. If the computational grid is locally orthogonal near the boundary, the reconstructed normal velocity component should be zero.

If the geometry is a circular cylinder, the radius of the local curvature is simply the radius of the cylinder. For an arbitrary curved boundary, other means must be developed to estimate the local curvature. Consider a curved boundary shown in Fig. 2. To estimate the curvature for boundary face a–b, first we use a–b and the point to the left of the boundary face (point l) to make one estimate, and then we use a–b and the right point r to perform another estimate. Then we simply average the two estimates to obtain the final radius. However, if either point a or b is a sharp corner (such as the trailing edge of an airfoil), then we use the estimate from the three points, avoiding the sharp corner. If both a and b are corner points, then the radius is infinity, that is, the curvature is zero. A point is considered a sharp corner if the two edges sharing the point form an angle larger than a threshold, such as 30 deg.

Given arbitrary three noncolinear points 1, 2, and 3, we can find the radius by solving the following equations:

$$(x_1 - x_0)^2 + (y_1 - y_0)^2 = R^2, \qquad (x_2 - x_0)^2 + (y_2 - y_0)^2 = R^2$$
$$(x_3 - x_0)^2 + (y_3 - y_0)^2 = R^2 \tag{18}$$

The solution can be expressed as

$$R = \Big\{\big[(x_1 - x_2)^2 + (y_1 - y_2)^2\big]\big[(x_1 - x_3)^2 + (y_1 - y_3)^2\big]$$
$$\times \big[(x_3 - x_2)^2 + (y_3 - y_2)^2\big]\Big\}^{\frac{1}{2}}$$
$$\times 1/\{2[x_3(y_2 - y_1) + x_2(y_1 - y_3) + x_1(y_3 - y_2)]\} \tag{19}$$

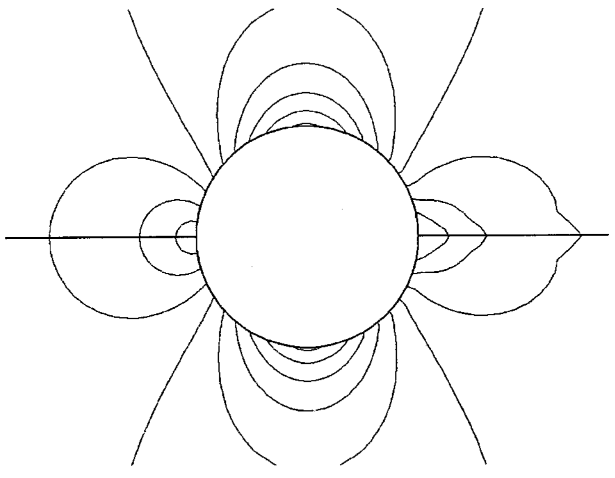
In the actual implementation, the curvature is used instead of the radius to prevent division by 0. Note that the actual pressure used in the momentum flux depends on the local mesh, which is used in the linear reconstruction, as well as the approximate Riemann solver.

## IV. Test Results

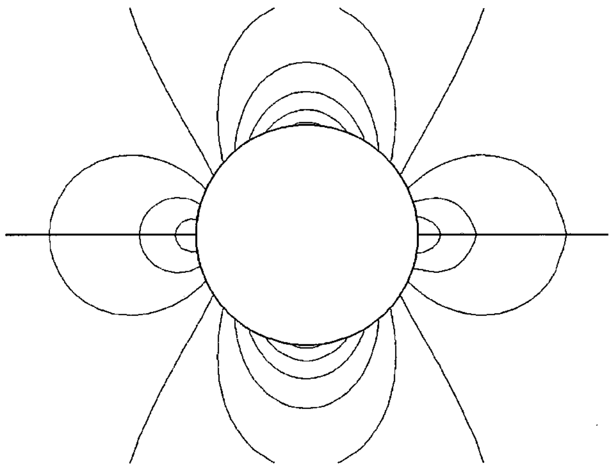
We have considered three representative test cases to validate and demonstrate the new curvature-based wall boundary condition. In all three cases, dramatic improvements over the noncurvature-based boundary condition have been achieved.

