INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 20, 1023–1037 (1995)

ADAPTIVE HYBRID PRISMATIC/TETRAHEDRAL GRIDS

Y. KALLINDERIS

Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin TX 78712, U.S.A.

SUMMARY

Employment of hybrid prismatic/tetrahedral grids is a relatively new approach for viscous flow computations with 3D complex geometries. The body surface is covered with triangles, which provides geometric flexibility, while the structure of the mesh in the direction normal to the surface provides thin prismatic elements suitable for the viscous region. The outermost layer of the prismatic grid is then used as the inner boundary surface for a tetrahedral grid which covers the rest of the computational domain. The hybrid grid is employed for performing Navier–Stokes calculations in the prismatic region near the body and Euler calculations in the tetrahedral regions away from the body.

The generation method of hybrid prismatic/tetrahedral grids marches a triangulated surface mesh away from the body to form a semi-unstructured prismatic grid. Tetrahedra are then constructed which are linked directly to the nodes of the outermost prismatic layer. The hybrid grid is locally refined over both the prismatic and tetrahedral regions. Directional embedding of the prisms preserves the structure of the mesh in the *normal-to-surface* direction.

KEY WORDS adaptive; hybrid grids; 3D

1. INTRODUCTION

Simulation of flows around three-dimensional bodies is a major issue in computational fluid mechanics. Complexity of the geometry and flow fields makes 3D computations a pacing item. Generation of a body-conforming grid proves to be a difficult task.[1,2] In cases of structured meshes a large number of separate blocks must be defined and hexahedral grids are generated within each block. In addition, a special algorithm is required in order to match the grids of neighbouring blocks. A radical alternative to a structured mesh is the use of tetrahedra. Tetrahedral grids provide flexibility in 3D grid generation since they can cover complicated topologies more easily compared with hexahedral meshes.[2–4] However, generation of tetrahedral cells for boundary layers is quite difficult. In those regions the main solution gradients usually occur in the direction normal to the surface, which requires cells of very large aspect ratio. It appears that structured grids are superior in capturing the directionality of the flow field in viscous regions. Furthermore, unstructured grids require a great deal more memory than their structured counterparts. While structured grid elements may be referenced by a simple (i, j, k) index, unstructured elements require pointers to provide information on connectivity between cells, faces, edges and nodes. Additionally, approximately five times more tetrahedra than hexahedra are required to fill a given region with a fixed number of nodes. This vast increase in the number of required cells leads directly to impractical memory requirements for 3D viscous flow computations.

CCC 0271–2091/95/091023–15

A compromise between the two different types of grids is provided by prismatic meshes.[5,6] Such grids consist of triangular faces that cover the body surface, while quadrilateral faces extend in the direction normal to the surface. The cells can have very high aspect ratio while providing geometric flexibility in the lateral directions. The memory requirement of the prismatic grid is much smaller than that of a tetrahedral grid. The prismatic grid requires a set of pointers to define a 2D triangular mesh combined with a single index for each element of the prism.[7] Since each stack of prisms is formed from the body triangular face, no additional data are required. A prism may be referenced by its corresponding triangular face on the boundary and an index indicating the position of the prism within the stack of prisms originating from the same boundary face. Additionally, the structured nature of the prismatic grid allows simple implementation of algebraic turbulence models[8] and semi-implicit numerical schemes.

The areas between different prismatic layers covering the surfaces of the domain can be quite irregular. Prismatic elements cannot cover domains that are multiply connected. Furthermore, the relevant flow features do not usually exhibit the strong directionality that the viscous stresses have. Tetrahedral elements appear to be appropriate for these regions. Their triangular faces can match the corresponding triangular faces of the prisms.[9]

Adaptive grid methods have evolved as an efficient tool to obtain numerical solutions without *a priori* knowledge of the nature and resolution of the grid necessary to efficiently capture the flow features. These algorithms detect the regions that have prominent flow *features* and increase the grid resolution locally in such areas. Significant progress has been made in the implementation of adaptive schemes for tetrahedral grids.[10,11] On the other hand, there have not been any such algorithms developed for prisms.

In the present work the hybrid grid is locally refined over both the prismatic and tetrahedral regions. A special type of adaptive refinement of prisms is applied in order to preserve the structure of the mesh along the *normal-to-surface* direction. The detected triangular faces on the surface are divided. Then all prisms above these faces are directionally divided along the lateral directions. The cells are not divided along the third direction which is normal to the surface. In this way grid interfaces within the prismatic region are avoided and the structure of the grid along the *normal-to-surface* direction is preserved.

2. GENERATION OF HYBRID GRID

2.1. Prismatic grid

An unstructured triangular grid is employed as the starting surface to generate a prismatic grid. This grid, covering the body surface, is marched away from the body in distinct steps, resulting in generation of structured prismatic layers in the marching direction (Figure 1). The process can be visualized as a gradual inflation of the body's volume. There are two main stages in the algebraic grid generation process: (i) determination of the directions along which the nodes will march (marching vectors) and (ii) determination of the distance by which the nodes will march along the marching vectors. Smoothness of the generated grid is attained by applying Laplacian-type smoothing steps.

Each node on the marching surface is advanced along a marching vector. If strict orthogonality is imposed, the marching vector is the surface normal vector at the node. However, the resulting grid may overlap in concave regions. Furthermore, the grid lines in the normal direction will have kinks from one layer of prisms to the next. The goal of the marching scheme is to reduce the curvature of the previous marching surface at each step while ensuring smooth grid spacing.

