SIAM J. SCI. COMPUT.
Vol. 31, No. 5, pp. 3979–3999
© 2009 Society for Industrial and Applied Mathematics

THREE-DIMENSIONAL ADAPTIVE CENTRAL SCHEMES ON UNSTRUCTURED STAGGERED GRIDS*

AZIZ MADRANE† AND RÉMI VAILLANCOURT‡

Abstract. We present an explicit second-order finite volume generalization of the one-dimensional (1D) Nessyahu–Tadmor schemes for hyperbolic equations on adaptive unstructured tetrahedral grids. The nonoscillatory central difference scheme of Nessyahu and Tadmor, in which the resolution of the Riemann problem at the cell interfaces is bypassed thanks to the use of the staggered Lax–Friedrichs scheme, is extended here to a two-steps scheme. In order to reduce artificial viscosity, we start with an adaptively refined primal grid in three dimensions (3D), where the theoretical *a posteriori* result of the first-order scheme is used to derive appropriate refinement indicators. We apply those methods to solve Euler's equations. Numerical experimental tests on classical problems are obtained by our method and by the computational fluid dynamics software Fluent. These tests include results for the 3D Euler system (shock tube problem) and flow around an NACA0012 airfoil.

Key words. 3D adaptive central schemes, unstructured staggered grid mesh adaptation, finite volume methods

AMS subject classifications. Primary, 65M55; Secondary, 65M60, 76M12

DOI. 10.1137/06066240X

1. Introduction.

The history of schemes on staggered grids can at least be traced back to the famous paper of Courant, Friedrichs, and Lewy in 1928 [13] in which they discovered a scheme on staggered grids for the linear wave equation in one-dimensional (1D). For a special system arising in fluid dynamic problems von Neumann and Richtmyer used staggered grids as well [38]. Four years later, Lax introduced the well-known Lax–Friedrichs scheme and analyzed it [29]. In 1990 Tadmor and Nessyahu [41] picked up the idea to use staggered grids, showed the connection to Godunov's method, and proposed a second-order extension to 1D systems.

The main advantage of these schemes is that no information about solutions to local Riemann problems is needed. Using staggered grids one can replace the upwind fluxes by central differences. The price one has to pay is the occurrence of excessive numerical viscosity since the resulting scheme can be interpreted as a Lax–Friedrichs scheme. Therefore, a higher order scheme of monotone upstream-centered schemes for conservation laws (MUSCL)-type in one spatial dimension was proposed in [41]. Later in [4, 3, 6, 7, 8, 36] central schemes were generalized to multidimensional schemes on unstructured grids. For a Cartesian grid, we refer to [9, 22, 30, 28, 31, 39, 40, 45, 32] for related work.

In the case of staggered unstructured multidimensional grids, there exist only a few convergence results. In [5] convergence of a second-order central scheme on two-dimensional (2D) grids has been proven for a linear conservation law. Convergence of the first-order Lax–Friedrichs scheme on the same staggered grids for nonlinear scalar problems has been proven in [19].

*Received by the editors June 7, 2006; accepted for publication (in revised form) August 17, 2009; published electronically November 11, 2009. This work was supported in part by NSERC of Canada and the Centre de recherches mathématiques of the Université de Montréal.
http://www.siam.org/journals/sisc/31-5/66240.html

†Institut for Aerospace-Technology, Airbus, Flughafenallee 10, D-28199 Bremen, Germany (Aziz.madrane@airbus.com).

‡Mathematics and Statistics, University of Ottawa, Ottawa, ON K1N 6N5, Canada (remi@uottawa.ca).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

The major drawback of central schemes is their numerical dissipation due to staggering. In order to eliminate this disadvantage, we implement an adaptive refinement strategy based on local *a posteriori* error estimates [25, 27, 26], which is a rigorous error estimator for central schemes on staggered grids.

In many applications, error, shock, or grid indicators have been used in order to find those regions with steep gradients (see, for example, [39, 40, 34, 37, 24, 44]). Usually these indicators are based on discrete gradients to control the local process of grid refinement. But these indicators give no information about the true error $\|u-u_h\|$.

Adaptive grid technology is a powerful tool in computational fluid dynamics, which provides three important benefits: automation, improved efficiency, and increase solution accuracy. In general, most adaptive methods fall into three broad categories: grid movement (r-refinement), grid enrichment (h-refinement), and local solution enhancement (p-refinement). While the methods in the first two classes modify the grid density to improve the solution accuracy, those in the third category enhance the order of numerical approximation at the location where the solution undergoes abrupt variations. Most adaptive techniques used in the computational fluid dynamic applications fall into the first two classes.

In the grid movement approach, nodes are redistributed and moved towards regions where a higher degree of accuracy is needed. Since the grid topology remains unchanged throughout the grid adaptation, the process of grid movement can be simply incorporated into the solver in a modular fashion. In addition, no data transfer is required since the grid structure remains intact during the process. Since the number of grid nodes remains constant, transferring nodes from one part of the grid to another may cause local depletion of grid elements, and thus severe distortion of the grid may be introduced.

In the grid enrichment technique, more nodes are added to the regions where higher accuracy of the solution is desired. Nodes can also be removed from locations where the solution is smooth and requires less grid resolution. Due to node addition or deletion, the topology of the grid changes from one adaptation cycle to another. Consequently, interpolation of data between consecutive grids is required which curtails the applicability of these methods for unsteady problems. Adaptive methods by grid enrichment are particularly attractive for their flexibility, especially when applied in conjunction with unstructured grids. In the present work, an attempt has been made to combine the efficiency of h-refinement and the flexibility of remeshing for solution adaptive refinement. The focus of this paper is on the refinement mechanism aspect of the solution adaptive problem as applied to 3D problems. Our results are found to be in excellent agreement with previous simulations of the flow around NACA0012 airfoil carried out with upwind-type schemes (see [2]).

The paper is organized as follows. Section 2 describes the mathematical model. Section 3 deals with space and time discretization. In section 4, we present the mesh adaptation algorithm. Numerical results are in section 5.

2. Mathematical modeling.

2.1. Governing equations. Let $\Omega \subset \mathbb{R}^3$ be the domain of interest of the flow with boundary Γ. We write $\Gamma = \Gamma_B \cup \Gamma_\infty$, where Γ_B denotes the part of the body boundary which is relevant for the computational domain and Γ_∞ is the (upwind) farfield boundary. Three-dimensional compressible inviscid flows are described by Euler's equations, written in their conservation form [43]

$$\frac{\partial U}{\partial t} + \nabla \cdot \overrightarrow{\mathcal{F}}(U) = 0, \tag{1}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

where

$$U = (\rho, \rho u, \rho v, \rho w, E)^T, \quad \overrightarrow{\mathcal{F}}(U) = (F(U), G(U), H(U))^T.$$

Here $\overrightarrow{\mathcal{F}}(U)$ denotes the convective flux [43], ρ is the density, $\overrightarrow{V} = (u, v, w)^T$ is the velocity vector, E is the total energy per unit volume, and p is the pressure of the fluid. Let A, B, and C denote the Jacobian matrices $\partial F(U)/\partial U$, $\partial G(U)/\partial U$, and $\partial H(U)/\partial U$, respectively. Then (1) can be written in the nonconservative form

$$\frac{\partial U}{\partial t} + \overrightarrow{\mathcal{F}}'(U) \cdot \nabla U = \frac{\partial U}{\partial t} + A(U)\frac{\partial U}{\partial x} + B(U)\frac{\partial U}{\partial y} + C(U)\frac{\partial U}{\partial z} = 0. \tag{2}$$

2.2. Boundary conditions. The flow is assumed to be uniform at the farfield boundary Γ_∞, and we impose the following three conditions:

$$\rho_\infty = 1, \quad \overrightarrow{V}_\infty = \begin{bmatrix} \cos\alpha\cos\beta \\ \sin\beta \\ \sin\alpha\cos\beta \end{bmatrix}, \quad p_\infty = \frac{1}{\gamma M_\infty^2}, \tag{3}$$

where α is the angle of attack, β is the yaw angle, and M_∞ denotes the free-stream Mach number. On the wall boundary Γ_B, we use the slip condition $\overrightarrow{V} \cdot \vec{n} = 0$ where $\vec{n}$ is the outward unit vector normal to the boundary $\Gamma_\infty \cup \Gamma_B$. Finally, for unsteady calculations, an initial flow, $U(x, 0) = U_0(x)$, is prescribed on Ω.

3. Space and time discretization.

3.1. Definitions. We assume that Ω is a bounded polyhedral domain of $\mathbb{R}^3$ and we start from an arbitrary FEM tetrahedral grid $\mathcal{T}_h$, where h is the maximal length of the edges in $\mathcal{T}_h$.

A dual finite volume partition is derived from the construction of median planes, that is, for every vertex i of $\mathcal{T}_h$, a cell C_i is defined around i as follows.

Every tetrahedron having i as a vertex is subdivided into 24 subtetrahedra by planes containing an edge and the midpoint of the opposite edge; then the cell C_i is the union of subtetrahedra having i as a vertex (see Figure 1).

