PROSPECTS AND EXPECTATIONS FOR UNSTRUCTURED METHODS

Timothy J. Baker
Princeton University
Department of Mechanical and Aerospace Engineering
Princeton, New Jersey 08544 USA

SUMMARY

The last decade has witnessed a vigorous and sustained research effort on unstructured methods for computational fluid dynamics. Unstructured mesh generators and flow solvers have evolved to the point where they are now in use for design purposes throughout the aerospace industry. In this paper we survey the various mesh types, structured as well as unstructured, and examine their relative strengths and weaknesses. We argue that unstructured methodology does offer the best prospect for the next generation of computational fluid dynamics algorithms.

INTRODUCTION

Mesh generation has long been recognized as a major pacing item in computational fluid dynamics. But over the last decade, it has rapidly evolved into a broadly based area of research. Early meshes were almost exclusively quadrilateral or hexahedral, and were typically created by defining a suitable coordinate transformation from a unit square, or unit cube, into the physical domain. In this manner, one can create the familiar O, H and C-meshes around airfoils, wings and wing/fuselage combinations. These meshes inevitably possess a high degree of structure or regularity on account of their well defined set of coordinate directions. Regularity that is inherent in a structured mesh can, however, be a serious drawback when generating a mesh to conform with all boundaries of the domain. By the end of the seventies, it had become quite clear that this approach would not, by itself, suffice to handle complex configurations.

The formidable challenge presented by complex geometry has been met in various ways. The use of non-aligned meshes (ref. 31) circumvents the problem of conforming with boundaries at the cost of requiring interpolation formulae to impose the boundary conditions. A series of separate overlayed meshes, each of which conforms to the surface, represents an alternative approach that lifts the burden of interpolation from the boundaries to the flow domain (refs. 5 and 27).

To avoid interpolation issues, it is necessary to create a mesh that conforms with all boundaries and also maintains contiguity of meshlines. In the case of hexahedral meshes, this is best achieved by a multi-block technique which splits the flowfield into a number of blocks or subdomains (refs. 11, 17, 35 and 36). The splitting is chosen so that each subdomain can be covered by a structured hexahedral mesh with an associated set of coordinate directions. In general, the orientation of neighboring blocks will differ and the associated coordinate lines cannot be extended across block boundaries. A multi-block mesh can thus be thought of as structured at the level of an individual block, but unstructured when viewed globally as a collection of blocks. The introduction of paving techniques (refs. 6 and 39) takes this process one step further and generates a hexahedral mesh that can be thought of as entirely unstructured. In fact, one can interpret such a mesh as a multi-block mesh whose blocks are the individual hexahedra.

Triangular or tetrahedral meshes were for a long time the preserve of those in the finite element community. They were of little more than passing interest to developers of the finite difference and finite volume methods that were the main staple of research into computational fluid dynamics. The recent intense interest in tetrahedral meshes (refs. 1, 12, 18, 28, 37 and 38) and flow solvers is, however, an acknowlegment of their utility and the relative ease with which tetrahedral meshes can be generated for complex shapes. The example shown in figure 1, of the flow solution over a supersonic transport, is typical of the type of problem that is now routinely handled by tetrahedral based methods (refs. 8 and 14).

In this paper we review the strengths and weaknesses of different mesh types in terms of ease mesh generation and the implications for flow algorithms. In particular, we consider the issues of flow adaption and viscous calculations. Both of these concerns must addressed to ensure accurate and reliable solutions for almost any fluid dynamics problem. On the basis of these considerations, we argue that unstructured meshes, using tetrahedra or a mixture of tetrahedra and prisms, do offer the best prospect for the next generation of computational fluid dynamics software.

A TAXONOMY OF MESHES

The challenge of generating meshes for complex configurations has encouraged several innovative ideas for meshes based on hexahedra, tetrahedra as well as hybrid mesh types. Traditionally, the term structured has been associated with hexahedral meshes on account of the regularity that is inherent in any hexahedral mesh with a well defined set of coordinate directions. Since no global coordinates can be associated with a tetrahedral mesh, it was therefore natural to refer to these as unstructured meshes.

It is certainly apparent that a multi-block mesh is usually only structured at the block level. One can imagine increasing the number of blocks until each block degenerates into a single hexahedron and one has a fully unstructured hexahedral mesh. Examples of unstructured hexahedral meshes are those that are typically produced by paving techniques (refs. 6 and 39). Likewise, a hexahedral mesh that undergoes mesh enrichment (ref. 10), with the consequent addition of extra points on cell faces and edges, can no longer be thought of as structured. Indeed, flow solvers for such meshes rely on pointers and indirect addressing, and must thus maintain a data structure similiar to that of a tetrahedral flow solver. The more natural subdivision of mesh types is therefore into hexahedral, tetrahedral and hybrid (i.e. those using a mixture of element types). The adjectives "structured" and "unstructured" more properly refer to the degree of regularity whose presence can be discerned by a locally well defined set of coordinate directions.

Among hexahedral meshes there is a wide range from single block, structured to completely unstructured. In between these extremes are (i) the multi-block meshes which can be further subdivided into those with contiguous blocks and those whose blocks are overlayed (refs. 5 and 27), and (ii) non-aligned (i.e. non-boundary conforming) meshes (ref. 31). Contiguous multi-block meshes can be further subdivided into composite or patched, according to whether or not the mesh lines are continuous across block boundaries. Finally, the hybrid meshes include both mixed tetrahedral/hexahedral (refs. 26 and 33) and tetrahedral/prismatic (refs. 16 and 25), as well as the Cartesian meshes (refs. 9, 15, 23 and 32) whose cells are mostly hexahedra, but with some cells consisting of variously shaped polyhedra in regions adjacent to the domain boundaries. An attempt to catalog these mesh types is shown in figure 2.

