AIAA JOURNAL
Vol. 33, No. 6, June 1995

Semistructured Mesh Generation for Three-Dimensional Navier–Stokes Calculations

Stuart D. Connell* and Mark E. Braaten†
General Electric Corporate R&D Center, Schenectady, New York 12301

To mesh complex geometries efficiently, it is the authors' view that a fully unstructured mesh generation technique holds the most promise. There are many papers detailing mesh generation and flow solutions for inviscid flows on unstructured meshes; however, the computation of viscous flows on unstructured meshes is a relatively new area of research. To accurately and efficiently compute viscous flows, highly stretched cells aligned in the proper direction are required in the viscous layers. Many of the currently available unstructured mesh generators provide at most a limited ability to create these cells. In the current work a method that builds layers of highly stretched prismatic cells on an existing unstructured surface triangulation is described. The outer surface of this inflated triangulation provides a natural starting point for the advancing front algorithm used to fill the interior with tetrahedra. Care is taken to limit the possibility of a crossed-over mesh in concave geometrical regions. If the mesh does cross over, then the offending cells are removed. The combined algorithm of surface inflation and advancing front is demonstrated on a range of internal and external flow test cases. A limited number of flow solutions are also presented.

I. Introduction

THE generation of a computational mesh around a complex geometry as part of a computational fluid dynamics (CFD) solution is by no means a trivial task. To be of use in a design environment, a high-quality mesh is required that has been generated in a reasonable time. There are two main approaches to this problem, namely block-structured and unstructured meshes.

With the block-structured approach the connectivity of the individual blocks is specified before meshing each block. The blocks may or may not overlap. This method has been widely used to mesh a variety of three-dimensional geometries.[1,2] Meshing times can be excessive, with many man months being typical for complex geometries such as complete aircraft with an installed nacelle. The main difficulty with this approach is the specification of the initial block topology. This is not an easy task, although there has been recent progress aimed at automating this procedure.[3]

The unstructured approach has no requirement for an initial blocking topology. Meshes are made starting from a solid model of the geometry in an automatic fashion. Mesh generation times are reduced from man months with a block-structured method to a few CPU hours. There are a variety of competing approaches: Delaunay-based methods,[4,5] advancing front,[6] and octree-based subdivision.[7] There are also some hybrid methods that begin with block-structured mesh and then destructure it[9,10] or reconnect the points using a Delaunay-based algorithm.[8]

Of the two approaches it appears that the solid model based unstructured approach[4,6,7] holds the most promise for producing a fully automatic mesh generator for complex geometries. Of the available unstructured mesh generators there is no accepted "best" method. It is possible using any of these methods to produce a mesh adequate for an inviscid calculation. Advancing front and Delaunay-based methods have been found to produce smoother and more regular meshes than the irregular meshes typically produced by octree approaches. A comparison between the surface meshes produced by the advancing front[6] and an octree method[7] is shown in Fig. 1. The meshes were produced on identical geometries and contain a comparable number of nodes. The reader is left to decide which is more desirable.

Received Aug. 8, 1994; revision received Jan. 21, 1995; accepted for publication Feb. 5, 1995.

*Mechanical Engineer; currently at Sowerby British Aerospace Research Center, Bristol, England, United Kingdom. Member AIAA.
†Mechanical Engineer.

The desire to perform viscous calculations adds further constraints to the mesh generation algorithm. For these calculations high-aspect-ratio cells aligned with the flow gradient are required in the boundary layer and wake regions. The unstructured mesh generation algorithms[4,6,7] aim to produce tetrahedra close to equilateral. The use of equilateral cells in viscous regions results in a prohibitively expensive algorithm. Ultimately there will be a requirement to adapt the mesh in viscous regions. To be efficient this adaption must be performed in an anisotropic manner.

A recent paper[11] demonstrated a two-dimensional algorithm for generating high-aspect-ratio cells able to be refined anisotropically. A layer of structured quadrilateral cells was wrapped around the body and aligned with wake regions. The interior was then filled with triangular cells. An example of such a mesh is shown in Fig. 2. The quadrilateral mesh gives very efficient resolution of boundary layers and wakes. In addition it permits the directional refinement necessary in viscous regions. This is illustrated in Fig. 3. A natural extension of this approach to three dimensions is first to place structured layers of triangular based prismatic cells on an existing surface triangulation. This process can be thought of as inflating the surface triangulation. The interior is then filled with a tetrahedral mesh. The prismatic cells can have a high aspect ratio that will permit efficient resolution of the boundary layers. They also possess quadrilateral faces that will ultimately permit the directional refinement illustrated in Fig. 3.

Octree methods do not lend themselves well to this approach. With these methods the surface triangulation results from the interior mesh generation. A prismatic mesh could be built on this triangulation; however, the interior mesh would have to be remade and required to conform to this new boundary triangulation. This capability has been demonstrated in Ref. 12, where the original surface triangulation came from a source other than the octree mesh generator.

It is the prismatic mesh generation that is the subject of this paper. There are a variety of methods currently being developed to produce near-wall prismatic meshes.[12–15] All of the methods begin with a triangulated surface that is then marched (inflated) towards the interior in a series of steps.