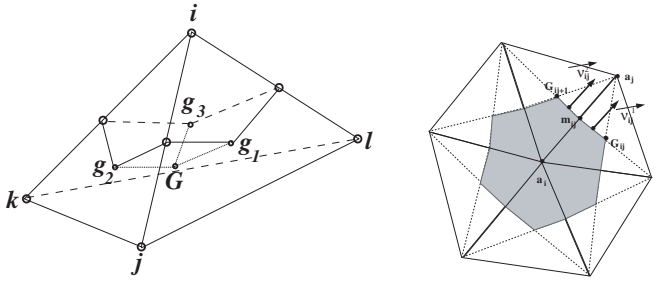

FIG. 1. *Barycentric cells C_i in 3D and 2D.*

In particular, the boundary ∂C_i of C_i is the union of $\partial C_{ij} = \partial C_i \cap \partial C_j$ that can be defined as the union of triangles (see Figure 2) such that

- one vertex is the midpoint of the edge $[i, j]$,
- one vertex is the barycenter of the tetrahedron T having $[i, j]$ as an edge, and
- one vertex is the barycenter of a (triangular) face of T having $[i, j]$ as a side.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

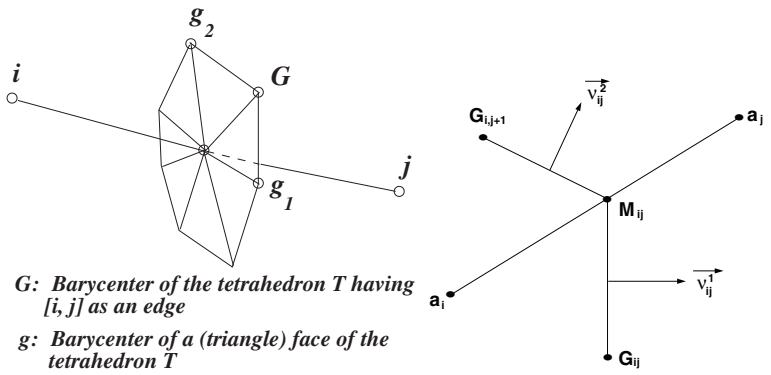

FIG. 2. *Part of a boundary of* C_i, $\partial C_{ij} = \partial C_i \cap \partial C_j$ *in* 3*D and* 2*D.*

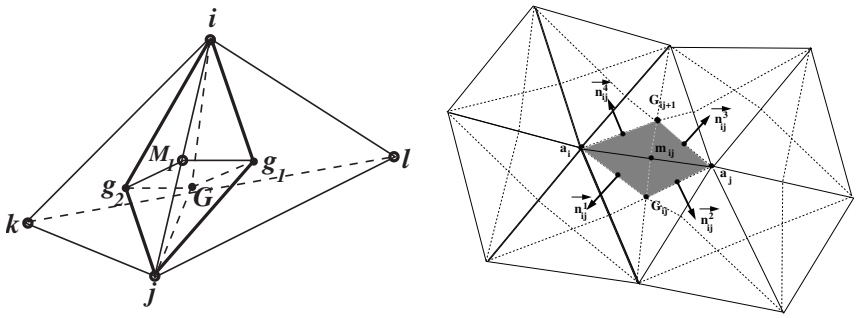

FIG. 3. *Diamond cells* L_{ij} *in* 3*D and* 2*D.*

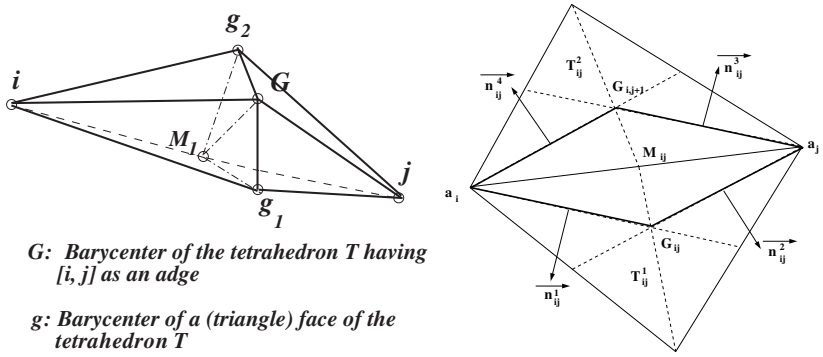

FIG. 4. *Boundary of cell* L_{ij} *in* 3*D and* 2*D.*

As for 2D extensions [4], the present 3D extension also uses a dual grid, with dual cells L_{ij} associated with the edges of $\mathcal{T}_h$. The dual ("diamond") cell L_{ij} is composed of four subtetrahedra iGM_1g_1, iGM_1g_2, jGM_1g_1, jGM_1g_2 (defined above) sharing edge $[i, j]$ (see Figures 3 and 4). For complete details of the domain of computation for the NACA0012 airfoil in the 2D and 3D cases see Figures 5 and 6.

The following notation will be needed.

NOTATION 1. *Let* i, j, k, l *be the four nodes defining a tetrahedron* τ, $\tau \in \mathcal{T}_h$. *Then*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

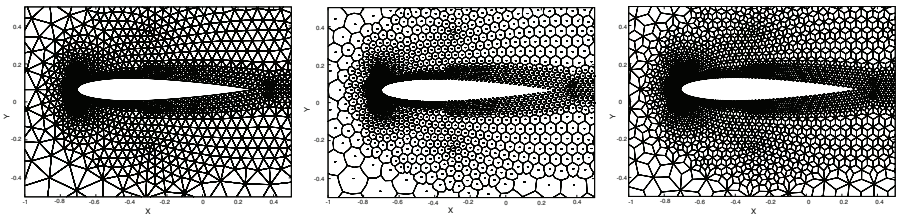

FIG. 5. *NACA*0012 *airfoil. Primary grid, barycentric cell* C_i*, and diamond cell* L_{ij} *in* 2*D.*

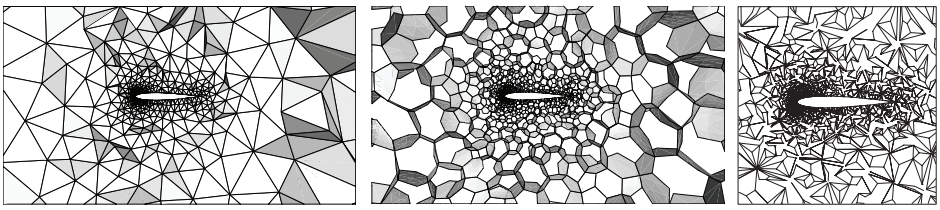

FIG. 6. *NACA*0012 *airfoil. Primary grid, barycentric cell* C_i*, and diamond cell* L_{ij} *in* 3*D.*

- T_{ij} *denotes the set of all tetrahedra which share edge* $[i,j]$ *as a common edge,*
- $K(i)$ *is the set of nodes (vertices) which are neighbors of node* i*,*
- $C_i = \bigcup_{j\in K(i)} (C_i \cap L_{ij})$ *and*
- $\partial C_i = \bigcup_{j\in K(i)} \{\partial C_i \cap \partial C_j\} \cup \{\partial C_i \cap \Gamma_B\} \cup \{\partial C_i \cap \Gamma_\infty\}$,
- $L_{ij} = \bigcup_{\tau\in T_{ij}} (L_{ij} \cap C_i \cap \tau) \cup (L_{ij} \cap C_j \cap \tau)$ *and*
- $\partial L_{ij} = \bigcup_{\tau\in T_{ij}} (\partial L_{ij} \cap \tau) \cup (\partial L_{ij} \cap \Gamma_\infty) \cup (\partial L_{ij} \cap \Gamma_B)$,
- $\vec{n}_{ij} = (n_{ij_x}, n_{ij_y}, n_{ij_z})$ *is the unit outward normal vector to* ∂L_{ij}*,*
- $\vec{\nu}_i = (\nu_{i_x}, \nu_{i_y}, \nu_{i_z})$ *is the unit outward normal vector to* ∂C_i*.*

Let m_{ij} denote the midpoint of edge $[i,j]$, also written as M_1 in Figures 3 and 4, and let $U_i^n \cong U(a_i, t^n)$ and $U_{ij}^{n+1} \cong U(m_{ij}, t^{n+1})$ denote the nodal (resp. cell average) values in the first and second grids at time $t = t^n$ and $t = t^{n+1}$, respectively (n even).

The union of all the barycentric cells constitutes a partition of the computational domain Ω_h and the same holds for diamond cells

$$\Omega_h = \bigcup_{i=1}^{\mathrm{nv}} C_i, \qquad \Omega_h = \bigcup_{k=1}^{\mathrm{ne}} L_k,$$

where nv and ne are the number of vertices and the number of edges, respectively, of the original finite element triangulation $\mathcal{T}_h$.

Remark. The quadrangle $\partial C_{ij} \cap \tau = M g_1 G g_2$ is planar because

$$\overrightarrow{g_1 M_1} \times \overrightarrow{g_2 M_1} = 2(\overrightarrow{g_2 G} \times \overrightarrow{g_1 G}).$$

3.2. High-order accurate approximations. Now we can define the two steps of our high-order accurate (staggered, Lax–Friedrichs type) finite volume method. To obtain second-order accuracy, we introduce cellwise piecewise linear interpolation

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

(MUSCL, [46]), where the gradient can be estimated by least squares [20, 4] and Green–Gauss' method [11].

First step: We integrate (1) on an extended control volume $L_{ij} \times [t^n, t^{n+1}]$, assuming we have obtained, from the cell average values U_i^n, piecewise linear reconstructions given by

$$U_h(\vec{x}, t^n)\big|_{C_i} = \mathcal{L}_i(\vec{x}, t^n) = U_i^n + \nabla U_i^n \cdot (\vec{x} - \vec{x}_i), \qquad \forall \vec{x} \in C_i,\ \vec{x} \in \mathbb{R}^3. \tag{4}$$

For the integration with respect to time, in order to ensure "nearly" second-order accuracy, we adopt a "quasi-midpoint formula" time discretization, where the convective flux is computed at the intermediate time $t^{n+1/2}$, thus requiring the computation of **predicted** values $U_h(\vec{x}, t^{n+1/2})$ given ∂L_{ij}.