TRIANGULATION TECHNIQUES

There are three main approaches to tetrahedral mesh generation, (i) octree decomposition, (ii) Delaunay based methods, and (iii) moving front techniques. A brief outline of the differences between these approaches follows.

Octree Decomposition

In two dimensions this procedure can be viewed as a division of the flowfield into a collection of rectangles followed by a division of rectangles into triangles. Rectangles can be further subdivided into four new rectangles, corresponding to the addition of an extra vertex at the mid-point of each side of the original rectangle. For rectangles which intersect the boundary, this subdivision can be repeated until a sufficiently fine resolution has been achieved. The first division of the flowfield into rectangles may be regarded as level zero, the subsequent division as level 1 and so on. Thus the final mesh will contain rectangles at level zero in the flowfield and highly refined rectangles in the vicinity of solid boundaries. Some degree of graduation in the mesh refinement is obtained by requiring adjacent rectangles to differ by no more than one level of subdivision.

The next stage consists of examining those rectangles which intersect the boundaries and replacing each such rectangle by a polygon consisting of that part of the rectangle lying in the flowfield together with the part of the solid boundary that lies inside the rectangle. After this cutting procedure has been applied, the mesh consists of a combination of rectangles in the flowfield together with polygons adjacent to the boundaries. This mesh could serve as a Cartesian mesh. Alternatively, the rectangles and boundary polygons could be further subdivided into triangles to provide a triangulation of the flowfield.

The concept generalizes in an obvious way to three dimensions, although the cutting procedure at the boundaries becomes much more complicated (refs. 34 and 38). The main drawback of this approach, however, is the inability to match a prescribed surface triangulation. In addition, since each surface triangle arises from the intersection of a hexahedron with the boundary, it is not clear how one can control the variation in triangle size and shape.

Delaunay Methods

The Delaunay triangulation provides a sound framework for tetrahedral mesh generation and several Delaunay based methods have been developed. Many of these methods (refs. 1, 12, 21, 29 and 37) exploit an incremental algorithm that starts with an initial triangulation of just a few points. The complete triangulation is generated by introducing a point and locally reconstructing the triangulation after each point insertion. Incremental algorithms can be used equally well for both the initial mesh algorithm and for any further enrichment that may be required by solution adaptive refinement. A particularly attractive feature of this approach is the opportunity to place new points at specified locations with the object of retaining, and in many cases improving, the quality of the mesh. Recent work in this area has produced encouraging results (refs. 3, 4, 29 and 37) showing that mesh quality can be controlled and meshes created to achieve a guaranteed level of quality according to a suitable set of criteria. Figure 3a shows an initial triangulation of a complex planar domain; figure 3b shows the result after selective refinement of this mesh by the so called Voronoi segment method (refs. 4 and 29).

The main difficulty, for any Delaunay method, is the need to ensure surface integrity. This usually requires the triangulation near the surface to be altered in some way by overriding the Delaunay algorithm. Most methods triangulate the entire domain, extracting the surface triangulation afterward. One possibility is to insert the surface points first, identify the tetrahedra which make up the object and then insert the flowfield points so that no flagged tetrahedra are removed (refs. 1 and 2). Other methods (refs. 12 and 37) allow the volume triangulation to proceed unchecked and then re-establish the surface edges and faces by a series of edge/face swaps and the occasional introduction of an extra point.

Moving Front Methods

The main virtue of this approach (refs. 18 and 28) is that it starts from a prescribed boundary triangulation which remains intact throughout the mesh generation process. The boundary triangulation is regarded as a front on which a new layer of tetrahedra is built. As a result, the original front triangles

C-4.

become interior faces of the mesh and a new set of front faces is created. The algorithm continues to build tetrahedra on the new front, growing more tetrahedra until the entire domain has been filled. A particular difficulty of this method occurs in the closing stages of the procedure when the front is folding in on itself and the final vestiges of empty space are replaced by tetrahedra. It is clearly necessary, in the final stages, to maintain good control over the size of the front faces as well as the shape of the unfilled domain that is left.

MESH IMPLICATIONS FOR FLOW ALGORITHMS

Single block structured meshes, which dominated the early development of computational fluid dynamics, are well suited to exploitation by implicit algorithms (e.g. Alternating Direction Implicit methods) as well as mapping easily onto vector computer architectires. The later development of unstructured meshes forces a reappraisal of this situation and there is a consequent need for algorithms which work well on unstructured meshes. In fact, an explicit method appears to be the natural choice for solving the flow equations on an unstructured mesh and the availability of very efficient multigrid techniques removes the otherwise serious restriction of an explicit time step stability limit.

Even so, there is an overhead for the computational work associated with tetrahedral methods. A tetrahedral mesh of N points has roughly 6N cells, 12N faces and 7N edges. The fluxes can be accumulated either across faces, or alternatively, along edges. For tetrahedral meshes there is an obvious advantage in operation count to exploit edge based data structures. A mesh of hexahedra has roughly N cells, 3N faces and 3N edges, so that the operation count is comparable whether fluxes are accumulated across faces or along edges. Either way there is still a clear 2:1 advantage in computational efficiency for a hexahedral flow solver over a tetrahedral flow solver using an edge based data structure on meshes with the same number of points. This leaves open the question of whether comparable accuracy can be achieved on a tetrahedral mesh with fewer points. For a simple shape (e.g. a wing) with an optimal mesh distribution, one would expect comparable accuracy from both mesh types for a similar number of mesh points. For a more complex shape, the regularity inherent in a structured mesh can often lead to an unnecessary refinement in areas far removed from the boundary. In such cases, there is an inefficient distribution in the structured mesh with the possibility that comparable accuracy could be achieved on an unstructured mesh with fewer points. The relative advantage in computational efficiency that is enjoyed by structured methods is therefore not excessive.