### A. Subsonic Flow over a Circular Cylinder

This case is the same as the one used to demonstrate CCST in Ref. 8, and it is chosen here as a validation case of the present implementation. The freestream Mach number is 0.38. The computations were performed with a grid of 128 × 32 cells (128 cells in the circumferential direction of 360 deg and 32 cells in the radial direction). The truncated far-field boundary is 20 radii away from the cylinder. Both ST and CCST were employed in the computations. The Mach number contours computed with both boundary conditions are compared in Fig. 3, and the relative entropy error contours are displayed in Fig. 4. In Figs. 3 and 4, the contours for the same variables are plotted at the same levels. Note that the Mach contours computed with CCST are much more symmetric than those computed with the conventional ST, indicating there is much less

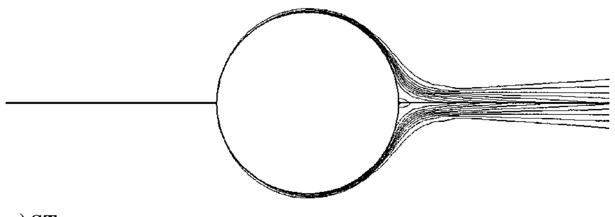


a) ST

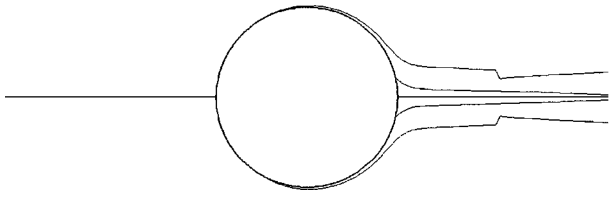


b) CCST

Fig. 3 Mach contours on the fine quadrilateral mesh with 128 × 32 cells.



a) ST



b) CCST

Fig. 4 Relative entropy error contours on the fine quadrilateral mesh with 128 × 32 cells.

spurious entropy production in the solution with CCST. This observation is visually verified by Fig. 4. In fact, the maximum relative entropy error and drag coefficient in the solution with ST are 0.0036 and 0.0080, whereas the entropy error and drag coefficient are only 0.0010 and 0.0047 in the solution with CCST. The case serves to verify the present implementation.

## B. Inviscid Flow Around an Elliptic Cylinder

The second case is an inviscid flow problem around an elliptic cylinder, with a major radius of 1 (in the $x$ direction) and a minor radius of 2, as shown in Fig. 5, which also displays the coarse quadrilateral and triangular grids with 64 × 16 and 64 × 16 × 2 cells. The coarse mesh has 64 cells in the circumferential direction. The far-field boundary extends to 20 major radii away. The freestream Mach number was chosen to be 0.26 so that the flow can reach a maximum Mach number close to 1. Three quadrilateral grids were employed in a grid refinement study. The medium and fine grids have 128 × 32 and 256 × 64 cells, respectively. The simulations all started from a uniform freestream, and the residuals were reduced by at least six orders of magnitude. The computed drag coefficients and maximum relative entropy errors with both boundary conditions on all three grids are plotted in Fig. 6. As expected, CCST produced less drag on the same grid than ST. What is remarkable in the comparison is the maximum relative entropy error. The coarsest mesh with CCST produced less entropy error than the finest mesh with ST. In fact, the entropy errors on the medium and fine mesh with CCST are more than an order of magnitude smaller than those with ST. Figures 7, 8, and 9 show the computed Mach, density, and relative entropy error contours on the finest mesh with both boundary conditions, respectively. Even on the finest mesh, two spurious separation bubbles still exist on the downwind side of the ellipse with ST, as shown in the entropy error contours in Fig. 9a. The spurious separation bubbles are illustrated in the velocity vector plot on the medium mesh shown in Fig. 10a. Note that CCST was capable of producing an attached flow, as shown in Fig. 10b. The same simulation was also performed with the finest triangular grid, which has 256 × 64 × 2 cells. The computed Mach number and entropy error contours are displayed in Figs. 11 and 12. By examining the Mach contours in Figs. 7 and 11, we can tell that the solution quality on the triangular grid is better because there are less wiggles in those contours. Again, the maximum entropy error with CCST is more than an order lower than that with ST on this triangular grid.