Predictor's first step: On each face of the cell L_{ij}, using Euler's equations, we define a predicted vector

$$U_{i,g_1,G}^{n+1/2} = U_{i,g_1,G}^n - \frac{\Delta t}{2} \overrightarrow{\mathcal{F}}'(U_{i,g_1,G}^n) \cdot \nabla U_i^n, \tag{5}$$

where, by (4), the value of U_h^n is taken equal to

$$U_h(\vec{x}, t^n) \cong U_i^n + \nabla U_i^n \cdot (\vec{x}_{i,g_1,G} - \vec{x}_i) \equiv U_{i,g_1,G}^n \tag{6}$$

along the face i, g_1, G of the diamond cell L_{ij}.

Corrector's first step: By (5) the corrector can be written as follows:

$$\begin{aligned} \mathrm{Vol}(L_{ij}) U_{ij}^{n+1} &- \sum_{\tau \in T_{ij}} \left[\int_{L_{ij} \cap C_i \cap \tau} \mathcal{L}(\vec{x}, t^n)\, d\vec{x} + \int_{L_{ij} \cap C_j \cap \tau} \mathcal{L}(\vec{x}, t^n)\, d\vec{x} \right] \\ &+ \Delta t \sum_{\tau \in T_{ij}} \left[\int_{\partial L_{ij}^1 \cap \tau} + \int_{\partial L_{ij}^2 \cap \tau} + \int_{\partial L_{ij}^3 \cap \tau} + \int_{\partial L_{ij}^4 \cap \tau} \right] \overrightarrow{\mathcal{F}} \left(U_h^{n+1/2} \right) \cdot \vec{n}_{ij}\, dA \\ &+ \Delta t \int_{\partial L_{ij} \cap \Gamma_B} \overrightarrow{\mathcal{F}}(U_h^n) \cdot \vec{n}\, dA + \Delta t \int_{\partial L_{ij} \cap \Gamma_\infty} \overrightarrow{\mathcal{F}}(U_h^n) \cdot \vec{n}\, dA = 0, \end{aligned} \tag{7}$$

where $\partial L_{ij}^1 \cap \tau = ig_1GM1, L_{ij}^2 \cap \tau = ig_2GM1, L_{ij}^3 \cap \tau = jg_1GM1,$ and $L_{ij}^3 \cap \tau = jg_2GM1$.

Using the midpoint rule, we can approximate the volume integrals as follows:

$$\begin{aligned} \int_{L_{ij} \cap C_i \cap \tau} \mathcal{L}(\vec{x}, t^n)\, d\vec{x} &= \int_{(L_{ij} \cap C_i \cap \tau)_{\mathrm{left}}} \mathcal{L}(\vec{x}, t^n)\, d\vec{x} + \int_{(L_{ij} \cap C_i \cap \tau)_{\mathrm{right}}} \mathcal{L}(\vec{x}, t^n)\, d\vec{x}, \\ \int_{(L_{ij} \cap C_i \cap \tau)_{\mathrm{left}}} \mathcal{L}(\vec{x}, t^n)\, d\vec{x} &= \mathrm{Vol}((L_{ij} \cap C_i \cap \tau)_{\mathrm{left}} \mathcal{L}(x_{C_1}, t^n), \end{aligned} \tag{8}$$

where $(L_{ij} \cap C_i \cap \tau)_{\mathrm{left}} = ig_1M_1G$ and $(L_{ij} \cap C_i \cap \tau)_{\mathrm{right}} = ig_2M_1G$, and C_1 is the barycenter of the subtetrahedron ig_1M_1G (see Figure 1).

For the flux integral we use the same procedure as for the volume integral,

$$\int_{\partial L_{ij} \cap \tau} \overrightarrow{\mathcal{F}} \left(U_h^{n+1/2} \right) \cdot \vec{n}_{ij} dA = \mathrm{Area}(\partial L_{ij} \cap \tau) \overrightarrow{\mathcal{F}} \left(U_{S_1}^{n+1/2} \right) \cdot \vec{n}_{ij}, \tag{9}$$

where S_1 is the barycenter of the subtriangle ig_1G (see Figure 2).

Second step: To obtain the second step of the time discretization, we integrate (1) on the cell $C_i \times [t^{n+1}, t^{n+2}]$, assuming that, from the diamond cell average values

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

U_{ij}^{n+1} computed in the first time step, we have obtained piecewise linear reconstructions given by

$$U_h(\vec{x}, t^{n+1})|_{L_{ij}} = \mathcal{L}_{ij}(\vec{x}, t^{n+1}) = U_{ij}^{n+1} + \nabla U_{ij}^{n+1} \cdot (\vec{x} - \vec{x}_{ij}). \tag{10}$$

Predictor's second step: Proceeding as in the first step, we obtain the predictor's second step:

$$U_{M_1,g_1,G}^{n+3/2} = U_{M_1,g_1,G}^{n+1} - \frac{\Delta t}{2} \overrightarrow{\mathcal{F}}' \left(U_{M_1,g_1,G}^{n+1} \right) \cdot \nabla U_{ij}^{n+1}, \tag{11}$$

where

$$U_h(\vec{x}, t^{n+1}) \cong U_{ij}^{n+1} + \nabla U_{ij}^{n+1} \cdot (\vec{x}_{M_1,g_1,G} - \vec{x}_{M_1}) \equiv U_{M_1,g_1,G}^{n+1} \tag{12}$$

defines an approximation to the value of U on the boundary element $[M_1, g_1, G]$ of the cell C_i.

Corrector's second step: The second step is

$$\begin{aligned} &\mathrm{Vol}(C_i) U_i^{n+2} - \sum_{j \in K(i)} \int_{C_i \cap L_{ij}} \mathcal{L}_{ij}(\vec{x}, t^{n+1})\, d\vec{x} \\ &\quad + \Delta t \sum_{j \in K(i)} \int_{\partial C_i \cap \partial C_j} \overrightarrow{\mathcal{F}}(U(\vec{x}, t^{n+3/2}) \cdot \vec{\nu}_i\, dA \\ &\quad + \Delta t \int_{\partial C_i \cap \Gamma_B} \overrightarrow{\mathcal{F}} \left(U_h^{n+1} \right) \cdot \vec{\nu}\, dA + \Delta t \int_{\partial C_i \cap \Gamma_\infty} \overrightarrow{\mathcal{F}} \left(U_h^{n+1} \right) \cdot \vec{\nu}\, dA = 0, \end{aligned} \tag{13}$$

where the volume and the boundary integrals are computed as above.

3.2.1. Approximation of the slopes and limitation. In order to compute the gradient ∇U_i^n of the piecewise linear interpolant $L(\vec{x}, t^n)$ for the cell C_i, we use least squares [20, 4] and Green–Gauss' method [11]. For the limitation we use several procedures, see [46]. Numerical experiments have led us to choose Green–Gauss' method for the gradients used in the reconstruction for the cells C_i and a least squares weighted procedure for the cells L_{ij}.

(a) Green–Gauss' method for cells C_i: The gradient vector ∇U_i of U at the point i is obtained by Green–Gauss' method

$$\begin{aligned} \nabla U_i &= \frac{1}{\mathrm{Vol}(C_i)} \sum_{j \in K(i)} \int_{\partial C_i \cap \partial C_j} U \cdot \vec{\nu}_i\, dA \\ &= \frac{1}{\mathrm{Vol}(C_i)} \sum_{j \in K(i)} \mathrm{Area}(\partial C_i \cap \partial C_j) \frac{(U_i + U_j) \cdot \vec{\nu}_i}{2}, \end{aligned} \tag{14}$$

where $\mathrm{Vol}(C_i)$ is the volume of the dual cell C_i around P_i.

(b) Least squares method for cells L_{ij}: This algorithm uses a Taylor expansion of a function U from a local point to each surrounding point [1] including its neighboring point

$$U_{jl} = U_{ij} + \nabla U_{ij} \cdot (\vec{x}_{jl} - \vec{x}_{ij}) + O\left((\vec{x}_{jl} - \vec{x}_{ij})^2 \right). \tag{15}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

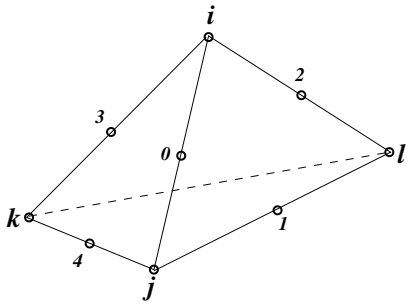

FIG. 7. *Surrounding face points used for the least squares algorithm.*

The system of linear equations derived from all neighboring face points surrounding edge $[i, j]$ (see Figure 7) can be expressed by the following system:

$$\underbrace{\begin{bmatrix} \Delta x_{01} & \Delta y_{01} & \Delta z_{01} \\ \Delta x_{02} & \Delta y_{02} & \Delta z_{02} \\ \vdots & \vdots & \vdots \\ \Delta x_{0N} & \Delta y_{0N} & \Delta z_{0N} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{\partial U}{\partial z} \end{bmatrix}_{ij}}_{\vec{x}} = \underbrace{\begin{bmatrix} U_1 - U_0 \\ U_2 - U_0 \\ \vdots \\ U_N - U_0 \end{bmatrix}}_{\vec{b}}, \tag{16}$$

where $\Delta x_{01} = x_1 - x_0$.

The solution of the linear system $A\vec{x} = \vec{b}$ can be obtained by using Householder's QR transformation (to avoid conditioning problem for highly stretched meshes), where the matrix $Q \in \mathbb{R}^{N\times 3}$ has orthonormal columns and the matrix $R \in \mathbb{R}^{3\times 3}$ is upper triangular.