The group of faces sharing a common node will be termed a *manifold*. Each manifold on the marching surface has a unique configuration, which may have singular topology. The initial step is

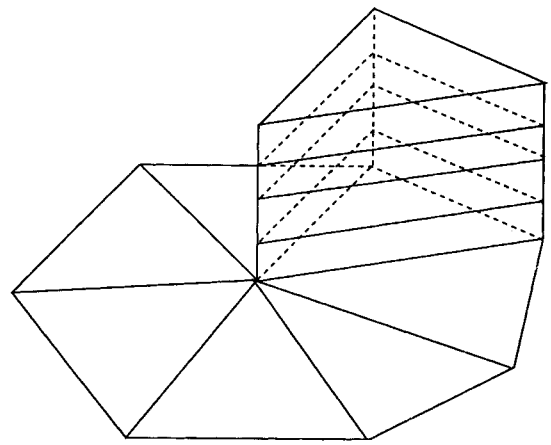

Figure 1. Semi-unstructured prismatic grid elements

calculation of the vectors that are normal to the surface of the manifold at the nodes ($\hat{N}_i$). A special technique determines those vectors.[5] The initial marching vectors $\hat{m}_i$ at each node i coincide with the normal vectors $\hat{N}_i$. However, this may not provide a valid grid, since overlapping may occur, especially in regions of the grid with closely spaced nodes. To prevent overlapping, the marching vectors must be altered to increase the divergence of all marching vectors at the nodes k that are connected to node i of each manifold. Weighted Laplacian smoothing is applied to the direction of the marching vectors in order to avoid overlapping of the marching lines of the prisms, as follows:

$$\hat{m}_i = \omega \hat{m}_i + (1 - \omega) \sum \hat{m}_k,$$

where $\hat{m}_i$ is the marching vector of node i and $\hat{m}_k$ are the marching vectors of the surrounding nodes k that belong to the manifold of node i. The weighting factor ω is a linear function of the manifold characteristic angle β_{ave}. This angle is the average dot product between the pairs of faces forming the manifold. It has small values in concave regions and relatively large ones in convex areas.

Determination of the marching distances is based on the characteristic angle β_{ave} of the manifold of each node to be marched. The marching distance is a linear function of β_{ave}. It yields relatively large marching distances in the concave regions and small distances in the convex areas of the marching surface. Specifically, the distance Δn is

$$\Delta n = (1 + \alpha)\Delta n_{\text{ave}},$$

where Δn_{ave} is the average edge length of the edges that form the node manifold and α is a linear function of the manifold angle β_{ave}. Concave regions have positive characteristic angle, while convex areas have negative angle. The sign of α is the same as the sign of the angle.

The prisms generated by marching the surface form a skeleton mesh of valid prisms within which any number of new cells may be generated. A relatively small number of layers of prisms can be generated occupying a large volume. Additional layers can be generated by subdividing the original prisms along the normal direction. The computation time required for generation of those additional layers is negligible compared with that for creation of the original layers via the marching method.

Two-dimensional data structure. Three-dimensional Navier–Stokes computations usually require a great amount of memory. In the present work the structure of the prismatic grid in one of the directions is exploited in order to reduce storage to the amount required for a two-dimensional Navier–Stokes

solver with triangles. All pointers that are employed refer to the triangular faces of the first prisms on the body surface (*base* faces), with all prisms above the same *base* face forming a *station*. A simple index (typical of structured grid indexing) is sufficient to refer to any prismatic cell belonging to the same station. The *base* face pointers connect the faces to their corresponding edges and nodes.

Application to aircraft geometry. An F-16A aircraft geometry was chosen as a case for the developed grid generator, since the complexity and singularities of the surface are a severe test for the method. The main features of the configuration are the forebody, canopy, leading edge strake, wing, shelf regions and inlet. The surface triangulation consists of 2408 triangular faces and 1259 points. Two parts of the generated prismatic grid require examination in terms of quality. The first is the grid formed by the triangular faces of the prisms (unstructured part), while the second is the grid formed by the quadrilateral faces (structured part).

A view of the initial and grown surfaces is shown in Figure 2. The growth of the grid is illustrated after 15 marching steps. The effect of the marching process is similar to inflation of the original body volume. It is observed that the distribution of points on the grown surface is quite smooth. The highly singular regions at the aircraft nose, the wing leading and trailing edges, the wing tip, the canopy and the inlet have been smoothed out on the grown surface. Furthermore, the grid spacing on the grown triangular surface is smoother compared with the initial triangulated body surface. The singular concave regions at the junctions between the wing and the fuselage and between the engine inlet and the body have been 'filled in' and the grid is more uniform over those regions compared with the initial grid on the body.

A view of the structured part of the prismatic grid is shown in Figure 3. It should be noted that the depicted surface is not planar. The grid spacing on the quadrilateral surface is quite uniform. Furthermore, the marching lines emanating from the aircraft surface are quite smooth, including the ones that correspond to singular points.

The final number of prismatic cells is 96,320, while the number of grid nodes is 50,360. The required computing time was 444 on a Sun workstation (2 Mflops speed), which implies generation of 217 prisms per second.