In Ref. 12 this marching is achieved through representing the initial surface triangulation by a number of nonintersecting hexahedral elements (voxels). The triangulation is contained within these voxels. The outer surface of these voxels is then smoothed to form the first inflated surface. Computing the intersection of normals from the original surface with the inflated surface forms the first prismatic layer. The process is then repeated to form the complete prismatic mesh. This method has the disadvantage (also common to octree-

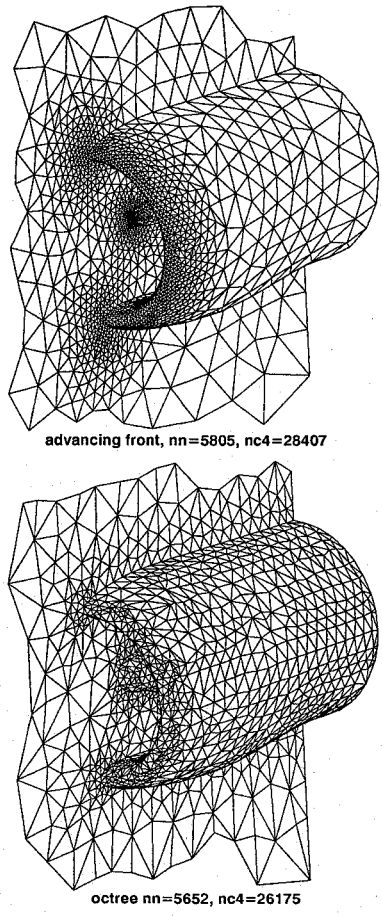


Fig. 1 Comparison of advancing front and octree meshes.

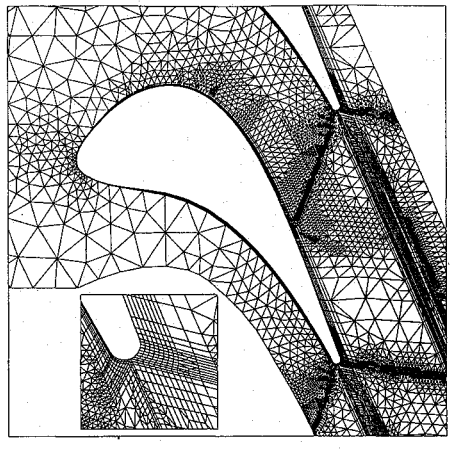


Fig. 2 Mixed triangular/quadrilateral mesh.

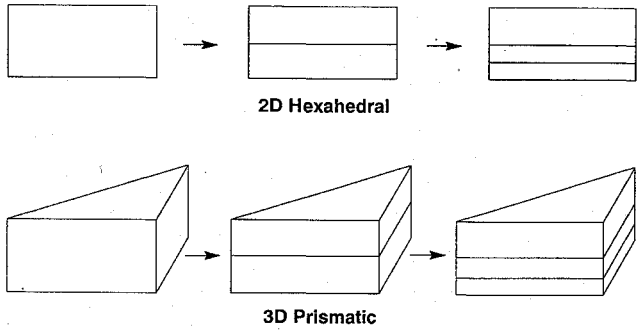


Fig. 3 Two-dimensional and three-dimensional directional cell division.

based methods) that the mesh will change if the geometry is rotated with respect to the coordinate system.

The hyperbolic mesh generation technique has been used widely for generating structured meshes about complex geometries. The procedure typically begins with a quadrilateral surface mesh that is marched into the interior using a set of hyperbolic equations to produce a hexahedral mesh. Initial attempts at extending the technique to start from a surface triangulation are presented in Ref. 14. The algorithm described here was found to be prone to crossovers at sharp internal and external corners unless many explicit steps of the algorithm were taken. For some relatively simple test cases, even with many steps, a valid mesh remained impossible to obtain.

The advancing layers method[13,15] inflates the surface along quasinormal directions with a modified advancing front type algorithm. When certain geometrical criteria are satisfied, such as distance from a wall, the algorithm reverts to the standard method. Numerical experiments indicate that this method can be made less prone to crossovers at sharp corners than the hyperbolic technique. It also has the advantage that near-wall spacing may be specified directly for turbulent flows.

The algorithm presented in the current work also uses quasinormal directions as a starting point for generating the prismatic mesh. New nodes are placed along the normal directions to form a structured mesh of prismatic elements that wrap around the viscous surfaces. The volume mesh generator uses this inflated triangulation as its initial front rather than the initial triangulation. Hence this inflation step forms a natural part of the advancing front method.

This algorithm has successfully generated high-quality meshes about many complex geometries.

The advancing front computer program (FELISA) supplied by K. Bey of NASA Langley and written by J. Peraire, K. Morgan, and J. Peiro was used (unmodified) to generate the surface triangulations and tetrahedral volume meshes.

II. Terminology

In this section some basic solid modeling and advancing front mesh generation terminology will be defined to provide a basis for descriptions used later in the paper.

A. Solid Modeling

Solid models are composed of two basic parts: topology and geometry. The topology describes the connectivity between various entities describing the model. Topology says nothing of the shape of these entities.

There are three basic topological entities: surface, edge, and vertex. An edge has a vertex at each end, and a surface has a number of surrounding edges. A number of these entities, when combined together, form a topological description of the solid model.

Each of these entities is tied to a geometrical definition to provide a complete description of the object. When tied to geometry, the entities are referred to as geometrical surfaces, geometrical edges, and geometrical vertices.

B. Advancing Front Algorithm

In this subsection the advancing front algorithm of Ref. 6 will be briefly described.

The method begins with a solid model of the geometry and a background mesh. The background mesh specifies the desired mesh spacings in the region to be meshed.

The first stage of the algorithm is to place mesh points on the geometrical edges to reflect the spacing distribution defined by the background mesh. For each geometrical surface the distribution of points on the geometrical edges surrounding the surface form a closed front of edges. The algorithm now advances this front, building triangular cells and thereby adding new nodes. This process is repeated until the surface is covered with triangular cells.

Having triangulated all surfaces, the second stage of the algorithm can start. Beginning with the surface triangulation which comprises a closed front, tetrahedra are now built on these triangular faces adding new nodes. This front is advanced until the domain is filled with tetrahedral cells.