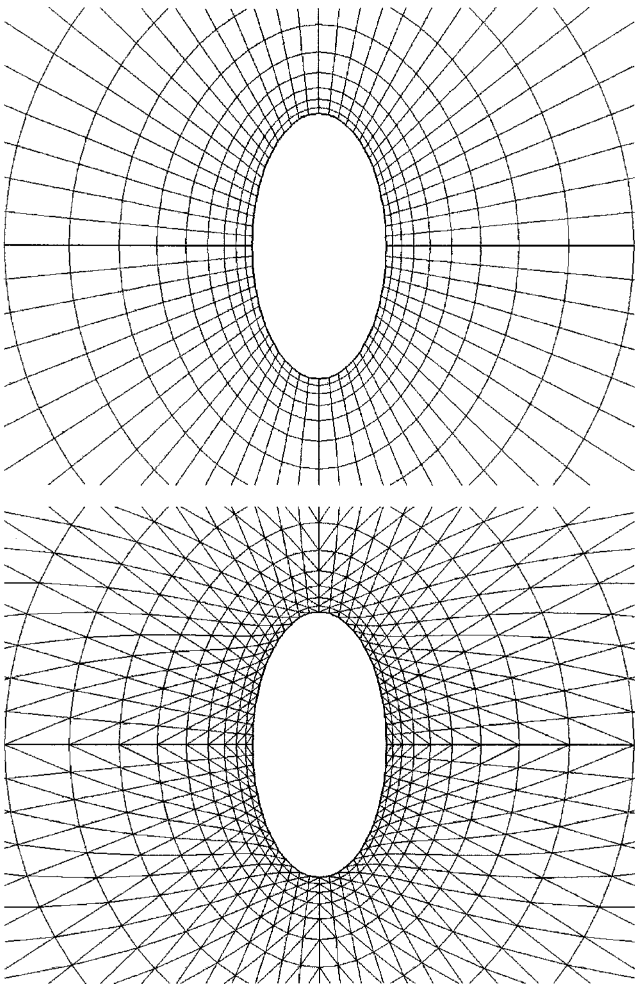
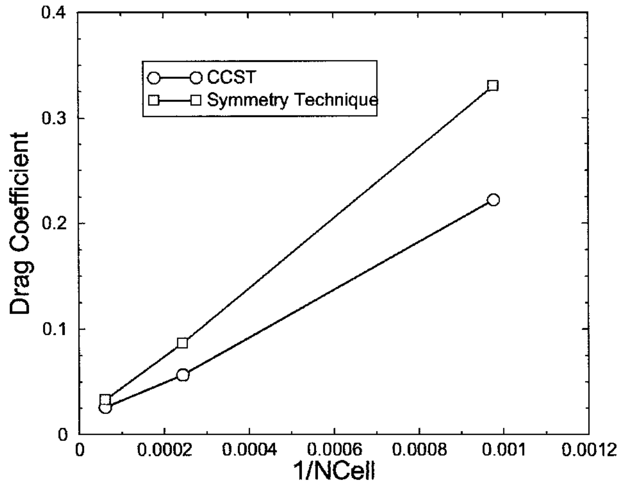
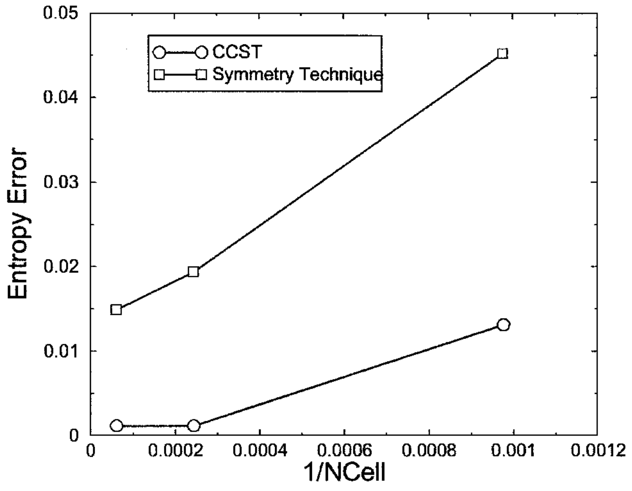


Fig. 5 Coarse computational grids around an elliptic cylinder.