2.2. *Generation of tetrahedral grid*

One approach for generating the tetrahedra covering the outer inviscid region of the computational domain is via a *divide-and-conquer* type of method. In this process an initial hexahedron is subdivided into eight smaller hexahedra called octants.[9,12] Any octant which contains a portion of the outermost prismatic surface is a *boundary octant* and is subdivided further (inward refinement). The hexahedral

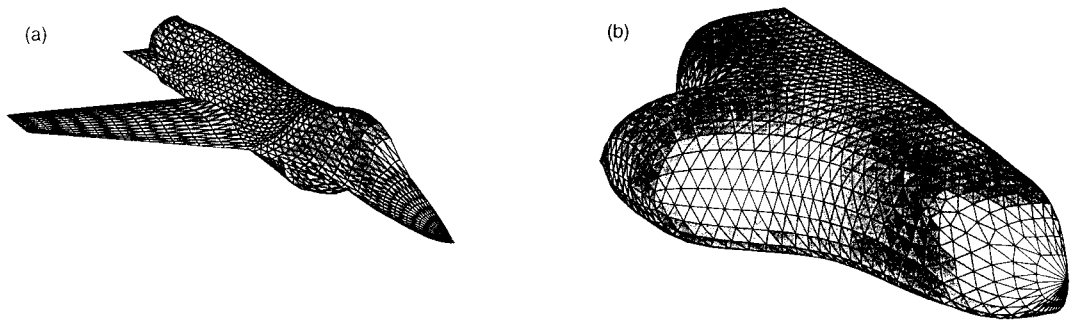

Figure 2. Foreview of (a) initial and (b) outermost prismatic surface for aircraft geometry

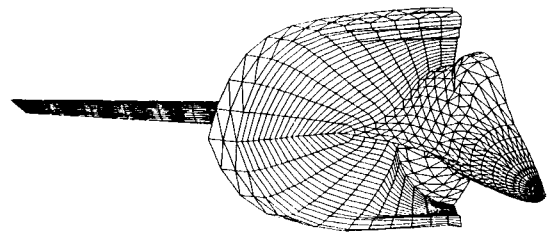

Figure 3. Cut through prismatic grid showing structured part in area of wing strake/fuselage junction

grid is further refined in a *balancing* process to prevent neighbouring octants whose degree of refinement differs by more than one (outward refinement). Finally, all hexahedra are divided into tetrahedra to produce the outer grid. Figure 4 shows the octants around the outermost prismatic layer of the aircraft. The final tetrahedral grid is shown in Figure 5 and consists of 168,568 tetrahedra and 28,302 nodes. The required computing time for the tetrahedral grid was 153 on the Sun workstation of 2 Mflops speed.[9]

3. PRISMATIC MESH ADAPTATION

3.1. Directional division of prisms

A special type of adaptive refinement of prisms is applied in the present work in order to preserve the structure of the mesh along the *normal-to-surface* direction. The detected triangular faces on the surface are divided as shown in Figure 6. Then all prisms above these faces are directionally divided along the lateral directions. The cells are not divided along the third direction which is normal to the surface. In this way grid interfaces within the prismatic region are avoided and the structure of the grid along the *normal-to-surface* direction is preserved. Furthermore, such a division is not needed, since

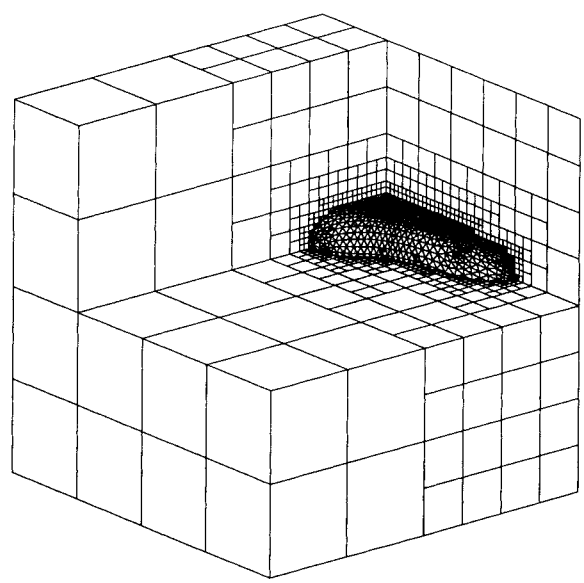

Figure 4. 3D section of domain showing outermost prismatic surface and octree hexahedra for aircraft geometry

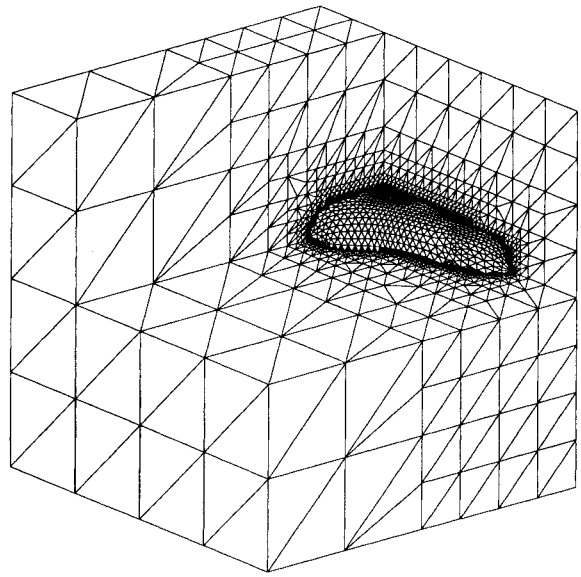

Figure 5. 3D section of domain showing outermost prismatic surface and final tetrahedral grid for aircraft geometry

the points can be redistributed along that direction in such a way that the viscous stresses are resolved.[5] Therefore adaptation of the prisms reduces to adaptation of the triangular grid on the surface. This results in a simpler and less expensive algorithm in terms of storage and CPU time compared with a 3D adaptation algorithm.