In order to attain an optimal use of computer resources, some rearrangement and pre-sorting of a tetrahedral mesh is necessary. For any given tetrahedral cell, the addresses of its forming points may occupy widely separated positions in memory. This feature will cause a severe degradation on computers that depend on a small high speed memory cache. In addition, it is necessary to employ indirect addressing, with the result that each point will be referenced every time it appears as the vertex of a cell. Since many cells are incident at a given mesh point, each point will be referenced several times, and the possibility of vector dependency will inhibit vectorization by the compiler. It is possible to overcome this problem by first sorting the cells into groups so that no point is referenced more than once in each group. On can then override the compiler and force vectorization.

A recursive bi-section procedure can efficiently decompose a mesh into any number of sub-domains with almost identical numbers of cells and edges. This provides a particularly convenient decomposition of the mesh for parallel computer architectures (refs. 7 and 14). Further re-addressing and sorting within each subdomain leads to efficient vectorization and cache use on each of the individual processors.

For a structured multi-block mesh, it is usually necessary to assign whole blocks to an individual processors. Since individual blocks may vary greatly in size, this means that one processor may be assigned to one block while another processor is assigned to several small blocks. It is difficult in this situation to achieve very good load balancing and some compromise in parallel performance can be expected. In contrast, the subdivision of an unstructured mesh maps very readily onto a parallel architecture with nearly perfect load balancing.

An example of the parallel efficiency that can be achieved is shown in figure 4. The figure presents results of running an unstructured flow solver with a tetrahedral mesh on an IBM SP2 parallel system (ref. 14). There is almost perfect scalability for up to 64 processors.

ADAPTIVE REFINEMENT

The use of p-refinement which increases the accuracy of the discrete approximation is particularly effective for elliptic problems where a high degree of smoothness is expected. For problems involving discontinuities and sharply defined features, the use of mesh movement (r-refinement) or mesh enrichment (h-refinement) appears to be more suitable. Mesh movement has the virtue of leaving the mesh size and connectivity intact. Thus, a partition of the mesh into subdomains remains well balanced and re-partitioning of the mesh is not required. However, the movement or repositioning of mesh points inevitably reduces the definition in some parts of the domain in order to increase the resolution elsewhere. Furthermore, moving mesh lines is an inherently perilous procedure and special care must be taken to avoid overlapping of mesh lines and the appearance of cells with negative volume.

The remaining alternative, mesh enrichment, is particularly attractive since the mesh is altered only in the region where greater resolution is needed. For a parallel implementation of the flow algorithm, this requires a new partitioning of the mesh to maintain good load balancing. The computational cost of re-partitioning and sorting will therefore influence how often the mesh should be refined over the course of a calculation.

For a hexahedral mesh, the application of mesh enrichment (refs. 9, 10 and 15) introduces new points on the edges and faces as well as the center of a hexahedron. At the interface between a refined cell and a non-refined cell, four refined faces will abut a non-refined cell face. This upsets the regularity of a structured mesh and imposes alterations (e.g. the use of pointers and indirect addressing) on the flow solver, causing the flow algorithm data structure to closely resemble that of a tetrahedral based flow solver.

For a triangular or tetrahedral mesh, the triangulation that results from mesh enrichment (refs. 19, 21 and 24) will consist of a new collection of triangles or tetrahedra. In this case, no modification of the flow solver is needed and a tetrahedral mesh thus provides a natural setting for mesh enrichment. This characteristic is one of the key advantages of an unstructured, and more particularly, a tetrahedral based method.

There are two main approaches to mesh enrichment. Edge bisection (ref. 19) (i.e. adding new mesh points at the mid-points of the edges of candidate tetrahedra) leads to refined cells of the same aspect ratio as the original cell. This method is simple to implement and can be applied to any triangular or tetrahedral mesh. However, although it preserves cell aspect ratio, addition of new points in this manner will not improve the mesh quality if the original mesh is extremely coarse with several bad aspect ratio cells. An alternative approach, as mentioned above, is available with a Delaunay based method (refs. 4, 21 and 24). The insertion of new points at specified locations and re-triangulation by a Delaunay algorithm provides a great deal of flexibility and control over mesh quality.

NAVIER STOKES MESHES

There is ample empirical evidence to show that solutions of the Euler equations for inviscid flow can be carried out to a high degree of accuracy on a tetrahedral mesh. However, there is still considerable debate as to whether high aspect ratio triangular, or tetrahedral, cells can properly resolve a shear layer. In other words, can the Reynolds averaged Navier Stokes equations be solved with sufficient accuracy on a tetrahedral mesh?

It is known that the truncation error of a finite volume discretization depends on the shape of the control volume. In particular, a trapezoidal approximation for a cell vertex method though nominally second order, becomes only first order accurate unless the control volume possesses central symmetry (ref. 30). The effect is localized and it appears that the global solution error remains second order (ref. 13). For the Euler equations, the possible loss of second order accuracy in truncation error does not appear to be a serious problem. However, it is most likely that this will pose a serious problem when trying to resolve a boundary layer flow.