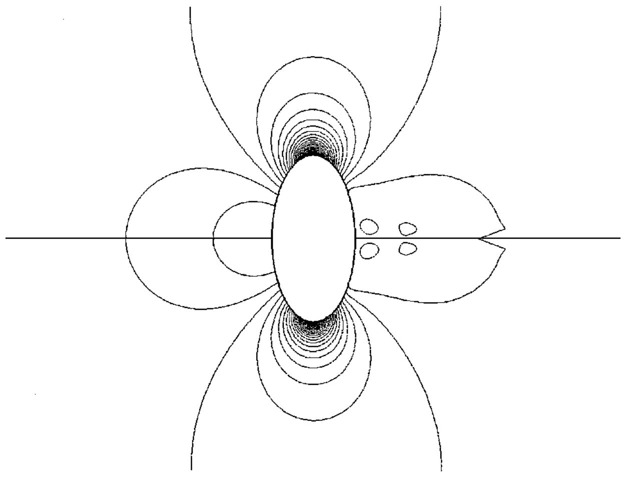


a)

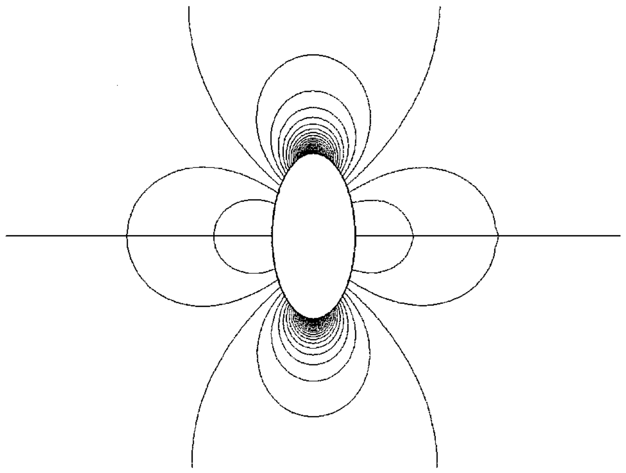


b)

Fig. 6 Comparison of a) drag coefficients and b) entropy errors.

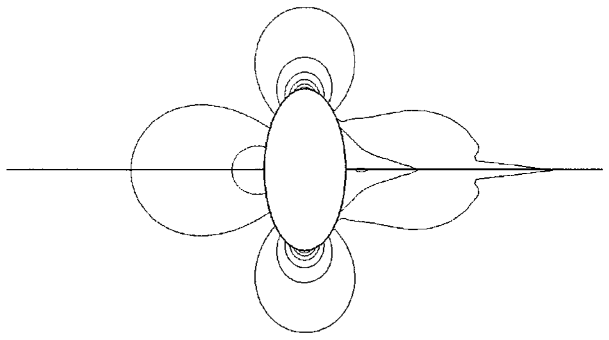


a) ST

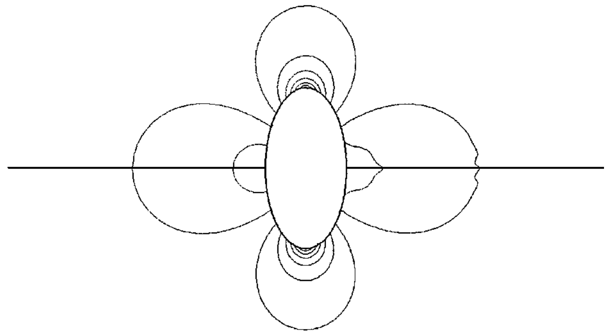


b) CCST

Fig. 8 Density contours on the fine quadrilateral mesh with 256 × 64 cells.