Two types of division are applied. The first divides the triangular faces of the prisms into four smaller triangles, while the second divides them into two, as shown in Figure 6. In the first case the

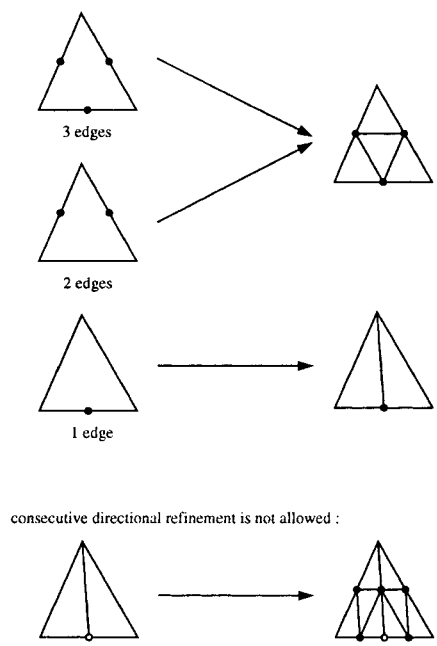

Figure 6. Division rules for base triangles of prismatic grid

parent triangle is divided into four *children*, while it is divided into two *children* in the second case. If two edges of the triangle are to be refined, the third is also refined automatically to avoid stretched elements. Division of cells is also employed to divide *transition* cells at the interface between different embedded regions that contain *hanging* nodes in the middle of some of their edges owing to refinement of neighbouring cells. Furthermore, the position of newly created surface nodes is corrected so that the original geometry of the surface is kept. In addition, coarsening of the adapted prismatic grid is applied over regions where the embedded cells are no longer needed.

In order to avoid stretched meshes being created owing to several refinements of the same cells, the following rules are applied: (i) only one level of refinement/coarsening is allowed during each adaptation; (ii) if the *parent* cell is refined according to the one-edge type of division, then it is divided according to the three-edge type of division at the next refinement (Figure 6); (iii) if the maximum adaptation level difference of neighbouring surface triangles around a node is more than one, the coarsest triangles will be refined according to the three-edge type of division. Grid coarsening is conducted in the same manner. If the refinement and coarsening occur in the same triangle (different edges), then refinement is applied only. Refinement always has a higher priority than coarsening.

Also adaptation may yield embedded regions that are slightly smaller than the features which are detected, which results in interfaces being located within or very close to the regions of relatively large gradients. In order to avoid such situations, the algorithm places extra rows (typically two) of embedded cells surrounding the detected regions.

To satisfy all the above rules for grid smoothing, some *iterations* are required; typically the number of iterations is less than five.

3.2. *Redistribution of grid nodes*

Flexibility in specifying grid spacing along the marching lines is crucial for accuracy of Navier–Stokes computations. A scheme is employed which redistributes the nodes along each of the marching lines emanating from points on the body surface; in other words, the marching directions are maintained but the marching distances are modified. This is accomplished by performing a cubic spline fit to each of the marching lines using the prismatic node locations for the spline knots. The nodes are then redistributed along the splined lines. The distribution is such that the new node positions satisfy certain grid-spacing requirements. In the present work the spacing of the first point off the body surface is specified along with a constant stretching factor ω. Then the positions of the points along the marching lines (n) are given by the formula

$$n_{j+1} = n_j + \omega(n_j - n_{j-1}); \tag{1}$$

a typical value of ω is 1·1.

Adaptive redistribution of the grid points can also be employed with the present technique. The evolving flow solution can be monitored and the nodes can be moved along the marching lines so that the boundary layer is resolved sufficiently. For example, the points can be moved so that the y^+-values at the wall are smaller than a specified number.[13]

An example of redistribution of a prismatic grid around the ONERA M6 wing is illustrated in Figure 7. The figure shows the initial grid before redistribution as well as the final grid that is obtained after moving the points along the marching lines. The figure includes part of the wing triangulated surface (unstructured) as well as the surface of the quadrilateral faces (structured). The spacing of the initial grid along the marching lines is quite large close to the wing, which is unacceptable for Navier–Stokes computations. The final redistributed grid has a specified minimum spacing at the wall equal to 0·005 and constant stretching factor ω = 1·1.

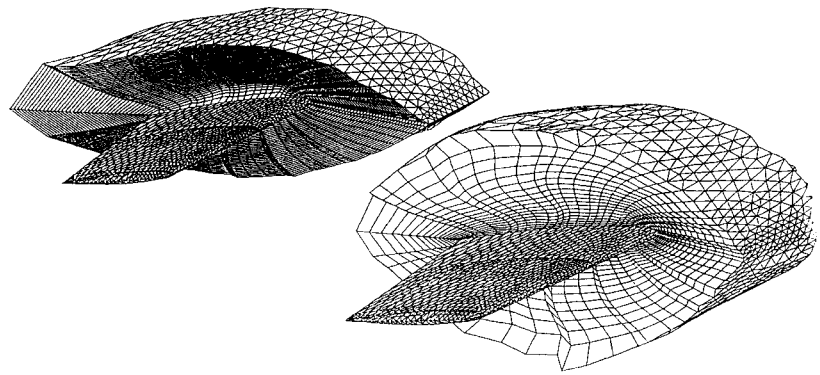

Figure 7. Effect of grid redistribution along marching lines of prismatic grid (initial and redistributed grid around ONERA M6 wing)

4. TETRAHEDRAL GRID ADAPTATION

Adaptive embedding introduces finer cells in those regions of the field that the algorithm senses need to be resolved, while simultaneously removing cells from previously embedded regions that do not require extra resolution. Three types of tetrahedral cell division are employed[10] as shown in Figure 8: (i) one edge is refined; (ii) three edges on the same face are refined; (iii) all six edges are refined.

After all edges are flagged, each tetrahedral cell is visited and the flagged edges are counted. Then the cell is flagged for division according to the above three types. In the cases that are different from

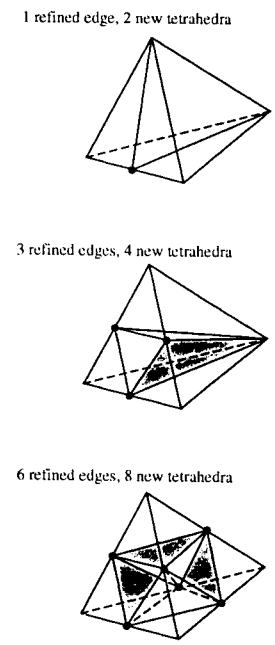

Figure 8. Cell division rules for tetrahedra of hybrid mesh

the three cases above, the cell is divided according to the third type of division. If two edges on the same face are refined, the third edge of that face is refined according to the second type of division.