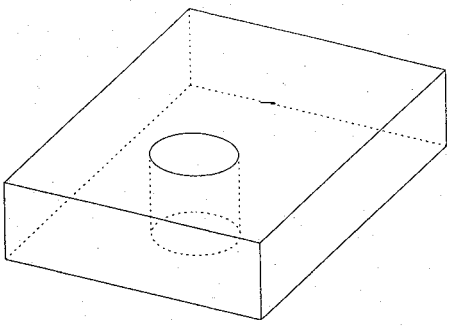


Fig. 4 Cylindrical hole in box.

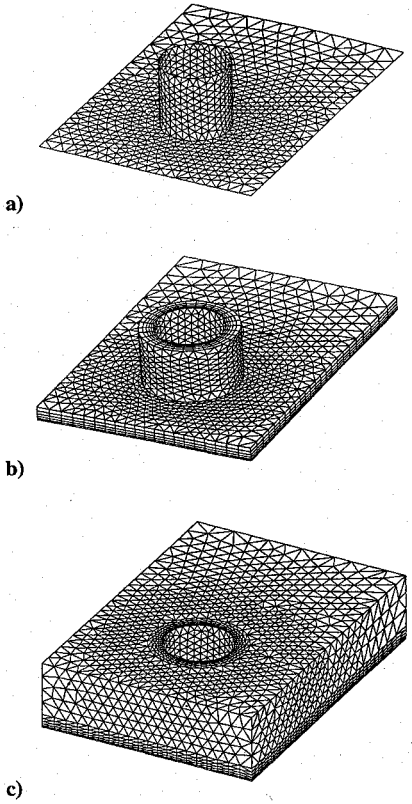


Fig. 5 Stages of viscous mesh generation.

III. Semistructured Mesh Generation

To describe the combined algorithm of surface inflation and advancing front, a rectangular box with a cylindrical hole will be used as an example. This geometry is illustrated in Fig. 4. The cylindrical surface and lower wall are marked as viscous and are to be inflated; all other boundaries are either inlet surfaces, exit surfaces, or inviscid walls and will not be inflated.

The procedure begins by triangulating the viscous walls using the surface meshing part of the advancing front algorithm. This surface mesh is shown in Fig. 5a.

The next stage in the procedure is to compute a quasinormal direction at each node on the surface triangulation. The algorithm used to compute these normal directions is described in Sec. IV. The initial surface is now inflated a specified distance δ along these quasinormal directions. This gives the first layer of prismatic cells. This inflation process is repeated a number of times using different values of δ but the same normal direction. Typically δ varies between one layer and the next by a fixed geometrical factor ranging between 1.0 for a uniform mesh and 5.0 for a highly stretched mesh. The resultant prismatic mesh is shown in Fig. 5b. The total thickness of this prismatic region encompasses the expected boundary layer.

The remaining nonviscous surfaces are now triangulated. Any new nodes and edges generated by the inflation algorithm that lie on these surfaces are required to form part of the initial front for these surfaces. The newly triangulated surfaces and the inflated triangulation are combined to form a closed front. This becomes the initial front for the interior mesh generation algorithm.

The prismatic and tetrahedral meshes are now combined into one mesh. Many flow solvers are limited to a purely tetrahedral mesh. In this case the prismatic cells are divided into three tetrahedra. The algorithm used to perform this division is described in Sec. V. The resulting semistructured mesh is shown in Fig. 5c.

For many cases this scheme is found to work well. However, for some complex geometries a suitable algorithm to produce a set of normals, which in turn lead to a valid prismatic mesh of reasonable quality, remains elusive. In addition, for boundaries that are close to one another, it may be possible for the prismatic meshes to overlap. To produce a robust mesh generator, a fallback position is adopted. After the prismatic mesh is generated, a number of checks for cell quality are made. These checks ensure the following:

1) Each prismatic cell and its resultant tetrahedra have positive volume.

2) No cell intersects any other.

3) Cells are of reasonable quality.

Any cell failing these tests is tagged for deletion from the mesh.

The intersection test is achieved through an advancing front type algorithm. The initial surface triangulation is regarded as the starting front. The first prismatic cell is now considered. A check is made to ensure that no edges of the prism intersect any face in the front. If no intersections are found, then the front is modified to include the faces and edges of this prism and to delete the base face. If intersections are found, then the cell is tagged for deletion, and the front remains unchanged. This process is repeated for all cells in the current layer and then for each layer until all cells have been checked.

Occasionally after identifying the intersecting cells, the case of two cells that are very close to intersecting but do not actually intersect may arise. In this region the advancing front algorithm may generate some poor quality tetrahedra. Thus it has proved advantageous to provide the option to remove a number of layers of cells surrounding cells marked for deletion. This is done by looping over cells, transferring the deletion tags to the nodes of the cell. A second loop over cells is now performed, transferring the tags back to cells. Each time these two loops are completed a further layer of cells is tagged for deletion.

When cells are removed, triangular and quadrilateral faces of elements below the inflated surface are exposed. To form a front the quadrilateral faces are divided into triangles. It is possible for these triangular faces to possess a high aspect ratio especially near the wall. A front containing such faces has proven problematical to the volume mesh generator. To remove these faces a two-step procedure is adopted. This is shown in Fig. 6 for the simple case of a flat plate where a single prismatic element has been tagged for deletion. The vertical dimension has been exaggerated to better illustrate the procedure.

Step 1: Additional cells are removed to provide a "stepped" front as shown in Fig. 6b.

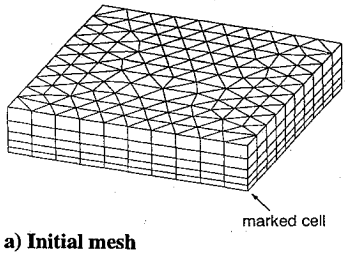
Step 2: The exposed quadrilateral faces are now removed by collapsing the top edge of each quadrilateral face into the bottom edge. The modified front is shown in Fig. 6c.