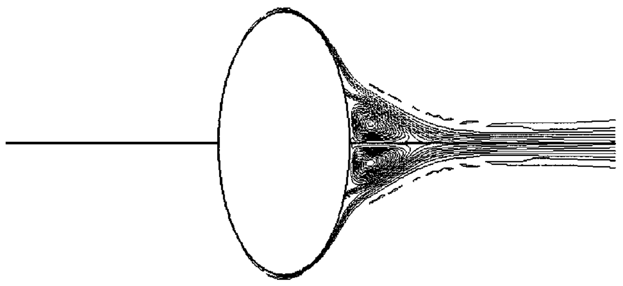


a) ST

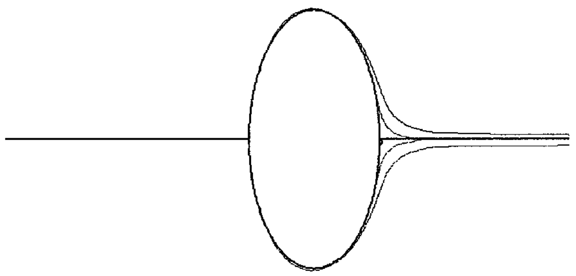


b) CCST

Fig. 7 Mach contours on the fine quadrilateral mesh with 256 × 64 cells.

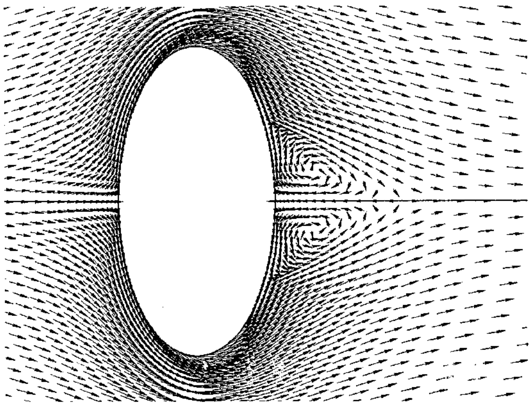


a) ST



b) CCST

Fig. 9 Entropy error contours on the fine quadrilateral mesh with 256 × 64 cells.


a) ST

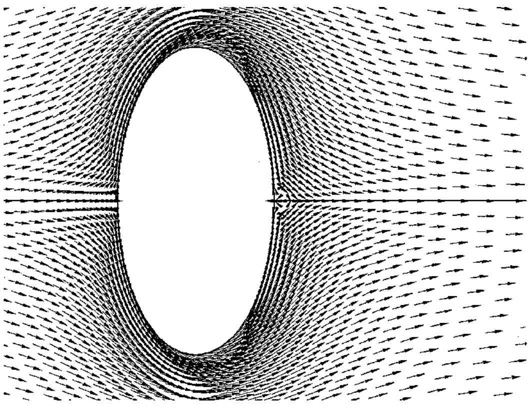
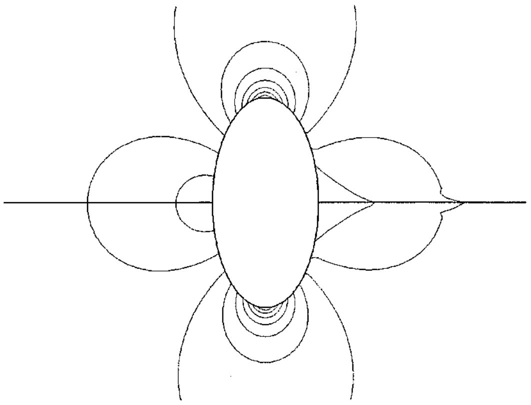

b) CCST

Fig. 10 Velocity vector plots on the medium quadrilateral mesh with 128 × 32 cells.