An important function of the developed grid adaptation algorithm has been the deletion of previously divided tetrahedra. Flow fields with evolving flow features are very common and an adaptive scheme which eliminates resolution at a region in which additional nodes are not needed any more is essential. In principle, the unrefinement process is the inverse process of the refinement scheme. The mid-edge nodes on the 'parent' cell edges are removed, resulting in deletion of the corresponding child edges, faces and cells. Cells that have been flagged to be deleted are eliminated along with their sibling cells which were formed from the same parent cell. In this way the parent cells are recovered after the deletion of their child cells.

The adaptation processes for prisms and tetrahedra are coupled through the outermost triangular surfaces of the prismatic grids, which coincide with tetrahedral triangular faces. The pairs of *interface* cells (between prisms and tetrahedra) are divided if one or both cells are flagged for division. In this way additional mid-edge nodes are avoided.

4.1. Adaptation procedure

The criterion for division of a cell was based on monitoring the values of the velocity differences and gradients across the cell edges (*detection parameters*). A threshold value for the detection parameters is employed which is set by using the mean and standard deviation of the distribution of each parameter. More specifically, $\Phi_{\text{thre.ref.}} = \Phi_{\text{ave}} + \alpha \cdot \Phi_{\text{sd}}$, where Φ_{ave} and Φ_{sd} are the average and standard deviation values of the detection parameter Φ respectively. If the detection parameter is above the threshold for refinement, the edge is flagged for division. Similarly, if the detection parameter is lower than $\Phi_{\text{thre-unref.}} = \Phi_{\text{ave}} - \alpha \cdot \Phi_{\text{sd}}$, then the edge is flagged for deletion. Here α is an empirically chosen value, typically 0·4. This method follows previous work in References 13 and 14. An upper bound on the number of grid cells is imposed owing to computer memory limitations. If this bound is exceeded, then the algorithm increases the value of α. An additional layer of cells is added surrounding the region to be embedded in order to ensure that the flow feature lies within the refined region. In order to avoid increase in skewness of the interface cells, those cells are not flagged for further refinement. This results in a gradual decrease in the size of the cells as the finer embedded grids 'focus' more on the regions with the highest flow gradients (e.g. shock waves).

The unrefinement process starts at the highest embedding level (finest cells) and henceforth proceeds by visiting the cells marked for deletion in decreasing order of their embedding levels. Cells flagged for unrefinement with siblings flagged for refinement are not deleted. If at least one sibling is flagged for coarsening while the rest are not flagged, then all of them are flagged for deletion. An embedding level difference higher than one is not allowed between two cells that share a common face or edge in order to avoid multiple hanging nodes at the edges. If this is violated, then the cell that is flagged for deletion is not removed. It should be emphasized that the above constraint is not restrictive to the extent of hindering the adaptive algorithm from resolving the prominent flow features.

4.2. Application to transport aircraft

Computations were carried out for a low-wing transport (LWT) configuration. Inviscid, transonic flow of $M_\infty = 0{\cdot}768$ was considered with an angle of attack $\alpha = 1{\cdot}116°$. A semispan computational grid was generated for the LWT aircraft geometry without the nacelles. The initial grid is comprised of 48,828 nodes and 266,400 tetrahedra.

All the computations were performed on a CRAY Y-MP. The code was vectorized to run at a speed of about 100 Mflops. The memory required for the solver was 30 words per node.

The computed solution obtained on the initial grid is shown in Figure 9. Mach number contour lines are plotted at intervals of $\Delta M = 0{\cdot}02$. A single shock wave is formed on the upper surface of the wing. The embedded grid on the aircraft surface is shown in Figure 10. The adapted grid has 185,230 nodes. The Mach number contour lines are shown in Figure 11, plotted at intervals of $\Delta M = 0{\cdot}02$. The plot shows that the shock is distinctly sharper on the adapted grid solution. Detailed accuracy evaluations can be found in References 10 and 15.

5. IMPLEMENTATION OF ADAPTIVE HYBRID GRIDS

A number of test cases are employed in order to provide an assessment of accuracy and robustness of a finite volume Navier–Stokes method with adaptive hybrid meshes. The numerical integration scheme is of central spatial differencing type. The solution is marched in time using a Taylor series expansion following the Lax–Wendroff approach. The cases include flow over a cylindrical bump in a channel and flow around a spherical body.

5.1. Cylindrical bump in a channel

Supersonic laminar flow of Mach number $M_\infty = 1{\cdot}4$ and Reynolds number $Re = 16{,}000$ is computed. This has been a standard test case for two-dimensional Navier–Stokes schemes. The same case of a cylindrical bump in 3D is considered here. The flow is quasi-two-dimensional, which permits direct comparison with existing 2D numerical results.

Two different types of grids are employed and compared in terms of accuracy: (i) a hybrid grid with prisms covering the boundary layer and tetrahedra used elsewhere; (ii) a globally refined hybrid grid that contains no grid interfaces.