This new front has no high-aspect-ratio triangles and provides a good starting point for the volume mesh generation. The collapsing of the quadrilateral faces results in the formation of pyramid and tetrahedral elements.

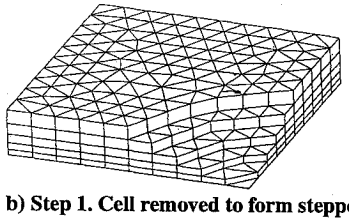
The newly exposed triangular faces of prisms, pyramids, and tetrahedra, together with the inflated faces and the triangulation of the inviscid surfaces, are combined to form a front for the volume meshing. The volume mesh is generated using the advancing front algorithm.[20] It should be noted here that alternatively a Delaunay-based three-dimensional meshing algorithm, for example, Ref. 4, could be used to fill the volume with tetrahedra.

IV. Quasinormal Generation

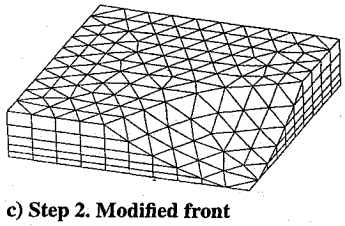
The algorithm for generating the normals has been developed to minimize the possibility of mesh crossovers and produce a high-quality mesh. This results in as much of the object as possible being wrapped in a prismatic mesh.



a) Initial mesh



b) Step 1. Cell removed to form stepped front



c) Step 2. Modified front

Fig. 6 Removal of high-aspect-ratio faces.

The normal generation is based around information from the underlying surface model rather than just the surface triangulation. The reasons for using the solid model are threefold:

1) It has been found to produce smoother meshes.

2) Interrogation of the surface model is crucial for nodes that lie on the junction of inviscid and viscous surfaces.

3) A much better normal representation is achieved for coarse meshes. These coarse meshes will ultimately provide the basis for an adaptive algorithm.

The generation of the quasinormals is a three stage process:

1) Normals are first computed at geometrical vertices. These are calculated as the angle weighted average of the adjacent surface normals. The angle is the one between the two geometrical edges adjacent to the geometrical surface in question meeting at the vertex. A modified scheme is used for sharp leading and trailing edges as illustrated in Fig. 7. Here the normal is required at $\boldsymbol{P}$. An initial estimate of the normal n_1 is obtained using the preceding scheme but not including the two surfaces S_2 and S_3 adjacent to the sharp edge. The normal to the sharp edge n_2, is now computed as the average of the two adjacent surface normals. The vertex normal n is then taken as the average of n_1 and n_2.

2) Next, normals at nodes on geometrical edges are now computed. An initial estimate of this normal is the average of the two adjacent surface normals. These normals are now decomposed into edge tangent and edge normal components. The direction normal to the edge is the average of the two adjacent surface normals. These two components are smoothed independently with a Laplacian-based smoother using the vertex values as boundary conditions. For sharp edges the edge tangent smoothing changes are neglected.

3) Finally, the normals at nodes on geometrical surfaces are computed. Considering each geometrical surface in turn, an initial estimate of the normal direction is computed as the surface normal from the underlying solid model. Using the normals on the geometrical edges surrounding the surface as boundary conditions, these normals directions are smoothed again using a Laplacian-based smoother.

In general, there will be surfaces that are not required to be inflated, for example, inlets, exits, symmetry planes, and periodic boundaries. If these surfaces are adjacent to an inflated surface, then some special action is required. The algorithm is modified to constrain the inflated nodes to lie on the adjacent geometrical edge or surface. This is done by first modifying the surface normals at these nodes so that they are locally tangent to the adjacent entity. Then after the inflation step is taken, the new nodes are snapped to the closest point on the associated entity.

V. Division of Prismatic Cells

The division of prismatic cells into tetrahedra is not as straightforward as it may first appear. Consider the prismatic cell shown on Fig. 8. What is required is that a split orientations, be assigned to each quadrilateral face such that the prism may be divided into tetrahedra. These orientations may be represented as the arrows on the upper triangular face of the prism. Of the eight possible orientations, two result in cases where the division of the prism is impossible. These are the cases where the arrows are joined nose to tail in a clockwise or counterclockwise fashion around the face. Thus an algorithm is required that will assign edge directions to an arbitrary triangulation that avoids the two disallowed cases. Initially the approach described in Ref. 15 was adopted. However, this algorithm was found to fail in some cases.

A modified scheme has been adopted and may be described as follows:

Loop over nodes. For each node assign a direction to all edges containing the current node. This direction points to the current node. If a direction has already been assigned to the edge, then no action is taken.

This algorithm will work for any triangulation. The proof is straightforward. At any stage on the procedure a triangle will have zero, two, or all three edge directions assigned. Note that it is impossible for a triangle to have only one edge direction assigned. Consider the intermediate case of two edge directions assigned. These directions will point towards the node of the triangle that has already been visited, as is illustrated in Fig. 9. If node c or b is visited next, it is impossible to assign a clockwise or counterclockwise direction to the triangle.

VI. Adaptive Multigrid Flow Solver

The inviscid and viscous solution algorithm is fully described in Refs. 10 and 16. A brief overview will be presented here.

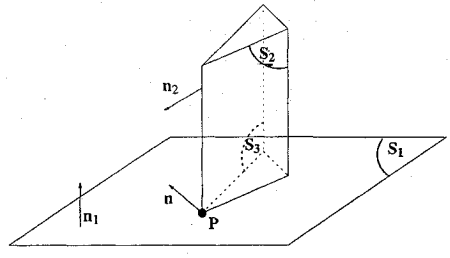


Fig. 7 Normal at sharp edge/endwall.