The control volume associated with a given point P corresponds to the boundary of the collection of cells incident at point P. In the planar case, the cells are triangles and the control volume is the polygon formed by the edges opposite P (figure 5). In three dimensions, the cells are tetrahedra and the control volume is the polyhedron formed by the faces opposite P. Although it is very difficult to ensure central symmetry, a good quality mesh should be close enough to this ideal for there to be little degradation in truncation error. In particular, this should be the case for an isotropic distribution of mesh points with low aspect ratio cells.

In the case of a highly stretched mesh, it is much more difficult to maintain a control volume that has nearly central symmetry. It seems plausible that one way to come close to this ideal, is to use a high degree of structure in the mesh. Thus, in a boundary layer, one could insert points normal to the surface, just as one does for a structured mesh. In two dimensions these points could be connected to form right triangles (see figures 5 and 6). The surface curvature would cause the "right angles" to be slightly greater or less than 90^o, but for a continuously turning surface these deviations would be small. When the surface tangent is discontinuous, however, special treatment is needed. At a trailing edge, for example, a wake region of similarly constructed cells should be inserted. At a re-entrant corner, the thin layer Navier Stokes assumption breaks down and an isotropic mesh distribution is then appropriate.

The requirement of central symmetry imposes rigid constraints on the triangle connectivity. Figure 5 shows that central symmetry of the control volume occurs when the hypotenuse of each right triangle is oriented in the same direction. In figure 7, one of the diagonals has switched direction and the control volume is no longer centrally symmetric.

In three dimensions this issue becomes more problematic and it is unlikely that central symmetry of control volumes can be properly assured. However, it again seems plausible that a carefully constructed mesh of tetrahedra, each with three mutually orthogonal faces, is most likely to approach this ideal. Work along these lines has been presented by Pirzadeh (ref. 28) for the moving front method and by Marchant and Weatherill (ref. 20) and Marcum (ref. 22) for Delaunay based methods.

It remains to be seen whether accurate solutions of shear layer flows can be obtained for the Reynolds numbers of interest on meshes composed entirely of triangles or tetrahedra. An alternative approach is the use of a hybrid mesh. For example, one could use a quadrilateral, or hexahedral, mesh close to the surface and change to an unstructured mesh outside the viscous region. The disadvantage of this remedy is that one is again faced with most of the restrictions that arise with structured meshes. Another version of the hybrid mesh that offers more of the flexibility one looks for in an unstructured method, is provided by using prisms in the boundary layer region. Work in this area has been pioneered by Nakahashi (ref. 25) and more recently by Kalinderis et al (ref. 16). There are still some disadvantages to this approach; the use of two different cell types complicates the flow solver and mesh adaption no longer has the simplicity available on purely tetrahedral meshes. However, this may be the price that one has to pay in order to obtain highly accurate Navier Stokes solutions.

FUTURE DIRECTIONS

In assessing the current status of mesh generation methods and future developments, it seems likely that there will be an increasing emphasis on two key requirements. First, there is a strong drive towards methods that are easy to use and reliable (i.e. user friendly). Second, there is an increasing expectation that the computational results should be accurate, ideally to within some known tolerance. As improved computer hardware opens up the opportunity to attack increasingly complex problems, an effective response to both of these issues will become even more urgent.

Although often regarded as a separate entity, it is important to remember that the mesh generator is a central part of the computational environment, interfacing on one side (through the CAD system) with the surface definition of the geometry, and on the other side with the flow solver. The quality of the mesh directly influences the quality of the solution, unfortunately in ways that are usually difficult to quantify. Mesh quality in the volume mesh can be related to cell aspect ratio and size variation. But there is also the question of the degree to which the surface mesh matches the true surface. Surface definition is therefore an important concern. It is clear that the interface between a surface mesh generator and the CAD system should preserve the surface definition. However, it is also necessary to ensure that the CAD description is sufficiently well defined to meet the needs of aerodynamic flow calculations. For example, it is often the case that a CAD definition is provided by a series of patches which do not always abut correctly at their joins. Furthermore, for Navier Stokes computations, small perturbations in the surface definition could easily project a significant way into the boundary layer, wreaking havoc on a Navier Stokes mesh and the resulting flow prediction.

It is therefore necessary to (i) quantify the quality of the underlying surface, placing tolerances on what deviations are acceptable, (ii) obtain measures of how well a surface mesh matches the true surface definition, and (iii) generate a volume mesh that conforms to the surface mesh and meets a set of previously defined mesh quality measures. It is important that adaptive refinement should be carried out without the need for any user intervention. Moreover, the refined mesh should also meet the same mesh quality measures that are required for the original mesh.

In order to put together a seamless, user friendly software environment, a high degree of modularity with clean interfaces between each software module is desirable. First, one requires a link between the CAD system and the surface mesh generator. It is likely that the interface will need to carry out checks on the CAD surface definition, and possibly process the CAD data to obtain a surface description that meets the stringent requirements for computational aerodynamics. The surface mesh generator will require some input from the user to define the mesh density, perhaps to the extent of defining surface patches and indicating which parts of the surface require high resolution. However, the generation of the surface mesh to a specified quality, in terms of mesh aspect ratio and size variation, should be handled entirely automatically. It would appear that surface mesh generation can be carried out with comparable ease for either quadrilateral or triangular elements.