The case of adapting the hybrid (prismatic/tetrahedral) mesh is considered. The initial hybrid grid is generated by subdividing a $33 \times 5 \times 33$ hexahedral mesh. The prismatic region consists of 22 layers of prisms and has thickness equal to 0·264 channel heights. Figure 12 illustrates (a) the initial coarse grid and (b) the one-level adapted hybrid grids in the XZ-plane at midspan of the channel. The prisms cover the boundary layer and are embedded directionally, while the tetrahedra cover the rest of the domain and are embedded in the shock regions. Figure 13 shows the flow field in the middle of the span of the channel in terms of Mach number contours. The + symbols indicate the interface between the prismatic and tetrahedral regions. It is observed that there are no kinks at the interface. Figure 14 shows a comparison of wall friction coefficient distributions between the locally refined hybrid grid

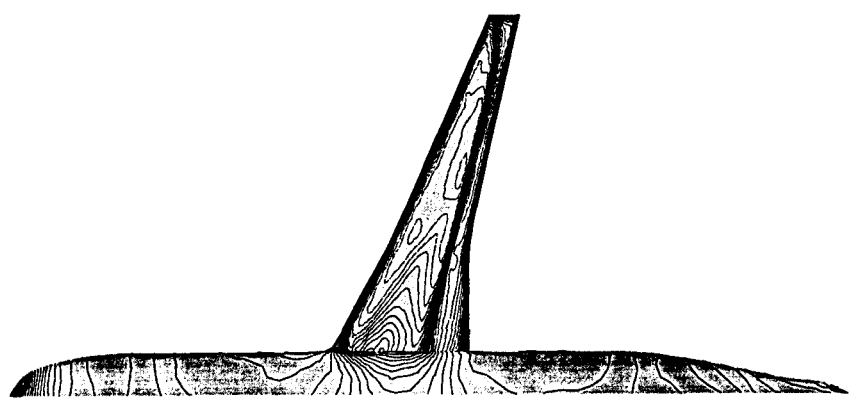

Figure 9. Mach number contour lines on LWT aircraft upper surface—solution obtained on initial grid ($\Delta M = 0{\cdot}02$)

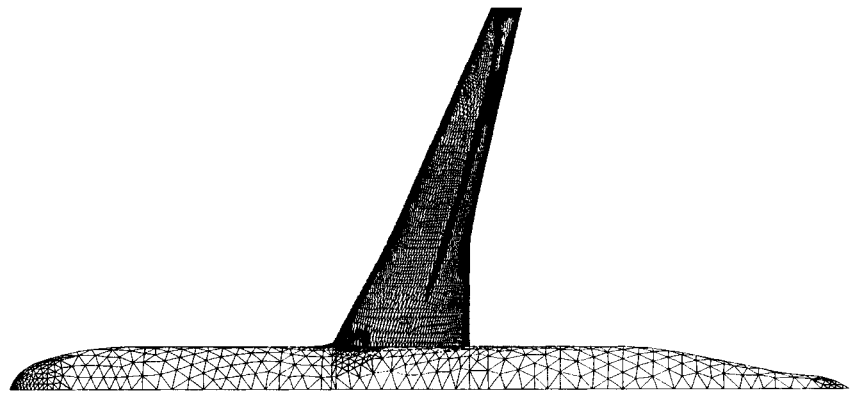

Figure 10. Triangulation on wing and body surface corresponding to once-adapted grid

and the corresponding globally refined one. The results are in close agreement. It should be noted that a stable solution was obtained even though the mesh of Figure 12 is quite non-uniform. The adapted grid requires 50% of the CPU time required by the globally fine grid in this case. The one-level adapted grid has 13,273 *nodes*, 14,960 *prisms* and 22,503 *tetrahedra*, while the globally refined grid has 19,305 *nodes*, 22,508 *prisms* and 30,720 *tetrahedra*.

5.2. Flow around spherical body

Subsonic flow around a hemisphere is considered. The Mach number is $M_\infty = 0{\cdot}5$ and the Reynolds number $Re_\infty = 10^3$. An *all-prismatic* grid and a hybrid grid are generated from a hexahedral spherical mesh consisting of 36 nodes along the peripheral direction, 19 nodes in the azimuthal direction and 31 nodes in the radial direction. Both meshes consist of exactly the same nodes, which permits direct comparison between solutions obtained on them. The far-field boundary is placed at a distance of 15 radii away from the surface. The plane passing through the maximum diameter of the hemisphere was considered to be a symmetry boundary, while a no-slip wall condition is applied on the body surface.

Figure 15 shows the one-level adapted hybrid grid. It is observed that both the boundary layer and the wake regions are refined where appreciable velocity gradients occur. The corresponding *all-prismatic* mesh that contains no interfaces is refined globally once. Figure 16 shows a comparison of

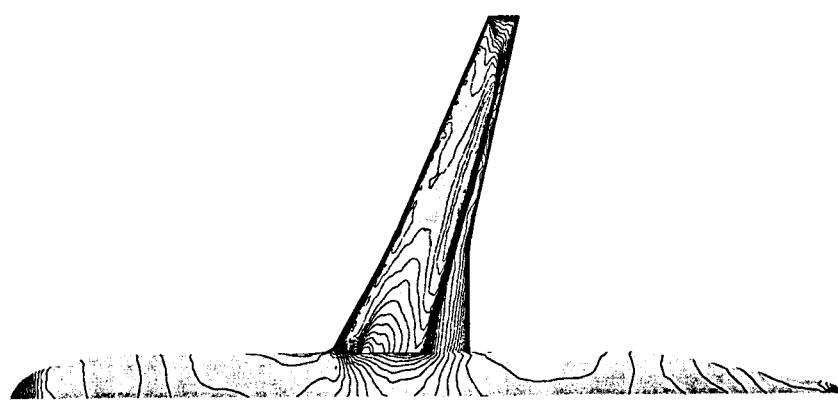

Figure 11. Mach number contour lines on LWT aircraft upper surface—solution obtained on once-adapted grid ($\Delta M = 0{\cdot}02$)