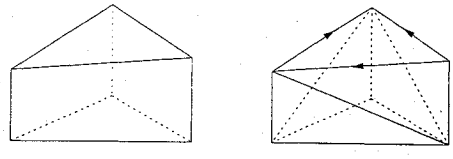


Fig. 8 Prismatic cell division.

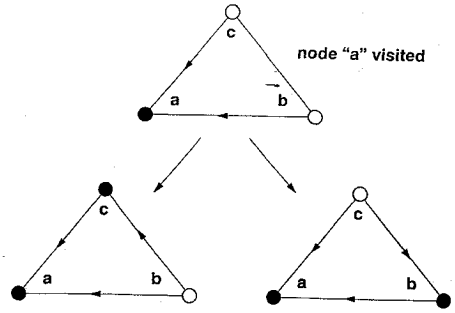


Fig. 9 Edge direction algorithm.

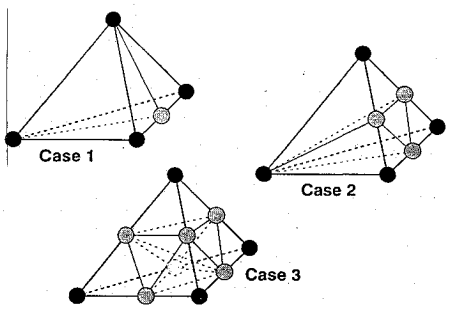


Fig. 10 Three permitted cell divisions.

The full Navier–Stokes equations are solved on a tetrahedral mesh. If the flow is turbulent, then a k-ε turbulence model is used. An explicit multistep Runge–Kutta scheme is employed to integrate the governing equations in time with blended second- and fourth-order smoothing. A series of node centered nonoverlapping control volumes are employed. The Euler operator is computed via a loop over edges. The flux computed through the edge area is accumulated to the adjacent nodes. The viscous part of the residual is obtained via a two-step process. Edge-based derivatives are computed, and these derivatives combined with the edge areas then give the shear stress contribution for that control volume face.

The procedure begins with an initial coarse mesh that is refined based on solution gradients. This refinement process produces the multigrid levels and allows efficient computation of transfer operators. The refinement algorithm examines each edge in the mesh, and if some tolerance is violated, then a new node is placed at the center of that edge. By applying various refinement rules described in Ref. 10, the number of possible cell divisions can be limited to three as illustrated in Fig. 10. The solve and refine cycle is repeated until a solution of the required accuracy is obtained.

VII. Results

The test cases are divided into two groups. The first group (VII.A–VII.G) demonstrate the mesh generation algorithm on a wide range of potential applications. These include a variety of turbomachinery components and a complete aircraft configuration. The second group (VII.H and VII.I) validate the flow solver on two simple laminar test cases.

The CPU time required for the inflation procedure can be divided into two components: the time required to generate the surface normals and the time required to generate and to check the quality of the prismatic mesh. The normal generation time is independent of the number of prismatic layers, but the quality checks scale linearly with the number of layers. For the installed nacelle (case VII.G) the surface mesh contained 11,690 triangles. The CPU times required for the normal generation and cell checking were 11.07 and 18.6 s, respectively, for five layers on an HP 735 workstation. These times are small in comparison with the 840 s required to generate the interior tetrahedral mesh containing 90,000 cells.

A. Leaned Cylinder

This test case demonstrates the robustness of the algorithm for inflating meshes near sharp concave corners. A geometry similar to that in Fig. 4 is used, but the cylinder is leaned at an angle of 45 deg to the endwall. Only half of the geometry is meshed as this aids in viewing the mesh in the sharp corner. Figure 11 shows the prismatic mesh with an enlargement of the sharp corner region. As can be clearly seen, a high-quality mesh is produced in this difficult geometric region. Sharper corners with angles of 20 deg or less have been successfully meshed.

B. Biconvex Airfoil

This test case demonstrates the algorithm on a case with sharp edges. An annular cascade of biconvex airfoils are considered. The scheme described in Sec. IV is used for computing the quasinormal direction for the nodes on the sharp edges. Figure 12 shows the resulting prismatic mesh. Various cuts have been made through the mesh to show the extent and shape of the prismatic region. The upper surface has been removed to aid in viewing the mesh. The enlargement of the cut through the leading edge at the endwall junction illustrates the mesh in that region. As can be seen, a suitable normal has been computed, resulting in a high-quality mesh.

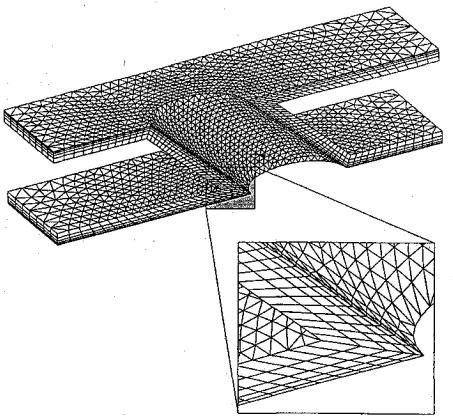


Fig. 11 Prismatic mesh for learned cylinder.

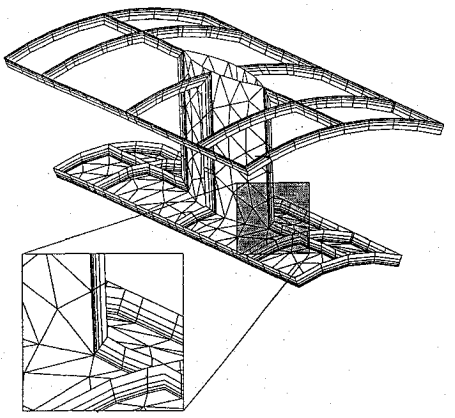


Fig. 12 Prismatic mesh for biconvex airfoil.