The volume mesh generator should create a mesh, entirely automatically, that matches the prescribed surface mesh. It is the opinion of the author that ease of volume mesh generation and adaptive refinement are key advantages that will continue to favor tetrahedral meshes over their hexahedral counterparts. For Navier Stokes computations, however, it is quite possible that some compromise must be made and tetrahedral elements will not be able to capture the features of shear layers with sufficient accuracy. The most promising combination in this case would be the use of a prismatic mesh in the shear layer with tetrahedra elsewhere. It seems likely, however, that purely tetrahedral meshes will be used in the near term for Navier Stokes calculations, since tetrahedral mesh generation has advanced to a high degree of automation. In the longer term, the demand for high accuracy will probably drive the development of methods based on a combination of prisms and tetrahedra for three dimensional Navier Stokes computations.

REFERENCES

1. Baker, T.J.: Three Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets, AIAA 8th Computational Fluid Dynamics Conference, Honolulu, Hawaii, June 1987, AIAA Paper 87-1124.

2. Baker, T.J.: Shape Reconstruction and Volume Meshing for Complex Solids, Int. J. Num. Meth. Eng., Vol. 32, No. 4, 1991, pp. 665-675.

3. Baker, T.J.: Point Placement and Control of Triangle Quality for Inviscid and Viscous Mesh Generation, Proc. 4th Int. Conference on Numerical Grid Generation, (edited by Weatherill), Swansea, UK, April 1994, pp. 137-149.

4. Baker, T.J.: Triangulations, Mesh Generation and Point Placement Strategies, in Computing the Future, (edited by D. Caughey), Cornell University, Ithaca, NY, November, 1994, pp. 101-115.

5. Benek, J.A.; Buning, P.G.; and Steger, J.L.: A 3-D Chimera Grid Embedding Technique, AIAA 7th Computational Fluid Dynamics Conference, Cincinnati, Ohio, June 1985, AIAA Paper 85-1523.

6. Blacker, T.D.; and Stephenson, M.B.: "Paving: A New Approach to Automated Quadrilateral Mesh Generation", Int. J. Num. Meth. Eng., Vol. 32, No. 4, 1991, pp. 811-847.

7. Cheng, W.; Jameson, A.; and Mitty, T.J.: AIRPLANE on the IBM Parallel System SP1, Parallel CFD '94 Conference, Kyoto, Japan, May 1994.

8. Cliff, S.; Hicks, R.; and Baker, T.J.: Design and Computational/Experimental Analysis of Low Sonic Boom Configurations, Proc. High Speed Research Sonic Boom Workshop, NASA Langley, June 1994.

9. Coirier, W.J.; and Powell, K.G.: A Cartesian, Cell-Based Approach for Adaptively-Refined Solutions of the Euler and Navier-Stokes Equations, AIAA Aerospace Sciences Meeting, Reno, NV, January, 1995, AIAA Paper 95-0566.

10. Davis, R.L.; and Dannenhoffer, J.F.: 3-D Adaptive Grid-Embedding Euler Technique, AIAA 31st Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1993, AIAA Paper 93-0330.

11. Eiseman, P.R.; Cheng, Z.; and Häuser, J.: Applications of Multiblock Grid Generations with Automatic Zoning, Proc. 4th Int. Conference on Numerical Grid Generation, Swansea, (edited by Weatherill), UK, April 1994, pp. 123-134.

12. George, P.L.; Hecht, F.; and Saltel, E.: Constraint of the Boundary and Automatic Mesh Generation, Proc. 2nd Int. Conference on Numerical Grid Generation, Miami, FL, December 1988, pp. 589-597.

13. Giles, M.B.: Accuracy of Node-Based Solutions on Irregular Meshes, 11th Int. Conference on Numerical Methods in Fluid Dynamics, Williamsburg, VA, June 1988.

14. Jameson, A.; Cliff, S.; Thomas, S.; Baker, T.J.; and Cheng, W.: Supersonic Transport Design on the IBM Parallel System SP2, Proc. Computational Aerosciences Workshop, Santa Clara, CA, March 1995.

15. Karman, S.L.: SPLITFLOW: A 3-D Unstructured Cartesian/Prismatic Grid CFD Code for Complex Geometries, AIAA 33rd Aerospace Sciences Meeting, Reno, NV, January 1995, AIAA Paper 95-0343.

16. Kallinderis, Y.; and Ward, S.: Prismatic Grid Generation with an Efficient Algebraic Method for Aircraft Configurations, "AIAA 10th Applied Aerodynamics Conference, Palo Alto, CA, June 1992, AIAA Paper 92-2721.

17. Lee, K.D.; Huang, M.; Yu, N.J.; and Rubbert, P.E.: Grid Generation for General Three-Dimensional Configurations, Proc. NASA Langley Workshop on Numerical Grid Generation Techniques, (edited by Smith), October 1980.

18. Löhner: Generation of Three-Dimensional Unstructured Grids by the Advancing-Front Method, AIAA 26th Aerospace Sciences Meeting, Reno, NV, January 1988, AIAA Paper 88-0515.

19. Löhner: Adaptive H-Refinement on 3-D Unstructured Grids for Transient Problems, AIAA 27th Aerospace Sciences Meeting, Reno, NV, January 1989, AIAA Paper 89-0365.

20. Marchant, M.J.; and Weatherill, N.P.: Unstructured Grid Generation for Viscous Flow Simulations, Proc. 4th Int. Conference on Numerical Grid Generation,(edited by Weatherill), Swansea, U.K., April 1994, pp. 151-162.