(c) Slope limiter: To prevent oscillations in nonsmooth regions, we introduce a slope limiter ϕ_i for both gradients. Then the reconstruction for cell C_i can be written as

$$U_h(\vec{x}, t^n)|_{C_i} = \mathcal{L}_i(\vec{x}, t^n) = U_i^n + \phi_i \nabla U_i^n \cdot (\vec{x} - \vec{x}_i), \quad \forall \vec{x} \in C_i,\ \vec{x} \in \mathbb{R}^3, \tag{17}$$

with the limiter $\phi_i \in [0, 1]$. To avoid creating a new extremum, the reconstruction $U_h(\vec{x}, t^n)|_{C_i}$ should be bounded by the local extrema

$$\min_{j\in K(i)} \{U_i, U_j\} \le U_h(\vec{x}, t^n)|_{C_i} \le \max_{j\in K(i)} \{U_i, U_j\}. \tag{18}$$

The monotonicity of U can be satisfied by enforcing (18) at quadrature points. The limiter function ϕ_i proposed by Venkatakrishnan [47] is currently used:

$$\phi_i = \min_{j\in K(i)} \begin{cases} \dfrac{1}{\Delta_2}\left[\dfrac{(\Delta_{1,\max}^2 + \epsilon^2)\Delta_2 + 2\Delta_2^2\Delta_{1,\max}}{\Delta_{1,\max}^2 + 2\Delta_2^2 + \Delta_{1,\max}\Delta_2 + \epsilon^2}\right] & \text{if} \quad U_j > U_i, \\ \dfrac{1}{\Delta_2}\left[\dfrac{(\Delta_{1,min}^2 + \epsilon^2)\Delta_2 + 2\Delta_2^2\Delta_{1,\min}}{\Delta_{1,\min}^2 + 2\Delta_2^2 + \Delta_{1,\min}\Delta_2 + \epsilon^2}\right] & \text{if} \quad U_j < U_i, \\ 1 & \text{if} \quad U_j = U_i, \end{cases} \tag{19}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

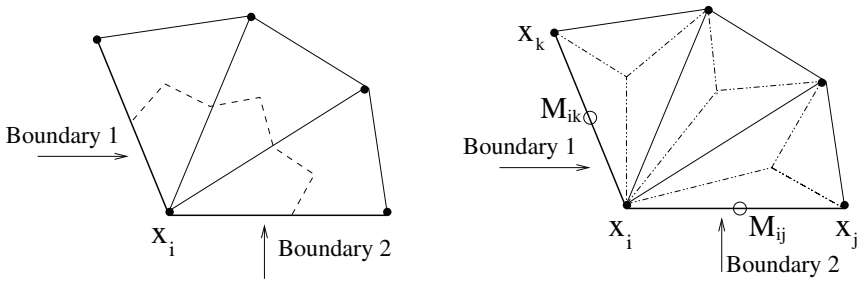

FIG. 8. *Implementation of boundary conditions for barycentric and diamond cells.*

where

$$\begin{aligned}
\Delta_2 &= \frac{1}{2}\nabla U_i \cdot (\vec{x}_j - \vec{x}_i),\\
\Delta_{1,\max} &= \max\{U_i, \max_{j\in K(i)} U_j\} - U_i,\\
\Delta_{1,\min} &= \min\{U_i, \min_{j\in K(i)} U_j\} - U_i.
\end{aligned} \tag{20}$$

A small number, ϵ^2, prevents division by zero when the gradient is very small. In the implementation, ϵ^2 is set to be a function of the local length scale,

$$\epsilon^2 = (k\,\delta h)^3, \tag{21}$$

where k is a user-specified constant and δh is the local mesh length scale. The present work uses $k = 5$ and $(\delta h)^3 = \mathrm{Vol}(C_i)$.

3.2.2. Treatment of the boundary conditions. The treatment of the boundary conditions is different from barycentric cells, C_i, to diamond cells, L_{ij}. Since a vertex-based (barycentric cells) approach results in degrees of freedom being located directly on the boundary (see Figure 8), this would seem to facilitate the implementation of the boundary conditions. However, a more fundamental problem arises for barycentric cells C_i, since individual boundary vertices may have ill-defined boundary conditions if they are located at the intersection of two faces with different boundary conditions as shown in the left of Figure 8. This situation never occurs for diamond cells (see the right of Figure 8). In order to implement such boundary condition, in the case of barycentric cells, a weak formulation must be used:

$$\int_{\partial C_i \cap \Gamma_B} \overrightarrow{\mathcal{F}}(U_h^{n+1}) \cdot \vec{\nu}_i \, dA = \int_{\partial C_i \cap \Gamma_B} \begin{bmatrix} 0 \\ p\nu_x \\ p\nu_y \\ p\nu_z \\ 0 \end{bmatrix}^{n+1} dA. \tag{22}$$

We shall use the following approximation:

$$\int_{\partial C_i \cap \Gamma_\infty} \overrightarrow{\mathcal{F}}(U_h^{n+1}) \cdot \vec{\nu}_i \, dA = \mathcal{A}^+(U_i, \vec{\nu}_{i\infty})U_i + \mathcal{A}^-(U_i, \vec{\nu}_{i\infty})U_\infty, \tag{23}$$

where

$$\mathcal{A}(U_i, \vec{\nu}_{i\infty}) = \overrightarrow{\mathcal{F}}'(U_i) \cdot \int_{\partial C_i \cap \Gamma_\infty} \vec{\nu}_i \, dA$$

and $\mathcal{A}^+$ and $\mathcal{A}^-$ are the positive and negative parts of $\mathcal{A}$, respectively [18].

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

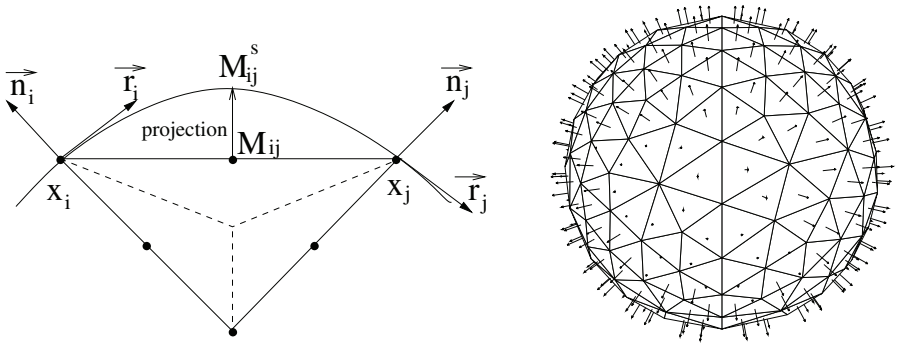

FIG. 9. *Treatment of a curved boundary for diamond cell* L_{ij}.

Another problem arises for the treatment of a curved boundary in the case of diamond cells L_{ij} as the midpoint of the cell L_{ij} is not located anymore on the boundary as shown in (Figure 9).

Remark on the treatment of a curved boundary. For a curved boundary, the midpoint M_{ij} of the diamond cell L_{ij} may not be located on the surface geometry of the model to be simulated because it was located at the midpoints of existing edges (see Figure 9). To address this issue, we have implemented a boundary curvature correction based on Hermite interpolation [35]. The projected point M_{ij}^s is defined by

$$x_{M_{ij}^s} = 0.5(x_i + x_j) + 0.125(r_i - r_j),$$

where

$$r_l = |s| \frac{n_l \times (s \times n_l)}{|n_l \times (s \times n_l)|}, \quad s = x_i - x_j, \quad (l = i, j),$$

and n_l is the normal to the node l to be the weighted average of the normals of the surface elements sharing that node. That is, for each node l the normal n_l is defined by

$$n_l = \frac{1}{\sum_{\tau \in T_l} |\tau|} \sum_{\tau \in T_l} n_\tau |\tau|,$$

where $|\tau|$ is the size of the surface element τ, T_l denotes the set of all the surface elements of the mesh that contains l, n_τ is the outer normal of the surface element τ, and x_i is the coordinate of the node i (see Figure 9).

3.2.3. Stability. We refer to [5] for a stability study of linear multidimensional advection models that is also valid in the 3D case. For the second time-step we take

$$\Delta t_i = \mathrm{CFL}_{C_i} \mathrm{Vol}(C_i) \Big/ \left(\lambda_{\max}^i \int_{\partial C_i} dA \right),$$

where

$$\lambda_{\max}^i = \max \left\{ \lambda_i, \max_{j \text{ neighbor of } i} \lambda_j \right\}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

and

$$\lambda_i = \|\overrightarrow{V}_i\| + c_i.$$

Here $\overrightarrow{V}_i$ and c_i refer to the values of the velocity vector and sound speed, respectively, in cell C_i. We then choose $\Delta t = \min_{1\le i\le \mathrm{nv}}\{\Delta t_i\}$, with a similar time-step definition for the first (odd) step ($t^n \to t^{n+1}$).

4. Mesh adaptation algorithm.

4.1. General description. The theory behind the mesh adaptation technique for central schemes on unstructured staggered grids has been developed in [25, 27]. We introduce the following three main steps of this technique.

- First, a strategy to determine where a modification is needed in the field of the grid, e.g., by means of an (*a posteriori*) error estimate.
- Secondly, a rule that selects the elements or edges in $\mathcal{T}_h$ (marking strategy).
- Thirdly, a rule that refines the elements in $\mathcal{T}_h$ (refinement strategy).