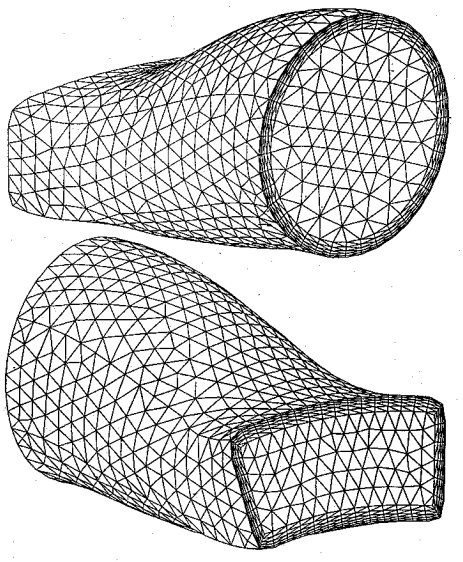


Fig. 13 Combined mesh for transition duct.

C. Transition Duct

The transition duct pictured in Fig. 13 is a component in a power generation gas turbine that directs the exhaust from a cylindrical can combustor into the first-stage turbine nozzle. The transition duct varies in cross section from circular at the inlet to an annular sector at the exit. The flow in the transition duct is nearly incompressible, with Mach numbers in the range from 0.1–0.2 prevailing throughout

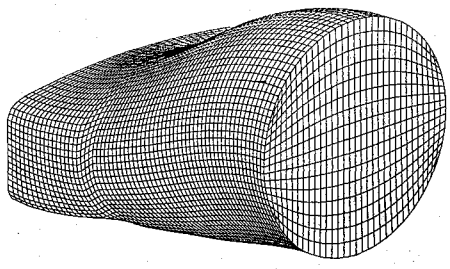


Fig. 14 Distorted structured mesh.

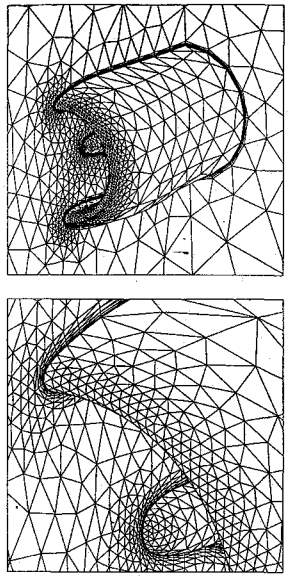


Fig. 15 Combined mesh for nacelle.

most of the duct. Although it may be possible to produce an adequate structured mesh for the annular cross section, any attempt for the circular cross section will result in distorted cells. These distorted regions occur where fictitious corners have been placed on the circular cross section. This is shown in Fig. 14. No such problem exists with the unstructured mesh.

D. Nacelle

A more complex geometry was considered next. This was an aircraft engine nacelle with a centerbody. As the nacelle was symmetrical, only half of it was considered. The combined prismatic and tetrahedral mesh is shown in Fig. 15. Again a high-quality mesh is observed near the viscous surface.

E. Turbine Nozzle

This was the first case examined that required the fallback position of identifying and removing cells that intersected with others or had a negative volume. These invalid cells occurred in the trailing-edge endwall region where the inflation distance is many times the local surface spacing and crossed-over cells may be expected. The resultant prismatic mesh is illustrated in Fig. 16 by making various cuts through the grid.

F. Mixer Nozzle

This nozzle is designed to mix two gas streams and is of similar design to that described in Ref. 17. It is a complex geometry and provides a good test case for the inflation algorithm. The prismatic mesh is shown in Fig. 17. Approximately 25 prismatic cells failed the quality and crossover checks and were removed.

G. Installed Nacelle

This is an extremely complex case on which to test the algorithm. A multiblock mesh on a geometry of this type would typically take many man months to generate. The prismatic mesh is shown in Fig. 18. Although not shown in the figure, the nacelle contains a centerbody that also was inflated. Cells were removed in the vicinity of the pylon/wing junction and the sharp trailing edges of the wing and nacelle.

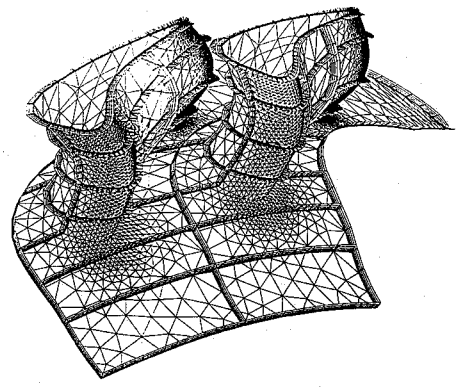


Fig. 16 Prismatic mesh for turbine.

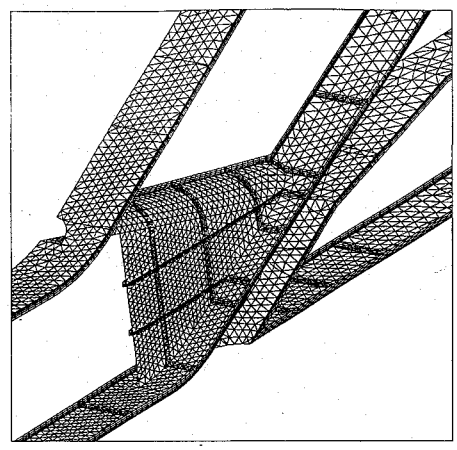


Fig. 17 Prismatic mesh for nozzle.