a) ST

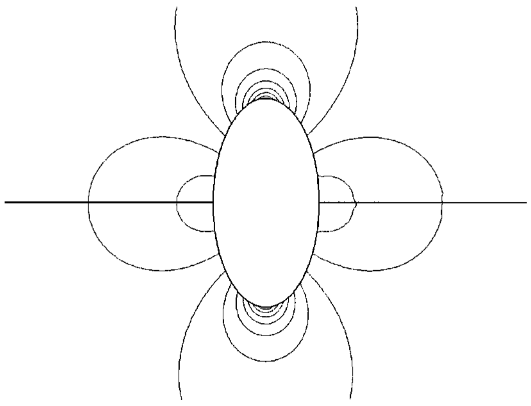
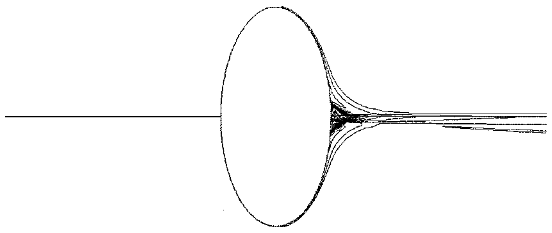

b) CCST

Fig. 11 Mach contours on the fine triangular mesh with 256 × 64 × 2 cells.


a) ST

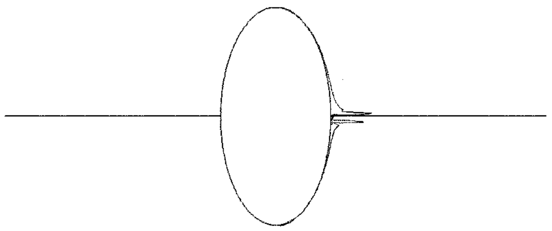

b) CCST

Fig. 12 Entropy error contours on the fine triangular mesh with 256 × 64 × 2 cells.

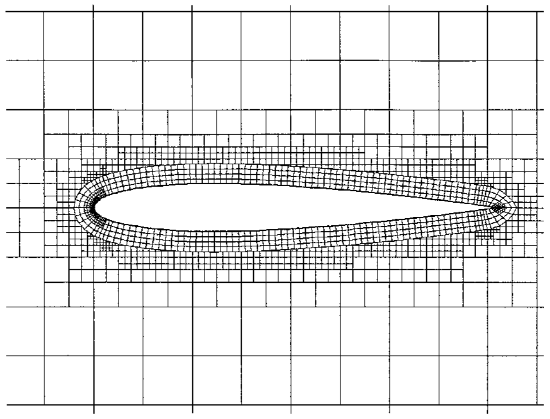
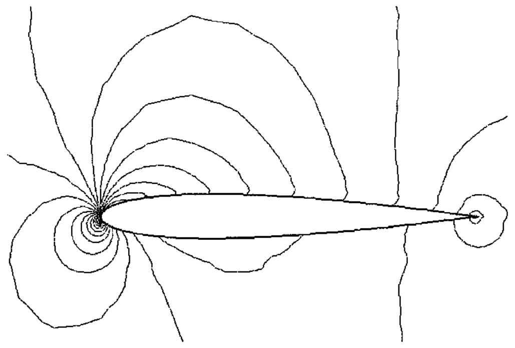


Fig. 13 Adaptive Cartesian/adaptive quad mesh for the NACA0012 airfoil.


a) ST

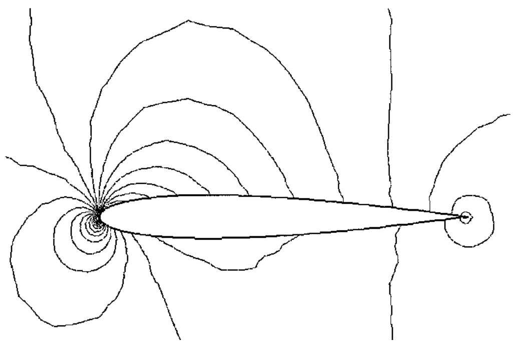
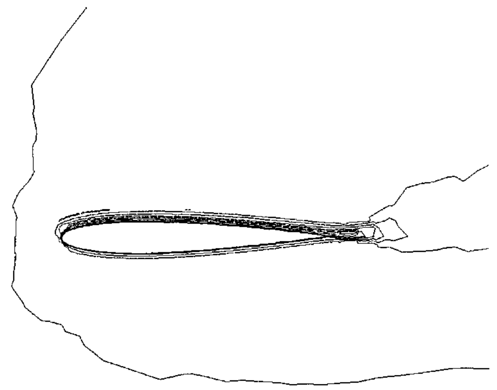

b) CCST

Fig. 14 Mach contours on the adaptive Cartesian/quad mesh.