21. Marcum, D.L.; and Weatherill, N.P.: A Procedure for Efficient Generation of Solution Adaptive Unstructured Grids, Proc. 4th Int. Conference on Numerical Grid Generation, (edited by Weatherill), Swansea, U.K., April 1994, pp. 639-650.

22. Marcum, D.L.: Generation of Unstructured Grids for Viscous Flow Applications, 33rd Aerospace Sciences Meeting, Reno, NV, January 1995, AIAA Paper 95-0212.

23. Melton, J.; Berger, M.; and Aftosmis, M.: 3-D Applications of a Cartesian Grid Euler Method, 33rd Aerospace Sciences Meeting, Reno, NV, January 1995, AIAA Paper 95-0853.

24. Mitty, T.J.; Jameson, A.; and Baker, T.J.: Solution of Three-Dimensional Supersonic Flowfields via Adapting Unstructured Meshes, Computers Fluids, Vol. 22, No. 2/3, 1993, pp. 271-283.

25. Nakahashi, K.: Computations of Three-Dimensional Navier-Stokes Equations by Using a Prismatic Mesh, Proc. 6th NAL Symposium on Aircraft Computational Aerodynamics, Tokyo, Japan, June 1988.

26. Nakahashi, K.; Obayashi, S.: Viscous Flow Computations Using a Composite Grid, AIAA 8th Computational Fluid Dynamics Conference, Honolulu, Hawaii, June 1987, AIAA Paper 87-1128.

27. Onslow, S.H.; Blaylock, T.A.; Albone, C.M.; and Hodges, J.: The FAME Mesh Generation System - its Synthesis, Flexibility and Capabilities, Proc. 4th Int. Conference on Numerical Grid Generation, (edited by Weatherill), Swansea, U.K., April 1994, pp. 259-269.

28. Pirzadeh, S.: Unstructured Viscous Mesh Generation by Advancing-layers Method, AIAA 11th Applied Aerodynamics Conf. AIAA Paper 93-3453, Monterey, August 1993.

29. Rebay, S.: Efficient Unstructured Mesh Generation by Means of Delaunay Triangulation and Bowyer-Watson Algorithm, J. Comp. Physics, Vol. 106, 1993, pp. 125-138.

30. Roe, P.L.: Error Estimates for Cell-Vertex Solutions of the Compressible Euler Equations, ICASE Report No. 87-6.

31. Samant, S.S.; Bussoletti, J.E.; Johnson, F.T.; Burkhart, R.H.; Everson, B.L.; Melvin, R.G.; Young, D.P.; Erickson, L.L.; Madson, M.D.; and Woo, A.C.: TRANAIR: A Computer Code for Transonic Analyses of Arbitrary Configurations, AIAA 25th Aerospace Sciences Meeting, Reno, NV, January 1987, AIAA paper 87-0034.

32. Schneiders, R.; Oberschelp, W.; Weiler, R.; Kopp, R.; Bünten, R.; and Franzke, M.: Automatic Generation of Hexahedral Element Meshes for the Simulation of Metal Forming Processes, Proc. 4th Int. Conference on Numerical Grid Generation, (edited by Weatherill), Swansea, U.K., April 1994, pp. 223-233.

33. Shaw, J.A.; Georgala, J.M.; Peace, A.J.; and Childs, P.N.: The Construction, Application and Interpretation of Three-Dimensional Hybrid Meshes, Proc. 3rd Int. Conference on Numerical Grid Generation, (edited by Arcilla), Barcelona, Spain, June 1991.

34. Shephard, M.S.; and Georges, M.K.: Automatic Three-Dimensional Mesh Generation by the Finite Octree Technique, Int. J. Num. Meth. Eng., Vol. 32, No. 4, 1991, pp. 709-747.

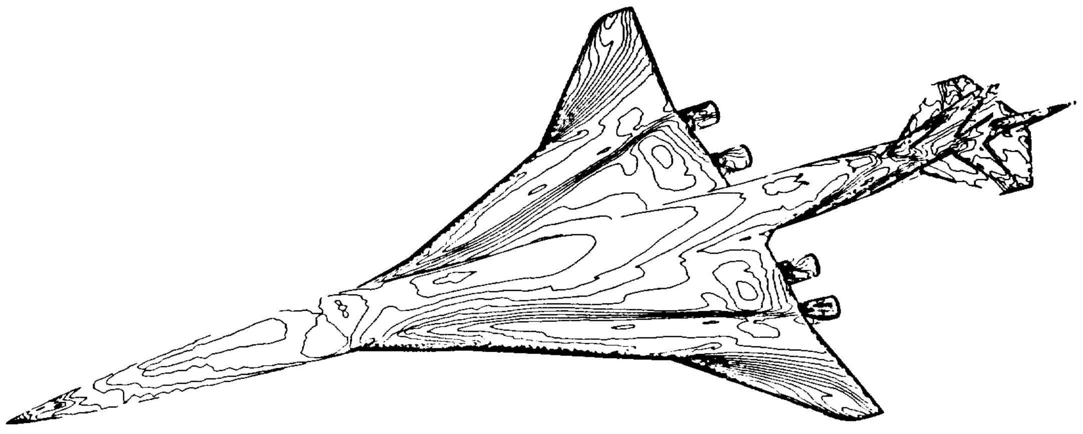
35. Steinbrenner, J.P.; Karman, S.L.; and J.B. Chawner: Generation of Multiple block Grids for Arbitrary 3-D Geometries, in Three Dimensional Grid Generation for Complex Configurations: Recent Progress, (edited by J.L. Steger and J.F. Thompson), 1988, AGARDograph No. 309.