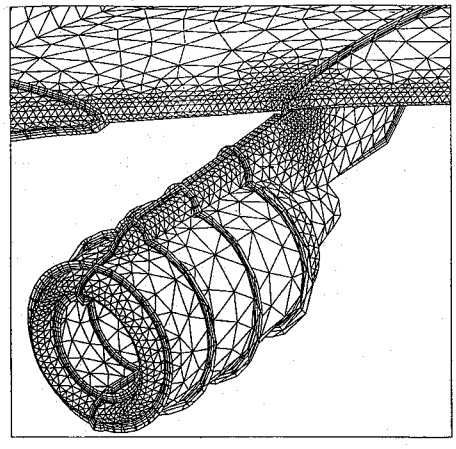


Fig. 18 Prismatic mesh for installed nacelle.

H. Laminar Flat Plate

This two-dimensional incompressible case was modeled in three-dimensions by the mesh shown in Fig. 19. The Reynolds number based on plate length was 2×10^5 and the inlet Mach number was closed to 0.1. The time marching scheme was converged on this initial grid. One refinement step was then performed. This refined mesh had 10 points across the boundary layer.

Results for this case are a comparison of skin friction and velocity profile with the Blasius solution[18] presented in Figs. 20 and 21. The computed results show excellent agreement with the Blasius solution. The slight noisiness in the predicted skin friction is attributable to the numerical scheme being somewhat sensitive to the

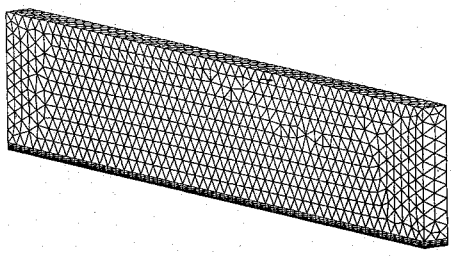


Fig. 19 Flat plate mesh.

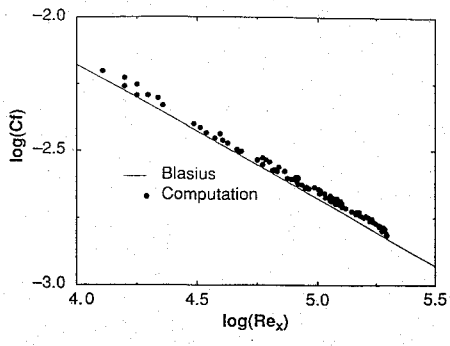


Fig. 20 Skin friction along flat plate.

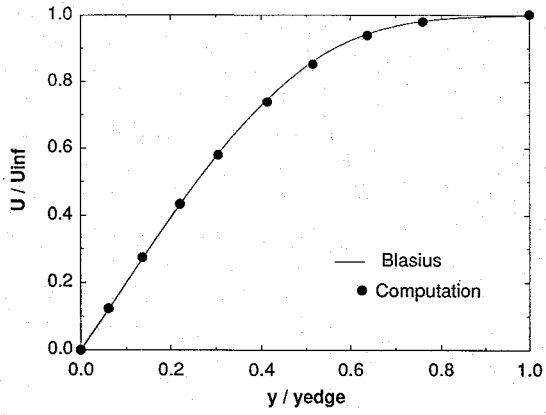


Fig. 21 Velocity profile for flat plate.

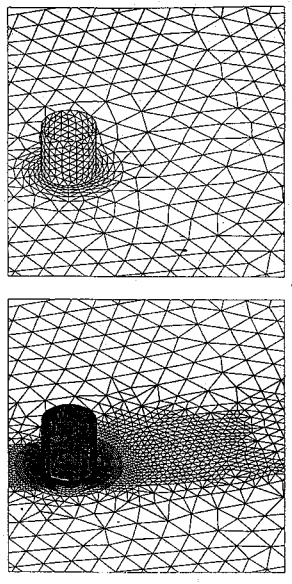


Fig. 22 Initial and final grids for cylinder.

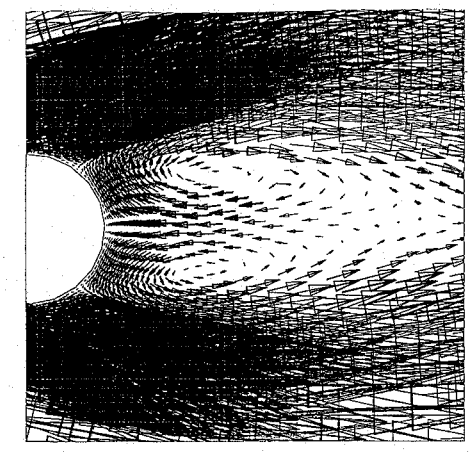


Fig. 23 Flow vectors behind cylinder.

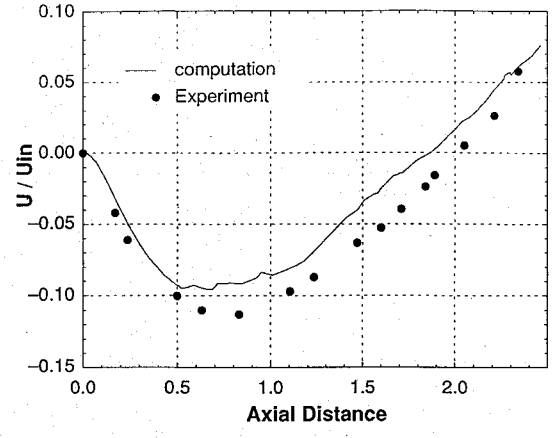


Fig. 24 Velocity magnitude behind cylinder.

mesh irregularities in the viscous region. No problems have been observed with the flow solver due to the large jump in mesh spacing at the edge of the prismatic region.