a) ST

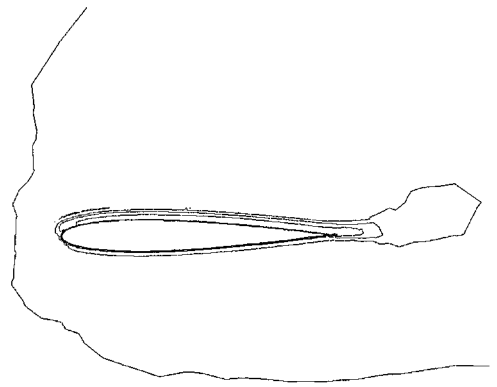

b) CCST

**Fig. 15 Entropy error contours on the adaptive Cartesian/quad mesh.**

### C. Subsonic Flow Around a NACA0012 Airfoil

Finally, to demonstrate the new boundary condition for a general curved geometry, subsonic flow around the NACA0012 airfoil is simulated with a freestream Mach number of 0.5 and an angle of attack of 3 deg. Near the sharp trailing edge, the curvature is estimated, avoiding the sharp corner. An adaptive Cartesian/adaptive quad computational mesh shown in Fig. 13 was used in this simulation. The mesh has a total of 2184 cells, 4887 faces, and 2703 nodes. The mesh was generated using a grid generation method presented in Ref. 19. Cell cutting was used near the outer boundary of the quad grid to "merge" the Cartesian and quad grid into a single grid with arbitrary polygons. The computed Mach number contours and the entropy error contours with both boundary conditions are displayed in Figs. 14 and 15, respectively. Note that the computed Mach contours with ST have sharp turns near the airfoil surface, indicating the generation of an entropy layer near the surface, whereas the contours with CCST are nearly straight near the surface. It is clear from the entropy error contours that the new boundary condition produced much less spurious entropy than the conventional boundary condition.

## V. Conclusions

A curvature-based boundary condition has been successfully extended to unstructured grids and to general curved geometries. A robust numerical procedure has also been developed to estimate the local curvature for an arbitrary two-dimensional curve. Numerical demonstrations with two-dimensional inviscid flow around an ellipse and the NACA0012 airfoil showed dramatic improvements over a conventional noncurvature-corrected approach. In all cases, spurious entropy productions with the new boundary treatment are significantly reduced, by up to several orders of magnitude more than with the noncurvature-based boundary condition.

## Acknowledgments

The first author gratefully acknowledges the startup funding from the Department of Mechanical Engineering, College of Engineering of Michigan State University. The authors also thank Michigan State University for an Intramural Research Grants Program grant and Ashok Singhal of CFD Research Corporation for granting a free software license of CFD-ACEU and CFD-FASTRAN.

## References

[1]Moretti, G., and Abbett, M., "Time-Dependent Computational Method for Blunt-Body Flows," *AIAA Journal*, Vol. 4, No. 12, 1966, pp. 2136–2141.

[2]Gottlieb, D., and Turkel, E., "Boundary Conditions for Multi-Step Finite Difference Methods for Time-Dependent Equations," *Journal of Computational Physics*, Vol. 26, 1978, pp. 181–196.

[3]Gustafsson, B., and Kreiss, H.-O., "Boundary Conditions for Time-Dependent Problems with an Artificial Boundary," *Journal of Computational Physics*, Vol. 30, 1979, pp. 333–351.

[4]Moretti, G., "The $\lambda$-Scheme," *Computers and Fluids*, Vol. 8, 1979, pp. 191–205.

[5]Yee, H. C., Beam, R. M., and Warming, R. F., "Boundary Approximations for Implicit Schemes for One-Dimensional Inviscid Equations of Gasdynamics," *AIAA Journal*, Vol. 20, No. 9, 1982, pp. 1203–1211.