36. Thompson, J.F.: A Composite Grid Generation Code for General 3-D Regions, AIAA 25th Aerospace Sciences Meeting, Reno, NV, January 1987, AIAA Paper 87-0275.

37. Weatherill, N.P.; and Hassan, O.: Efficient Three-Dimensional Grid Generation using the Delaunay Triangulation, Proc. 1st European Comp. Fluid Dynamics Conf., Brussels, September 1992.

38. Yerry, M.A.; and Shephard, M.S.: Automatic Three-Dimensional Mesh Generation by the Modified-Octree Technique, Int. J. Num. Meth. Eng., Vol. 20, 1984, pp. 1965-1990.

39. Zhu, J.Z.; Zienkiewicz, O.C.; Hinton, E.; and Wu, J.: A New Approach to the Development of Automatic Quadrilateral Mesh Generation, Int. J. Num. Meth. Eng., Vol. 32, No. 4, 1991, pp. 849-866.



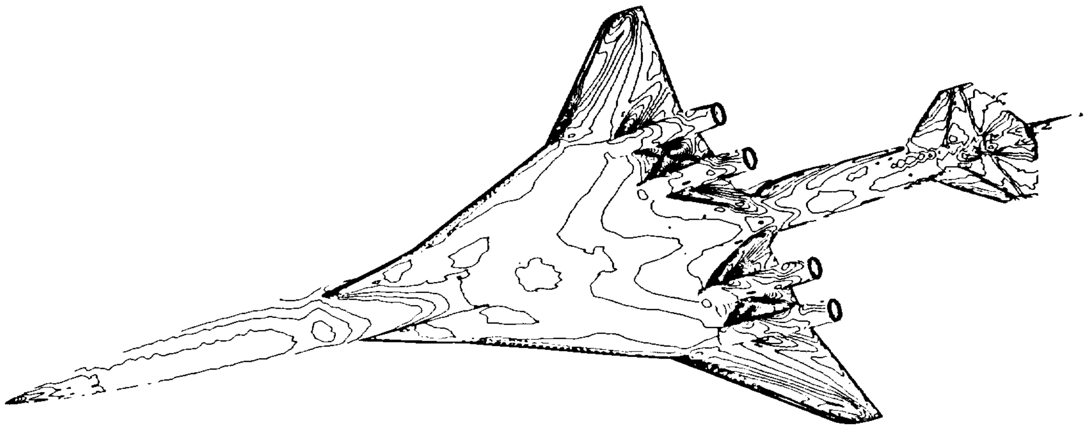


Figure 1.-Surface Pressure Contours of the Flow Solution for a Supersonic Transport, $M_\infty = 2.4$, $\alpha = 3.6^\circ$