4.1.1. A posteriori error estimate. For stationary problems, following the theory of [25, 27], for each edge $e_{ij} \in \mathcal{T}_h$, we have the error estimate $\eta_{e_{ij}}$:

$$\|u - u_h\|_{L^1(e_{ij})} \le \eta(u_{e_{ij}}) = aQ + b\sqrt{Q}, \tag{24}$$

where $a = 2 + 2\omega$, $\omega = 0.5$, $b = 4 + 2^d$, $d = 3$,

$$\begin{aligned} Q =& \frac{1}{2}\sum_{e_{ij}=1}^{ne} h_{e_{ij}} \mathrm{Vol}(L_{e_{ij}}) \sum_{\tau\in T_{ij}} \frac{\mathrm{Vol}(L_{e_{ij}}\cap C_i\cap\tau)}{\mathrm{Vol}(L_{e_{ij}})}\frac{\mathrm{Vol}(L_{e_{ij}}\cap C_j\cap\tau)}{\mathrm{Vol}(L_{e_{ij}})}|u_i - u_j| \\ &+ \sum_{e_{ij}=1}^{ne}\mathrm{Vol}(L_{e_{ij}})\left|u_{e_{ij}} - \sum_{j\in K(i)}\frac{\mathrm{Vol}(L_{e_{ij}}\cap C_i)}{\mathrm{Vol}(L_{e_{ij}})}u_{e_{ij}}\right| \\ &+ 6\sum_{e_{ij}=1}^{ne} h_{e_{ij}} \sum_{\tau\in T_{ij}} \mathrm{Area}(\partial C_i\cap\partial C_j\cap\tau)|u_i - u_j|, \end{aligned} \tag{25}$$

and $h_{e_{ij}} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$.

For transient problems, following the theory of [25, 27], for each edge $e_{ij} \in \mathcal{T}_h$, we have the error estimate $\eta_{e_{ij}}$:

$$\|u - u_h\|_{L^1(e_{ij})} \le \eta(u_{e_{ij}}) = T\left(\|u_0 - u_h(\cdot,0)\|_{L^1(e_{ij})} + aQ + \sqrt{bcQ}\right), \tag{26}$$

where a, b, and c are given in [25, 27], and Q is

$$\begin{aligned} Q =& \frac{1}{2}\sum_{n=0}^{N_0}\sum_{e_{ij}=1}^{ne} h_{e_{ij}}^{n+1}\mathrm{Vol}(L_{e_{ij}}^{n+1}) \sum_{\tau\in T_{ij}} \frac{\mathrm{Vol}(L_{e_{ij}}^{n+1}\cap C_i^n\cap\tau^n)}{\mathrm{Vol}(L_{e_{ij}}^{n+1})}\frac{\mathrm{Vol}(L_{e_{ij}}^{n+1}\cap C_j^n\cap\tau^n)}{\mathrm{Vol}(L_{e_{ij}}^{n+1})} \\ &\times |u_i^n - u_j^n| + \sum_{n=0}^{N_0-1}(\Delta t)^n \sum_{e_{ij}=1}^{ne}\mathrm{Vol}(L_{e_{ij}}^{n+1})\left|u_{e_{ij}}^{n+1} - \sum_{j\in K(i)}\frac{\mathrm{Vol}(L_{e_{ij}}^{n+1}\cap C_i^n)}{\mathrm{Vol}(L_{e_{ij}}^{n+1})}u_{e_{ij}}^n\right| \\ &+ 6\sum_{n=0}^{N_0}(\Delta t)^n \sum_{e_{ij}=1}^{ne}\left[h_{e_{ij}}^{n+1} + (\Delta t)^n\right]\sum_{\tau\in T_{ij}}\mathrm{Area}(\partial C_i^n\cap\partial C_j^n\cap\tau^n)|u_i^n - u_j^n|. \end{aligned} \tag{27}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Remark. For unsteady flows, the flow solver and unstructured dynamic mesh adaptation procedure should be coupled for the efficient capturing of the continuously varying flow physics. However, this requires very large computational resources, particularly for solving 3D flow problems. Therefore, unsteady simulation will not be presented in this paper.

4.1.2. Marking strategy. In this subsection, we introduce the maximum strategy to determine the set $\tilde{\mathcal{T}}_h$ in the general adaptive algorithm.

ALGORITHM 1 (Maximum strategy). *This algorithm determines the set* $\widetilde{\mathcal{T}_h}$.

(a) *Given: a partition* $\mathcal{T}_h$*, error estimates* $\eta_{e_{ij}}$ *for the edges* $e_{ij} \in \mathcal{T}_h$*, and a threshold* $\theta \in (0,1)$.
Sought: a subset $\widetilde{\mathcal{T}_h}$ *of marked edges that should be refined.*

(b) *Compute*

$$\eta_{\mathcal{T}_h,\max} = \max_{e_{ij}\in\mathcal{T}_h} \eta_{e_{ij}}.$$

(c) *If*

$$\eta_{e_{ij}} \geq \theta \eta_{\mathcal{T}_h,\max},$$

then mark the edge e_{ij} *for refinement and put it into the set*

$$\widetilde{\mathcal{T}_h} = \left\{ e_{ij} \in \mathcal{T}_h \mid \eta_{e_{ij}} \geq \theta \eta_{\mathcal{T}_h,\max}, \theta \in (0,1) \right\}.$$

4.1.3. Refinement strategy. The set of marked edges is examined, tetrahedron by tetrahedron, and additional edges are marked in an attempt to maintain the grid quality and to get a conforming mesh (see Figure 10). The final set of marked edges results in tetrahedra with one edge or three edges on one face, or all six edges. A tetrahedron with all six marked edges is shown in Figure 10. The mesh is then refined by inserting new nodes on the midpoints of the marked edges and reconnecting these nodes into new tetrahedra and boundary faces. For the last configuration, cutting off tetrahedra on all four corners leaves an octahedron which can be split into four tetrahedra by adding an inner edge connecting two diagonally opposite corners of the octahedron. To minimize distortion of the created tetrahedra, the shortest of the three possible inner diagonals should be chosen (see Figure 11).

4.1.4. Tetrahedral mesh improvement. Here we describe the mesh transformation operation that forms the core of our mesh improvement program.

4.1.5. Smoothing technique. The standard technique is Laplacian smoothing [21], in which a vertex is moved to the centroid of the vertices to which it is connected. Laplacian smoothing is effective for triangular meshes, but for tetrahedral meshes it is less reliable, and often produces poor tetrahedra. We use better smoothing algorithms which are based on numerical optimization [12, 42]. These algorithms define a smooth objective function that summarizes the quality of a group of elements, and use a numerical optimization algorithm such that as steepest descent or Newton's method to move a vertex to the optimal location.

4.1.6. Edge and face removal. Edge removal is a topological transformation that removes a single edge from a mesh, along with all the tetrahedra that include it. It includes 3-4 and 4-4 flips, but also includes other transformations that remove edges shared by any number of tetrahedra. In general edge removal replaces n tetrahedra

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

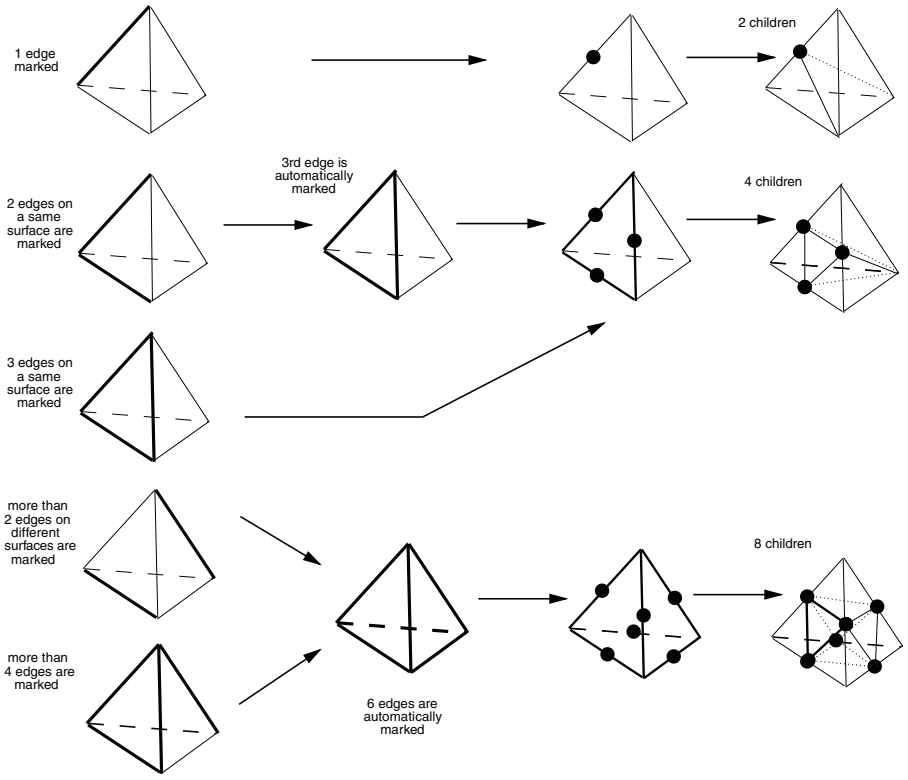

FIG. 10. *Refinement strategies for a tetrahedron.*

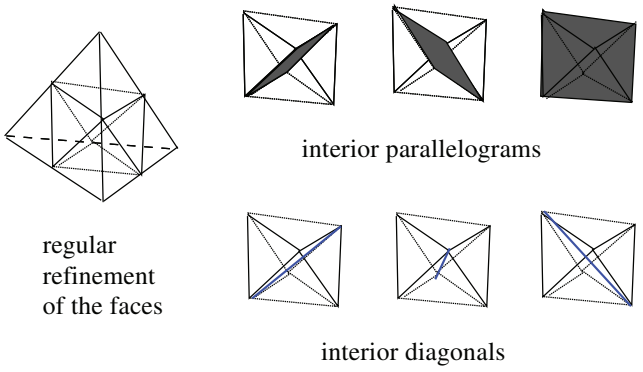

FIG. 11. *Regular refinement of the triangular faces and splitting of the remaining octahedron into four tetrahedra based on the shortest diagonal.*

with $2n - 4$. Face removal is the inverse of edge removal, and includes 2-3 and 4-4 flips. An n-face removal replaces $2n$ tetrahedra with $n + 2$.