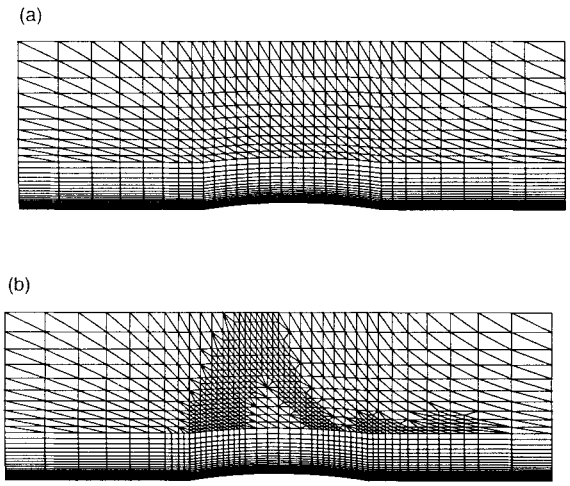

Figure 12. Hybrid grids for supersonic flow over 4% cylindrical bump (*XZ*-plane at midspan): (a) initial coarse prismatic/tetrahedral grid; (b) one-level adapted grid over both prismatic and tetrahedral regions

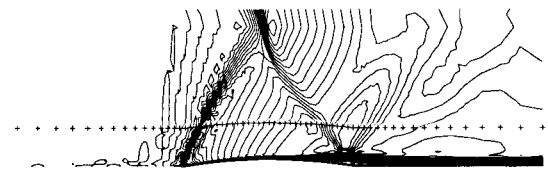

Figure 13. Supersonic flow over 4% cylindrical bump, M_{∞} = 1·4, Re_{∞} = 16,000. One-level adapted hybrid grid (prism layer thickness 0·264 channel heights). Mach number contours with min = 0, max = 1·65 and inc = 0·025 (+ symbols indicate prism/tetrahedron interface)

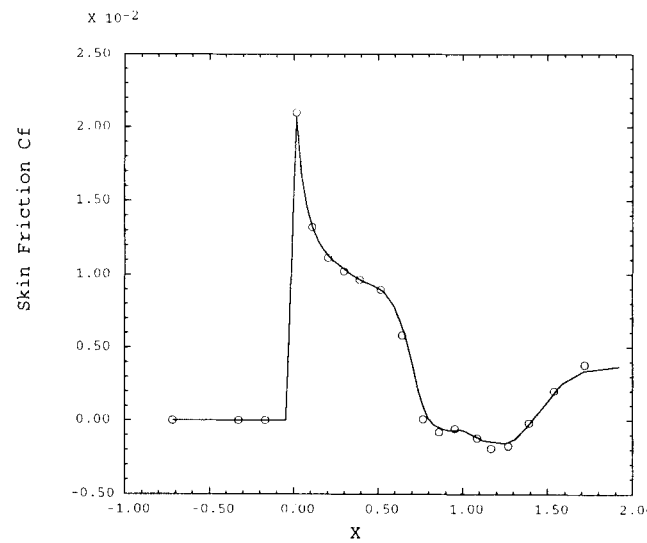

Figure 14. Supersonic flow over 4% cylindrical bump, M_{∞} = 1·4, Re_{∞} = 16,000. One-level adapted hybrid grid (prism layer thickness 0·264 channel heights). Comparison of skin friction distributions at wall: —, adapted; ○, globally refined hybrid grid

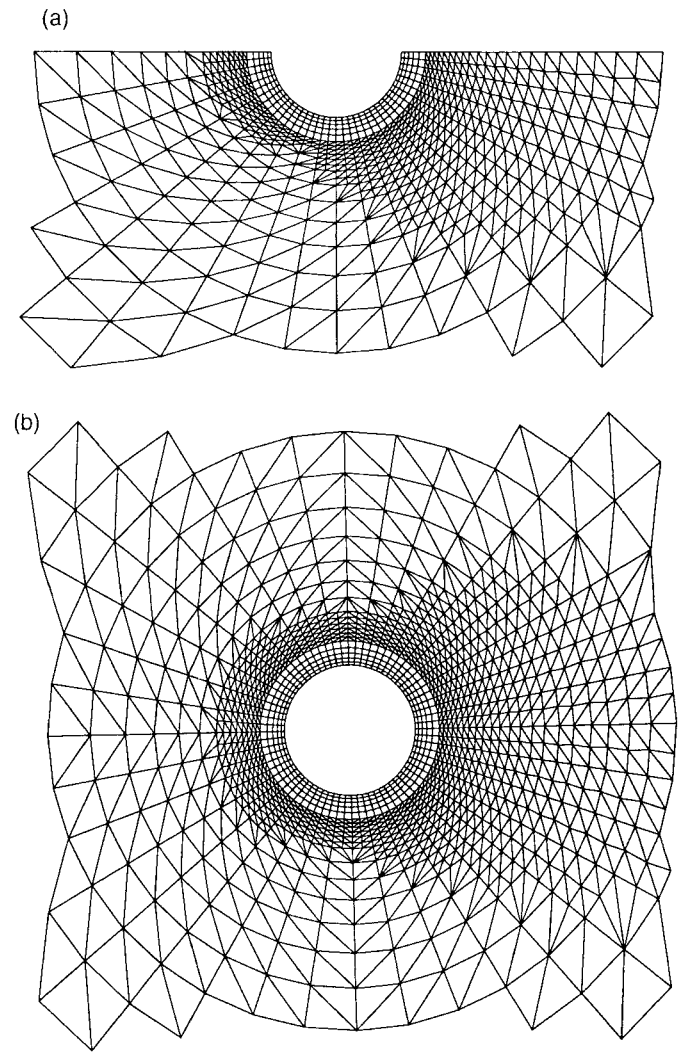

Figure 15. One-level adapted hybrid grid for subsonic flow over spherical body, $M_\infty = 0{\cdot}5$, $Re_\infty = 10^3$: (a) equatorial (XY-) plane; (b) symmetry (XZ-) plane

wall friction coefficient distributions between the globally refined prismatic grid and the hybrid grid at the junction between the body surface and the symmetry plane. The agreement between the results obtained with the globally fine *all-prismatic* grid and the adapted hybrid grid is good.