[6]Chakravarthy, S. R., "Euler Equations: Implicit Schemes and Boundary Conditions," *AIAA Journal*, Vol. 21, No. 5, 1983, pp. 699–706.

[7]Sparis, P. D., "Second-Order Accurate Boundary Conditions for Compressible Flows," *AIAA Journal*, Vol. 22, No. 9, 1984, pp. 1222–1228.

[8]Dadone, A., and Grossman, B., "Surface Boundary Conditions for the Numerical Solution of the Euler Equations," *AIAA Journal*, Vol. 32, No. 2, 1994, pp. 285–293.

[9]Dadone, A., and Grossman, B., "Surface Boundary Conditions for the Numerical Solution of the Euler Equations in Three Dimensions," *Proceedings of the Fourteenth International Conference on Numerical Methods in Fluid Mechanics*, Springer-Verlag, New York, 1994, pp. 188–194.

[10]Dadone, A., "Symmetry Techniques for the Numerical Solution of the 2-D Euler Equations at Impermable Boundaries," *International Journal for Numerical Methods in Fluids*, Vol. 28, No. 7, 1998, pp. 1093–1108.

[11]Dadone, A., and Grossman, B., "An Immersed Body Methodology for Inviscid Flows on Cartesian Grids," AIAA Paper 2002-1059, Jan. 2002.

[12]Barth, T. J., and Frederickson, P. O., "High-Order Solution of the Euler Equations on Unstructured Grids Using Quadratic Reconstruction," AIAA Paper 90-0013, Jan. 1990.

[13]Rausch, R. D., Batina, J. T., and Yang, H. T., "Spatial Adaptation Procedures on Unstructured Meshes for Accurate Unsteady Aerodynamic Flow Computation," AIAA Paper 91-1106, July 1991.

[14]Aftosmis, M., Gaitonde, D., and Tavares, S., "On the Accuracy, Stability and Monotonicity of Various Reconstruction Algorithms for Unstructured Meshes," AIAA Paper 94-0415, Jan. 1994.

[15]Luo, H., Sharov, D., Baum, J. D., and Lohner, R., "On the Computation of Compressible Turbulent Flows on Unstructured Grids," AIAA Paper 2000-0927, Jan. 2000.

[16]Kallinderis, Y., Khawaja, A., and McMorris, H., "Hybrid Prismatic/Tetrahedral Grid Generation for Complex Geometries," *AIAA Journal*, Vol. 34, No. 2, 1996, pp. 291–298.

[17]Venkatakrishnan, V., and Mavriplis, D. J., "Implicit Solvers for Unstructured Meshes," *Journal of Computational Physics*, Vol. 105, No. 1, 1993, pp. 83–91.

[18]Frink, N. T., "Assessment of an Unstructured-Grid Method for Predicting 3-D Turbulent Viscous Flows," AIAA Paper 96-0292, Jan. 1996.

[19]Wang, Z. J., "A Quadtree-Based Adaptive Cartesian/Quad Grid Flow Solver for Navier–Stokes Equations," *Computers and Fluids*, Vol. 27, No. 4, 1998, pp. 529–549.

[20]Wang, Z. J., "A Fast Nested Multi-Grid Viscous Flow Solver for Adaptive Cartesian/Quad Grids," *International Journal for Numerical Method in Fluids*, Vol. 33, No. 5, 2000, pp. 657–680.

[21]Godunov, S. K., "A Finite-Difference Method for the Numerical Computation of Discontinuous Solutions of the Equations of Fluid Dynamics," *Matematicheskie Sbornik*, Vol. 47, 1959, pp. 271–301.

[22]Van Leer, B., "Towards the Ultimate Conservative Difference Scheme V. A Second Order Sequel to Godunov's Method," *Journal of Computational Physics*, Vol. 32, No. 1, 1979, pp. 101–136.

[23]Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372.

[24]Chen, R. F., and Wang, Z. J., "Fast, Block Lower-Upper Symmetric Gauss–Seidel Scheme for Arbitrary Grids," *AIAA Journal*, Vol. 38, No. 12, 2000, pp. 2238–2245.

P. Givi
*Associate Editor*