The edge and face removal techniques effectively improve shape measures in combination with a smoothing technique. The algorithm minimizes a shape function, such as the aspect ratio, AR, for tetrahedra [33] as follows.

Let τ stand for a tetrahedron with vertices P_1, P_2, P_3, and P_4; $L_{ij} = \|P_i - P_j\|$, $1 \le i < j \le 4$, denote the length of the six edges P_iP_j of τ; S_1, S_2, S_3, and S_4 denote

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

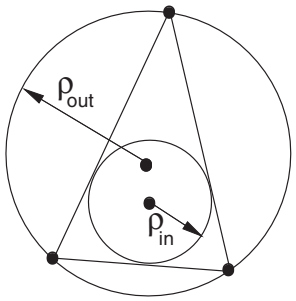

FIG. 12. *Definition of ρ_{in} and ρ_{out}.*

the area of the triangular faces $P_2P_3P_4$, $P_1P_3P_4$, $P_1P_2P_4$, and $P_1P_2P_3$, respectively, and V denote the volume of tetrahedron τ. Then we have

$$AR = \frac{1}{3}\,\frac{\rho_{\text{out}}}{\rho_{\text{in}}} \tag{28}$$

with

$$\rho_{\text{out}} = \frac{\sqrt{(a+b+c)(a+b-c)(a+c-b)(b+c-a)}}{24V},$$
$$\rho_{\text{in}} = \frac{3V}{S_1+S_2+S_3+S_4},$$

where ρ_{out} is the tetrahedral circumsphere radius, ρ_{in} is the tetrahedral in-sphere radius (see Figure 12), $a = L_{12}L_{34}$, $b = L_{13}L_{24}$, and $c = L_{14}L_{23}$ are the products of opposite edge lengths of τ. In a tetrahedron, two edges are opposed if they share no vertex. The value of the aspect ratio varies from 1, for an ideal element, to ∞, for badly shaped elements. Reconnections of tetrahedra with undesirable shape measures are investigated and new local configurations for tetrahedra are selected with better shape measures. Edges on boundary faces can also be swapped. Details of the way in which face swapping can be implemented in practice can be found in [23, 14, 17, 10].

4.1.7. Boundary modification. The inserted boundary nodes may not be located on the surface geometry of the model to be simulated because they were inserted at the midpoints of existing edges. To address this issue, we have implemented a boundary curvature correction based on Hermite interpolation [35, 48].

5. Numerical experiments.

5.1. Shock tube problem (global refinement). To illustrate the accuracy of our scheme, we present numerical results for a 3D extension of the shock tube problem introduced by Sod, where the domain of computation is $[0,1]^3$ (see Figure 13). In this problem, an initial discontinuity in the thermodynamical state of the gas breaks into a shock wave followed by a contact discontinuity and a rarefaction wave. Finite element meshes with 909 and 3609 nodes (see Figure 13) are used and the initial conditions at $t = 0$ are specified by the data

$$\begin{cases} \rho = 1, u = 0, v = 0, w = 0, p = 1.0 & \text{for } x \in [0, 1/2], \\ \rho = 0.125, u = 0, v = 0, w = 0, p = 0.1 & \text{for } x \in [1/2, 1]. \end{cases} \tag{29}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

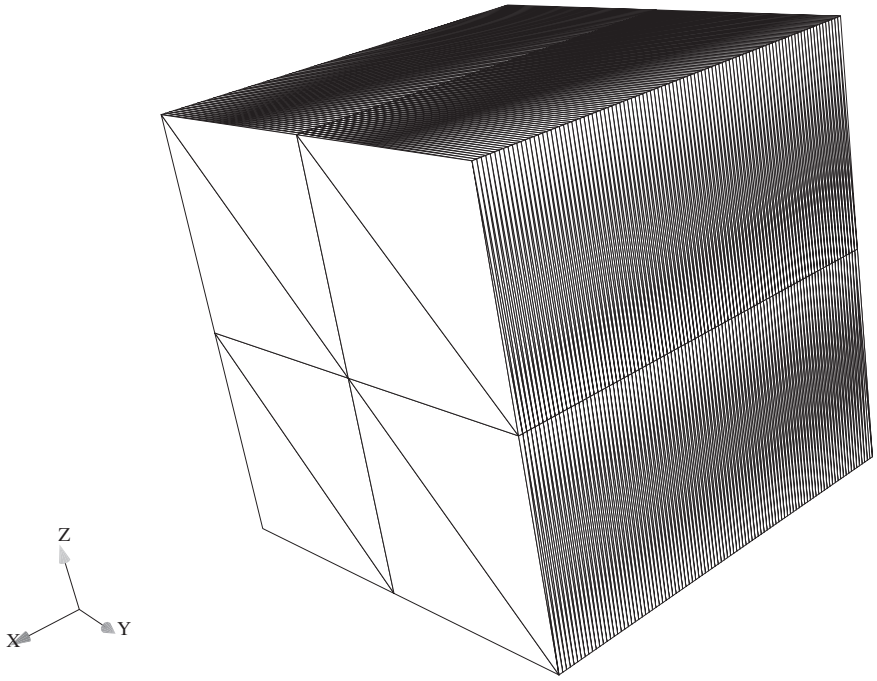

FIG. 13. *Shock tube problem, the domain of computation.*

The profiles of density ρ, x-velocity u, and pressure p are compared with the analytical solution (solid lines) at $y = z = 0.5$ and $t = 0.16$ (see Figure 14). The numerical solution of the 3D shock tube problem is clearly much more likely to suffer from excessive numerical dissipation than its 1D analogue (see e.g., [15]). In our case, this dissipation might be due to the fact that the tetrahedra are fairly stretched, with high aspect ratios and this damages the accuracy of our scheme (see Table 1 and Figure 14). With badly shaped tetrahedra (slivers) which are nearly flat and exhibit large dihedral angles (Figure 15), these problematic cells create an additional difficulty for the capture of the shock and particularly of the contact discontinuity. In [4, 6, 8] we have shown that our scheme is second-order for good shaped tetrahedra (isotropic elements) and also for 2D [27].

5.2. Transonic NACA0012 at $Mach = 0.85$ and $\alpha = 1°$ (local refinement). A NACA0012 wing configuration has been employed to demonstrate the transonic shock capturing capability of the present adaptive grid solution method. The initial grid consisted of 76,125 points and 395,203 tetrahedral cells (see Figure 16). The free stream Mach number was 0.85 and the angle of attack was set to 1.0 degree.

A fully converged steady-state solution was achieved in 6,000 iterations. The CPU time required 19,189 seconds.

An inviscid flow computation on this grid reveals the presence of a weak shock wave on the upper and lower surfaces of the wing. The Mach and C_p contours are also illustrated in Figure 16. As expected, the shock wave is diffused due to the grid coarseness and excessive numerical viscosity. Using the local remeshing procedures described earlier, after four refinement levels, the mesh has 295,201 points and 1,611,337 tetrahedra.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

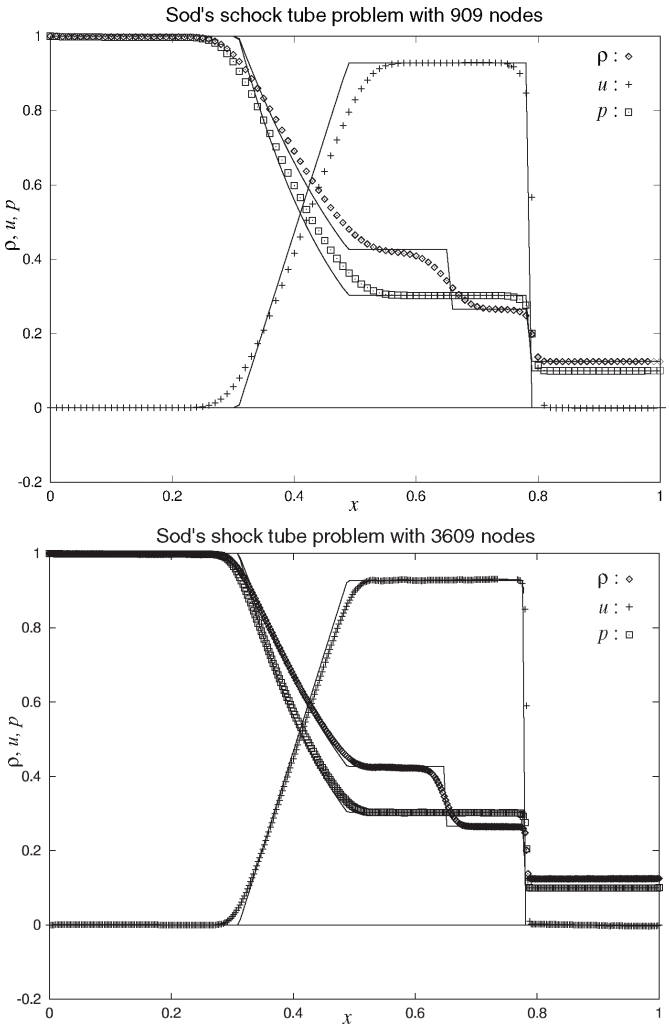

FIG. 14. *Sod's shock tube problem with* 909 *and* 3,609 *gridpoints, computed solution at* $y = z = 0.5$ *and* $t = 0.16$. *The solid lines are the exact solution.*

TABLE 1
L^2-error and accuracy order for Sod's problem.