6. CONCLUDING REMARKS

Combination of semi-unstructured prismatic and unstructured tetrahedral elements appears to be suitable for 3D viscous flow simulations around complex geometries. Generation of prisms employs an algebraic marching method which requires a computation time of the order of minutes on a low-end workstation.

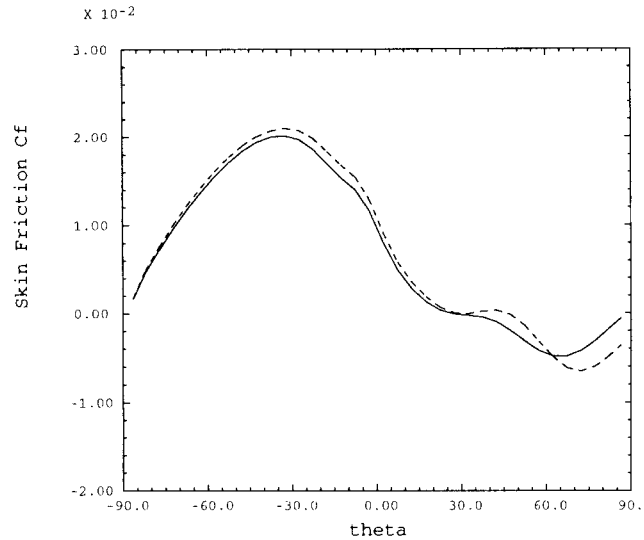

Figure 16. Subsonic flow over spherical body, $M_\infty = 0{\cdot}5$, $Re_\infty = 10^3$. Comparison of skin friction distributions at junction between wall surface and symmetry plane: —, globally refined prismatic grid; – –, one-level adapted hybrid grid

Combination of directional refinement/coarsening of the prisms along the lateral directions and redistribution of the nodes along the marching lines of the prismatic mesh yields increased accuracy with reduced computation costs.

Initial test cases of a typical finite volume Navier–Stokes solver with adaptive hybrid (prismatic/tetrahedral) meshes evaluated the accuracy and robustness of the method. Viscous flow simulations around aircraft configurations with hybrid grids are currently being pursued.

ACKNOWLEDGEMENTS

This work was supported by NASA Grant NAG1-1459, DARPA Grant DABT 63-92-0042, Texas Advanced Technology Program (ATP) Grant 003658-413, and NSF Grant ASC-9357677 (NYI program). Supercomputing time was provided by the NAS Division of the NASA Ames Research Center as well as by the NSF Pittsburgh Supercomputing Center.

REFERENCES

1. P. R. Eiseman and G. Erlebacher, 'Grid generation for the solution of partial differential equations', *ICASE Report 87-57*, 1987.
2. T. J. Baker, 'Developments and trends in three dimensional mesh generation', *Appl. Numer. Math.*, **5**, 275–304 (1989).
3. R. Löhner and P. Parikh, 'Generation of three-dimensional unstructured grids by the advancing-front method', *AIAA Paper 88-0515*, 1988.
4. J. Peraire, J. Peiro, L. Formaggia, K. Morgan and O. C. Zienkiewicz, 'Finite element Euler computations in three dimensions', *AIAA Paper 88-0032*, 1988.
5. Y. Kallinderis and S. Ward, 'Prismatic grid generation for 3-D complex geometries', *AIAA J.*, **31**, 1850–1856 (1993).
6. K. Nakahashi, 'Optimum spacing control of the marching grid generation', *AIAA Paper 91-0103*, 1991.
7. V. Parthasarathy, Y. Kallinderis and K. Nakajima 'Hybrid adaptation method and directional viscous multigrid with prismatic – tetrahedral meshes', *AIAA Paper 95-0670*, 1995.
8. Y. Kallinderis, 'Algebraic turbulence modeling for adaptive unstructured grids', *AIAA J.*, **30**, 631–639 (1992).
9. S. Ward and Y. Kallinderis, 'Hybrid prismatic/tetrahedral grid generation for complex 3-D geometries', *AIAA Paper 93-0669*, 1993.
10. Y. Kallinderis and P. Vijayan, 'An adaptive refinement coarsening scheme for 3-D unstructured meshes', *AIAA J.*, **31**, 1440–1447 (1993).

11. R. Löhner and J. Baum, 'Numerical simulation of shock interaction with complex geometry three-dimensional structures using a new adaptive H-refinement scheme on unstructured grids', *AIAA Paper 90-0700*, 1990.
12. W. Schroeder and M. Shepard, 'An $O(N)$ algorithm to automatically generate geometric triangulations satisfying the Delaunay circumsphere criteria', *RPI, SCOREC Report 12-1989*, 1989.
13. Y. Kallinderis and J. R. Baron, 'A new adaptive algorithm for turbulent flows', *Comput. Fluids J.*, **21**, 77–96 (1992).
14. Y. G. Kallinderis and J. R. Baron, 'Adaptation methods for a new Navier–Stokes algorithm', *AIAA J.*, **27**, 37–43 (1989).
15. V. Parthasarathy and Y. Kallinderis, 'A 3D finite-volume scheme for the Euler equations on adaptive tetrahedral grids', *J. Comput Phys.*, **113**, 249–267 (1994).