I. Laminar Circular Cylinder

This is the case of incompressible laminar flow over a circular cylinder. It is again a two-dimensional case and was modeled using a mesh shown in Fig. 22. This mesh is similar to that shown in Fig. 5 except that the lower wall is now inviscid and not inflated. The Mach number was 0.1. and the Reynolds number was 4×10^1 based on the diameter of the cylinder. This is close to the highest Reynolds number permitted before the initiation of vortex shedding. At this Reynolds number there is a large symmetric recirculation region downstream of the cylinder. Figure 23 shows a vector plot of the solution behind the cylinder after two refinements. A comparison of the strength of the recirculation region with test data[19] is shown in Fig. 24. Good agreement with test data is observed.

The refined mesh in Fig. 22 also illustrates why directional refinement is required. Here the desire to refine normal to the wall, to resolve the boundary layer, has resulted in many useless points being placed on the surface of the cylinder.

VIII. Conclusions

An algorithm for generating semistructured meshes suitable for the computation of viscous flows has been described. This method has been demonstrated on a wide range of geometries and forms a natural part of the advancing front algorithm.

IX. Future Work

The obvious next stage in this work is to add the directional refinement of the prismatic cells described earlier. With the meshes now having high-aspect-ratio cells adjacent to walls it is possible that resolving the correct boundary shape during refinement may lead to

a crossed-over mesh. It is planned to use the algorithm described in Ref. 11 to alleviate this problem.

The mesh crossover check currently used involves an $\mathcal{O}(n^2)$ search. For a given face a loop is made over all edges to check for intersection. This scheme is very CPU intensive and the alternate digital tree data structures used by Ref. 20 would reduce this cost significantly.

To improve the solution quality the ability to specify a nonconstant thickness would be useful. This would allow the more accurate modeling of the growth of boundary layers on viscous walls. The extension of prismatic meshes into wake regions would also be useful.

The resolution of wakes using a prismatic mesh is also important. One possible procedure for doing this would be to define a surface leaving the trailing edge of the body. This surface would then be triangulated and a prismatic mesh built on either side to provide the required resolution.

Acknowledgments

The authors wish to acknowledge the General Electric Company for permission to publish this paper. In addition the authors wish to thank K. Bey of NASA Langley for supplying the advancing front mesh generation code (FELISA) written by J. Peraire, K. Morgan, and J. Peiro.

References

[1]Cedar, R. D., Dietrich, D. A., and Ostrander, M. J., "Engine/Airframe Installation CFD for Commercial Transports: An Engine Manufacturers Perspective," Society of Automotive Engineers, SAE 932623, 1993.

[2]Meakin, R. L., "Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations," AIAA Paper 93-3350, July 1993.

[3]Dannenhoffer, J. F., "Automatic Topology Generation from Block Structured Grids," VKI Lecture Series, Jan. 1994.

[4]Weatheril, N. P., "Adaptive Inviscid Flow Solution for Aerospace Geometries on Efficiently Generated Unstructured Tetrahedral Meshes," AIAA Paper 93-3390, 1993.

[5]Holmes, D. G., and Snyder, D. D., "The Generation of Unstructured Triangular Meshes Using Delaunay Triangulation," *Numerical Grid Generation in Computational Fluid Mechanics '88*, Pineridge, Swansea, Wales, UK, 1988.

[6]Peraire, J., Morgan, K., and Peiro, J., "Unstructured Finite Element Mesh Generation and Adaptive Procedures for CFD," AGARD Specialists Meeting, Loen Norway, May 24–25, 1989.

[7]Shepperd, M. S., and Georges, M. K., "Automatic Three Dimensional Mesh Generation by the Finite Octree Technique," *International Journal for Numerical Methods in Engineering*, Vol. 32, 1991, pp. 709–739.

[8]Baker, T. J., "Three Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets," AIAA Paper 87-11124, 1987.

[9]Dawes, W. N., "The Extension of a Solution Adaptive 3D Navier–Stokes Solver Towards Geometries of Arbitrary Complexity," American Society of Mechanical Engineers, ASME Paper 92 GT-363, 1992.

[10]Connell, S. D., and Holmes, D. G., "A 3D Unstructured Adaptive Multigrid Scheme for the Euler Equations," AIAA Paper 93–3339, 1993.

[11]Connell, S. D., Holmes, D. G., and Braaten, M. E., "Adaptive Unstructured 2D Navier–Stokes Solutions on Mixed Quadrilateral/Triangular Meshes," American Society of Mechanical Engineers, ASME Paper 93 GT-99, 1993.

[12]Kallinderis, Y., and Ward, S., "Prismatic Grid Generation with an Efficient Algebraic Method for Aircraft Configurations," AIAA Paper 92-2721, 1992.

[13]Löhner, R., "Matching Semistructured and Unstructured Grids for Navier–Stokes Calculations," AIAA Paper 93-3348, 1993.

[14]Melton, J. E., Pandya, S. A., and Steger, J. L., "3D Euler Flow Solutions Using Unstructured Catesian and Prismatic Grids," AIAA Paper 93-0331, 1993.

[15]Prizadeh, S., "Unstructured Viscous Grid Generation by Advancing Layers Method," AIAA Paper 93-3453, 1993.

[16]Braaten, M. E., and Connell, S. D., "A 3D Unstructured Adaptive Multigrid Scheme for the Navier–Stokes Equations" (in preparation).

[17]Patankar, S. V., *Numerical Heat Transfer and Fluid Flow*, McGraw–Hill, New York, 1980.

[18]Schlicting, H., *Boundary Layer Theory*, 7th ed. McGraw–Hill, New York, pp. 139–143 .

[19]Coutanceau, M., and Bouard, R., "Experimental Determination of the Main Features of the Viscous Flow in the Wake of Circular Cylinder in Uniform Translation, Part 1. Steady Flow," *Journal of Fluid Mechanics*, Vol. 79, Pt. 2, 1977, pp. 231–256.

[20]Morgan, K., Peraire, J., and Peiro, J., "Unstructured Grid Methods for Compressible Flows," AGARD Rept. R-787, 1992.