Nodes	L^2 P	Order
909	2.778e-05	
1809	1.196e-05	1.216

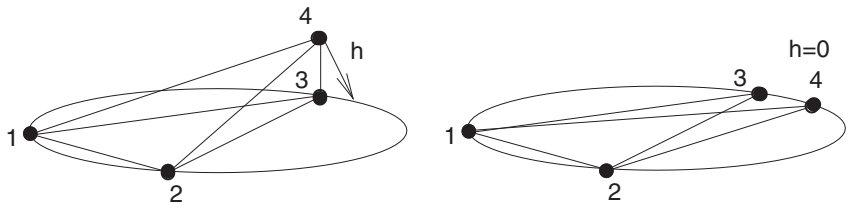

FIG. 15. *Sliver with one short edge.*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

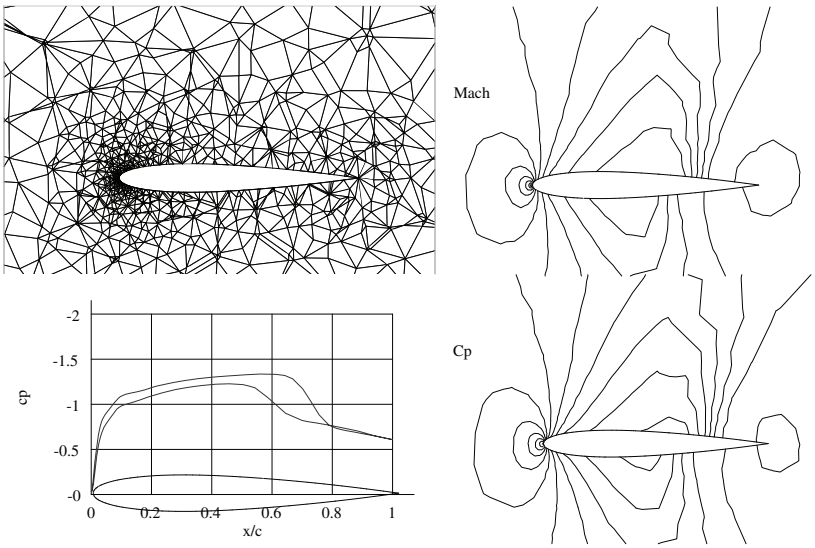

FIG. 16. *Mach and C_p contours at $M_\infty = 0.85$ and $\alpha = 1^\circ$ for the initial grid.*

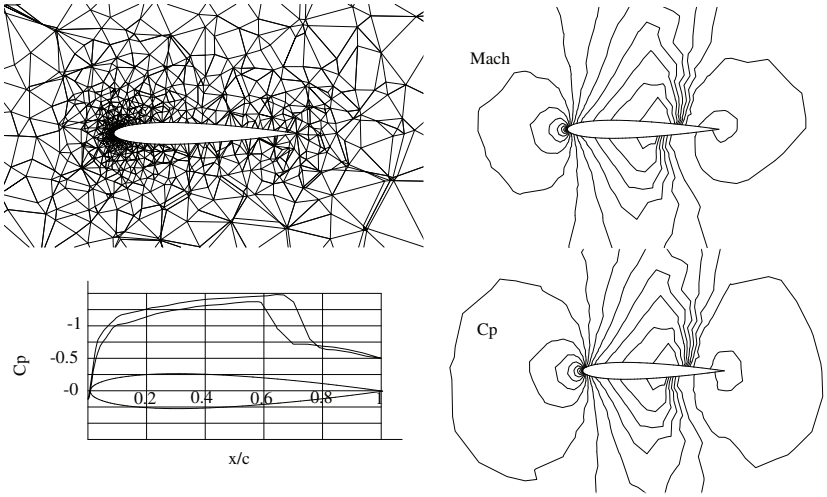

FIG. 17. *Mach and C_p contours at $M_\infty = 0.85$ and $\alpha = 1^\circ$ for the initial grid (Fluent simulation).*

A threshold value $\theta = 0.5$ is used. Figure 18 shows the adapted grid and the corresponding Mach and C_p contours. As evidenced, the grid is efficiently refined at the shock location, which shows a sharp shock definition. Figures 16 and 18 illustrate the chordwise distributions of the surface pressure coefficient C_p for the initial coarse and adapted grids. As expected, there are significant differences between the adapted and the initial grid results. From the C_p distribution, it appears that the shock location of adapted grids is well captured compared to that of coarse grids. Furthermore, this example emphasizes the advantage of grid adaptation in providing more accurate flow solutions economically. For comparison, we used the computational fluid dynamics software Fluent [16], which solves the full compressible Navier–Stokes equations. A

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

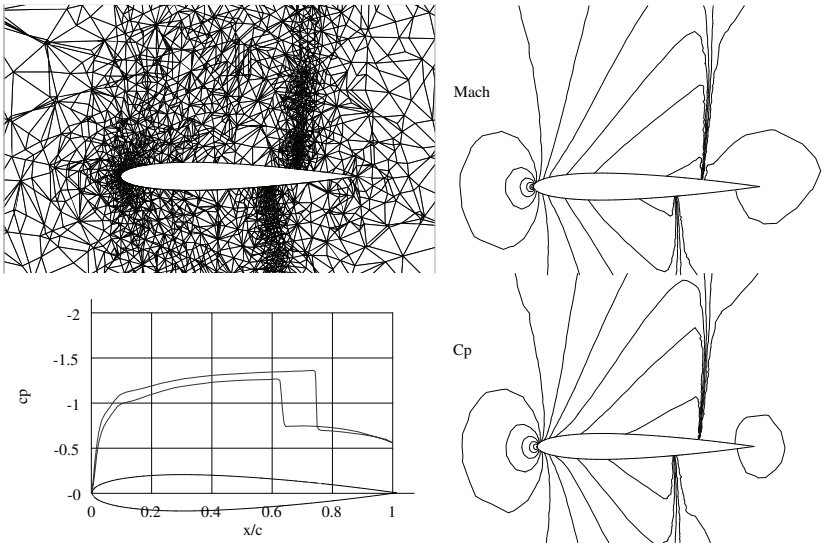

FIG. 18. *Mach and C_p contours at $M_\infty = 0.85$ and $\alpha = 1^\circ$ for the adapted grid.*

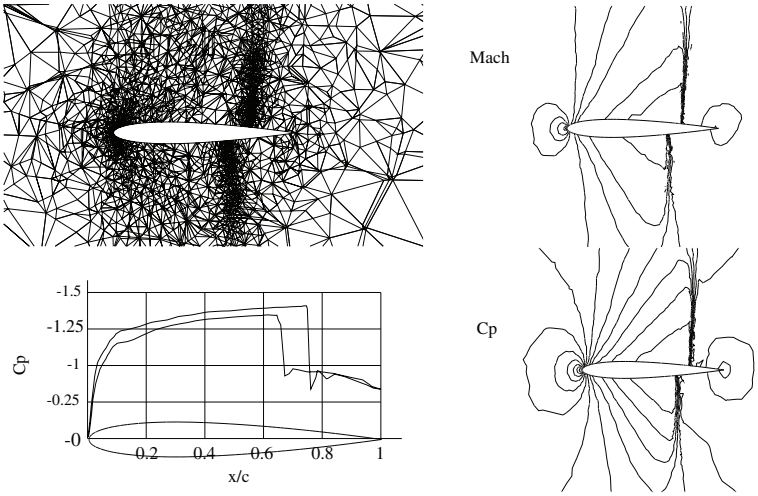

FIG. 19. *Mach and C_p contours at $M_\infty = 0.85$ and $\alpha = 1^\circ$ for the adapted grid (Fluent simulation).*

fully converged steady-state solution was achieved by means of Fluent in 37,500 iterations and the CPU time required 92,656 seconds. For this steady flow problem we compared our finite volume method with the Fluent solver using an upwind scheme which seems to be fairly competitive. We used the same meshes with both methods which gave fairly comparable results; notice that C_p and Mach contours can be nearly superposed (see Figures 16 and 17). This is an indication that both methods are indeed doing reasonable calculation. Going to the fourth adapted level, the advantage offered by our finite volume method become more obvious in Figures 18 and 19. Our method gives a nearly perfect shock resolution with very smooth contours, while the Fluent solver shows a serious breach of monotonicity (see the C_p

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

curve in Figure 19). The major difference between the two methods appears to lie in the computing times. Computing times (CPU:19,189 seconds for the proposed scheme and 92,656 seconds using Fluent) confirm the advantage of the proposed finite volume method. All calculations have been performed on a PC composed of Intel i7, 2.8 GHz processors connected with 1 Gbps fast Ethernet network cards and 6 GB of RAM at the Department of Mathematics of the University of Ottawa.

6. Conclusion. In this paper a new strategy was developed for adaptive unstructured staggered grids in 3D as an alternative to those presented in [4, 6, 7, 8], and tested on 3D cases.

Actual calculations have been restricted to the higher-order method. An error estimate for central schemes on staggered unstructured mesh adaptation has been developed for the simulation of such flows to improve the quality of the results, eliminate numerical diffusion and meet original expectations. It was shown that the present method is efficient and accurate for solving such flows. Further development includes the implementation of better error indicators for accurate adaptation of solutions involving multiple dominated flow features, solution interpolation between adaptation cycles, and extension of the method for the magnetohydrodynamics solution adaptive gridding.

Acknowledgments. The first author would like to express his appreciation to Professor Dr. Mario Ohlberger from Universität Münster for many helpful and fruitful discussions and for providing more information about *a posteriori* error estimates for central schemes. Thanks are due to the anonymous referees for many suggestions which clarified this paper. Thanks are also due to Professor Tavoularis from the University of Ottawa for providing us with the Fluent license to make some comparison with our scheme.

REFERENCES

[1] W. K. Anderson and D. L. Bonhaus, *An implicit upwind algorithm for computing turbulent flows on unstructured grids*, Comput. Fluids, 23, 1 (1994), pp. 1–21.

[2] Advisory Group for Aerospace Research and Development, *Test Cases for Inviscid Flow Field Methods*, AGARD Advisory Report No. 211, 1985.

[3] P. Arminjon, M. C. Viallon, A. Madrane, and L. Kaddouri, *Discontinuous finite elements and 2-dimensional finite volume versions of the Lax–Friedrichs and Nessyahu–Tadmor difference schemes for compressible flows on unstructured grids*, CFD Review, John Wiley & Sons, 1997, pp. 241–261.

[4] P. Arminjon, M. C. Viallon, and A. Madrane, *A finite volume extension of the Lax–Friedrichs and Nessyahu–Tadmor schemes for conservation laws on unstructured grids*, revised version with numerical applications, Int. J. of Comp. Fluid Dynam. 9, 1 (1997), pp. 1–22.

[5] P. Arminjon and M. C. Viallon, *Convergence of a finite volume extension of the Nessyahu–Tadmor scheme on unstructured grids for a two-dimensional linear hyperbolic equation*, SIAM J. Numer. Anal., 36 (1998), pp. 738–771.

[6] P. Arminjon, A. Madrane, and A. St-Cyr, *Non-oscillatory Lax–Friedrichs type central finite volume methods for 3-D flows on unstructured tetrahedral grids*, in Proc. of the Eighth Annual Conference of the CFD Society of Canada, 2000, Montréal, D. Pelletier, ed., 1 (2000), pp. 43–48.

[7] P. Arminjon, A. Madrane, and A. St-Cyr, *Numerical simulation of 3D flows with a nonoscillatory central scheme on staggered unstructured tetrahedral grids*, in H. Freistuehler, G. Warnecke, Birkhäuser, eds., in Proceedings of the Eighth International Conference on Hyperbolic Problems, Int. Series of Num. Math. 140, 2001, pp. 59–68.

[8] P. Arminjon and A. St-Cyr, *Nessyahu–Tadmor–type central finite volume methods without predictor for 3D Cartesian and unstructured tetrahedral grids*, Appl. Numer. Math., 46(2) (2003), pp. 135–155.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

[9] P. Arminjon, D. Stanescu, and M. C. Viallon, *A two dimensional finite volume extension of the Lax–Friedrichs and Nessyahu–Tadmor schemes for compressible flows*, M. Hafez and K. Oshima, eds., in Proc. of the 6th Int. Symp. on Comp. Fluid Dynamics, Vol. IV, 1995, pp. 7–14.
[10] T. J. Baker and J. C. Vassberg, *Tetrahedral mesh generation and optimization*, in Proc. 6th. International Conference on Numerical Grid Generation, ISGG, 1998, pp. 337–349.
[11] T. J. Barth and D. C. Jespersen, *The design and application of upwind schemes on unstructured meshes*, AIAA Paper No. 89-0366, 27th Aerospace Sciences Meeting, 1989, Reno, Nevada, 1989.
[12] S. A. Canann, M. Stephenson, and T. Blacker, *Optismoothing: An optimization-driven approach to mesh smoothing*, Finite Elem. Anal. Des., 13 (1993), pp. 185–190.
[13] R. Courant, K. O. Friedrichs, and H. Lewy, *Über die partiellen Differenzengleichungen der mathematischen Physik*, Math. Ann., 100 (1928), pp. 32–74.
[14] E. B. de l'Isle and P.-L. George, *Optimization of Tetrahedral Meshes*, Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations, IMA Volume in Mathematics and its Applications, 75, Springer, New York, 1995, pp. 97–127.
[15] L. Fezoui, H. Steve, and V. Selmin, *Simulation numérique d'écoulements compressibles 3-D par un schéma décentré en maillage non-structuré*, in INRIA Res. Rep. No. 0825, Institut National de Recherche en Informatique et Automatique, Roquencourt, 78153 Le Chesnay, France, 1988.
[16] http://www.fluent.com/
[17] L. A. Freitag and C. Ollivier-Gooch, *Tetrahedral mesh improvement using swapping and smoothing*, Internat. J. Numer. Methods Engrg., 40 (1997), pp. 3979–4002.
[18] E. Godlewski and P. A. Raviart, *Numerical Approximation of Hyperbolic Systems of Conservation Laws*, Springer-Verlag, 1996.
[19] B. Haasdonk, D. Kröner, and C. Rohde, *Convergence of a staggered Lax–Friedrichs scheme for nonlinear conservation laws on unstructured two-dimensional grids*, Numer. Math., 88 (2001), pp. 459–484.
[20] A. Haselbacher and J. Blazek, *On the accurate and efficient discretization of the Navier–Stokes equations on mixed grids*, AIAA J., 38 (2000), pp. 2094–2102.
[21] L. R. Hermann, *Laplacian-isoparametric grid generation scheme*, J. of the Engineering Mechanics Division of the American Society of Civil Engineers, 102 (1976), pp. 749–756.
[22] G. S. Jiang and E. Tadmor, *Nonoscillatory central schemes for multidimensional hyperbolic conservation laws*, SIAM J. Sci. Comput., 19 (1998), pp. 1892–1917.
[23] B. Joe, *Construction of three-dimensional improved quality triangulations using local transformations*, SIAM J. Sci. Comput., 16 (1995), pp. 1292–1307.
[24] Y. Kallinderis and P. Vijayan, *Adaptive refinement-coarsening scheme for three–dimensional unstructured grids*, AIAA J., 31 (1993), pp. 1440–1447.
[25] D. Kröner and M. Ohlberger, *A posteriori error estimates for upwind finite volume schemes for nonlinear conservation laws in multi dimensions*, Math. Comp., 69 (2000), pp. 25–39.
[26] M. Küther, *Error estimates for the staggered Lax–Friedrichs scheme on unstructured grids*, SIAM J. Numer. Anal., 39 (2001), pp. 1269-1301.
[27] M. Küther and M. Ohlberger, *Adaptive second-order central schemes on unstructured staggered grids*, in T.Y. Hou and E. Tadmor, eds., in Proceedings of the Ninth International Conference on Hyperbolic Problems, Caltech, Pasadena, CA, 2002, Springer, Berlin/Heidelberg/New York, 2003, pp. 675–684.
[28] A. Kurganov and E. Tadmor, *New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations*, J. Comput. Phys., 160 (2000), pp. 241–282.
[29] P. D. Lax, *Weak solutions of nonlinear hyperbolic equations and their numerical computation*, Commun. Pure Appl. Math., 7 (1954), pp. 159–193.
[30] D. Levy and E. Tadmor, *Non-oscillatory central schemes for the incompressible 2-D Euler equations*, Math. Res. Lett., 4 (1997), pp. 1–20.
[31] X. D. Liu and E. Tadmor, *Third order nonoscillatory central schemes for hyperbolic conservation laws*, Numer. Math., 79 (1998), pp. 397–425.
[32] Y. Liu, *Central schemes on overlapping cells*, J. Comput. Phys., 209 (2005), pp. 82–104.
[33] A. Liu and B. Joe, *Relationship between tetrahedron shape measures*, BIT, 34 (1994), pp. 268–287.
[34] R. Löhner, *An adaptive finite element scheme for transient problems in CFD*, Comput. Methods Appl. Mech. Engrg., 61 (1987), pp. 323–338.
[35] R. Löhner, *Regridding surface triangulations*, J. Comput. Physics, 126 (1996), pp. 1–10.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

[36] A. Madrane, *3D adaptive central schemes on unstructured staggered grids*, Hyperbolic Problems: Theory, Numerics, Applications, in Proceedings of the 11th International Conference held at Lyon, France, S. Benzoni-Gavage and D. Serre, eds., Springer, 2007, pp. 703–710.

[37] D. J. Mavriplis, *Adaptive meshing techniques for viscous flow calculations on mixed element unstructured meshes*, AIAA Paper 97-0857, 1997.

[38] J. von Neumann and R. D. Richtmyer, *A method for the numerical calculation of hydrodynamic shocks*, J. Appl. Phys., 21 (1950), pp. 232–237.

[39] S. Noelle, W. Rosenbaum, and M. Rumpf, *An adaptive staggered scheme for hyperbolic conservation laws*, Int. Ser. Numer. Math., 141 (2001), pp. 775–784.

[40] S. Noelle, W. Rosenbaum, and M. Rumpf, *3D adaptive central schemes: Part I, algorithms for assembling the dual mesh*, Appl. Numer. Math., 56 (2006), pp. 778–799.

[41] H. Nessyahu and E. Tadmor, *Non-oscillatory central differencing for hyperbolic conservation laws*, J. Comp. Phys., 87 (1990), pp. 408–463.

[42] V. N. Parthasarathy and S. Kodiyalam, *A constrained optimization approach to finite element mesh smoothing*, Finite Elem. Anal. Des., 9 (1991), pp. 309–320.

[43] R. Peyret and T.-D. Taylor, *Computational Methods for Fluid Flow*, Springer-Verlag, New York, Heidelberg, Berlin, 1983.

[44] S. Z. Pirzadeh, *An adaptive unstructured grid method by grid subdivision, local remeshing and grid movement*, AIAA Paper 99-3255, 1999.

[45] R. Touma and P. Arminjon *Central finite volume schemes with constrained transport divergence treatment for three–dimensional ideal MHD*, J. Comp. Phys., 212 (2006), pp. 617–636.

[46] B. van Leer, *Towards the ultimate conservative difference scheme V. A second-order sequel to Godunov's method*, J. Comp. Physics, 32 (1979), pp. 101–136.

[47] V. Venkatakrishnan, *Convergence to steady state solutions of the Euler equations on unstructured grids with limiters*, J. Comp. Phys., 118 (1995), pp. 120–130.

[48] N. P. Weatherill, O. Hassan, M. J. Marchant, and D.L. Marcum, *Adaptive Inviscid flow solutions for aerospace geometries on efficiently generated unstructured tetrahedral meshes*, in Proceedings of the 11th AIAA Conference, Orlando, FL, 1993, 93-3390.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.