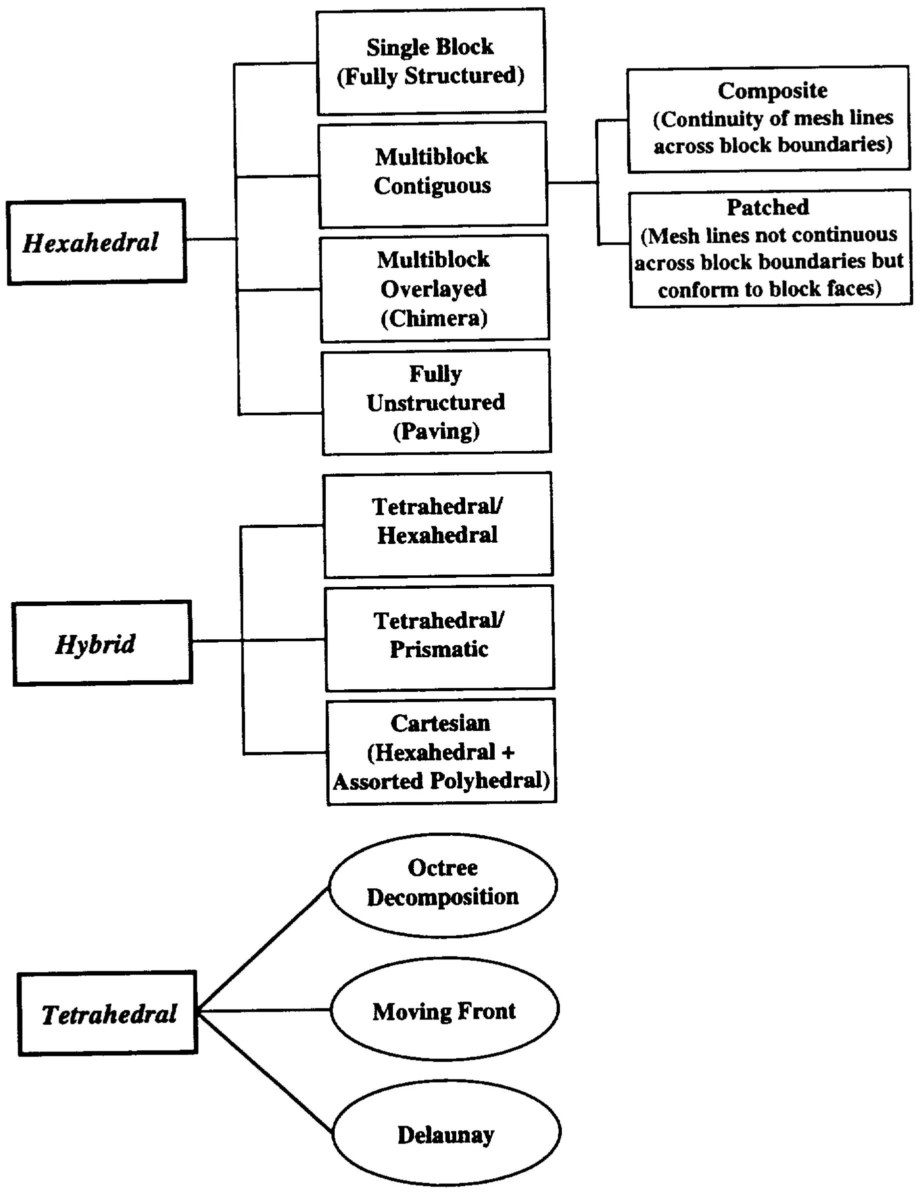

Hexahedral
Single Block (Fully Structured)
Multiblock Contiguous
Composite (Continuity of mesh lines across block boundaries)
Patched (Mesh lines not continuous across block boundaries but conform to block faces)
Multiblock Overlayed (Chimera)
Fully Unstructured (Paving)
Hybrid
Tetrahedral/ Hexahedral
Tetrahedral/ Prismatic
Cartesian (Hexahedral + Assorted Polyhedral)
Tetrahedral
Octree Decomposition
Moving Front
Delaunay

Figure 2.- A Selection of Mesh Types

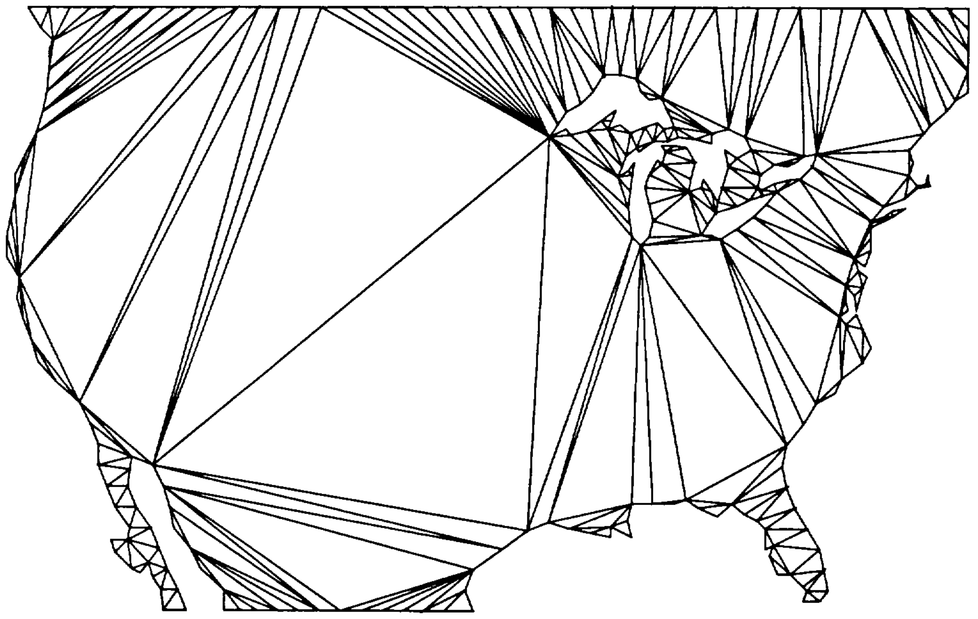


Figure 3a.-Delaunay Triangulation of the Boundary Points

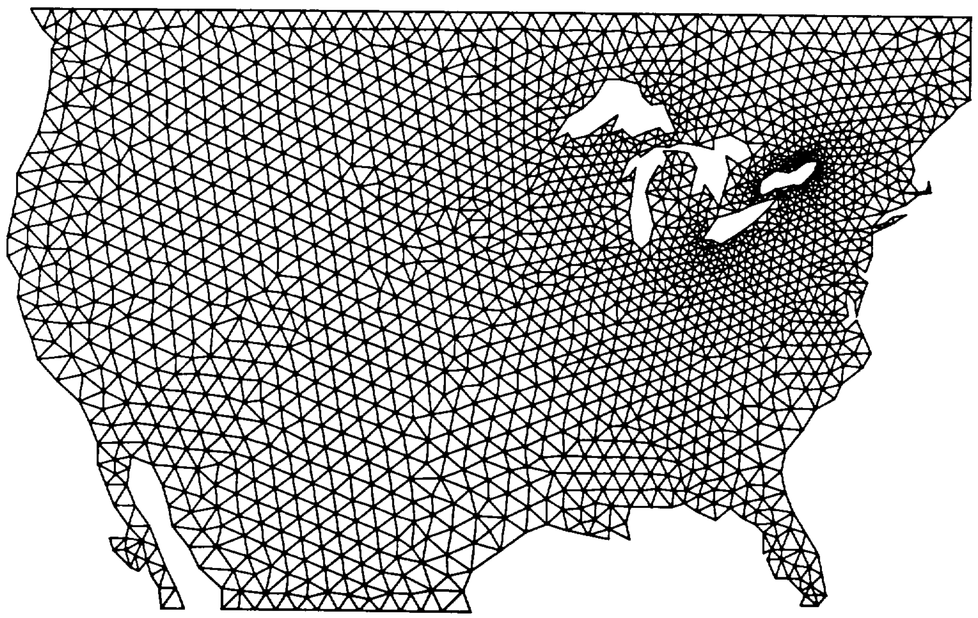


Figure 3b.-Delaunay Triangulation of the Interior of the Domain Using Selective Refinement

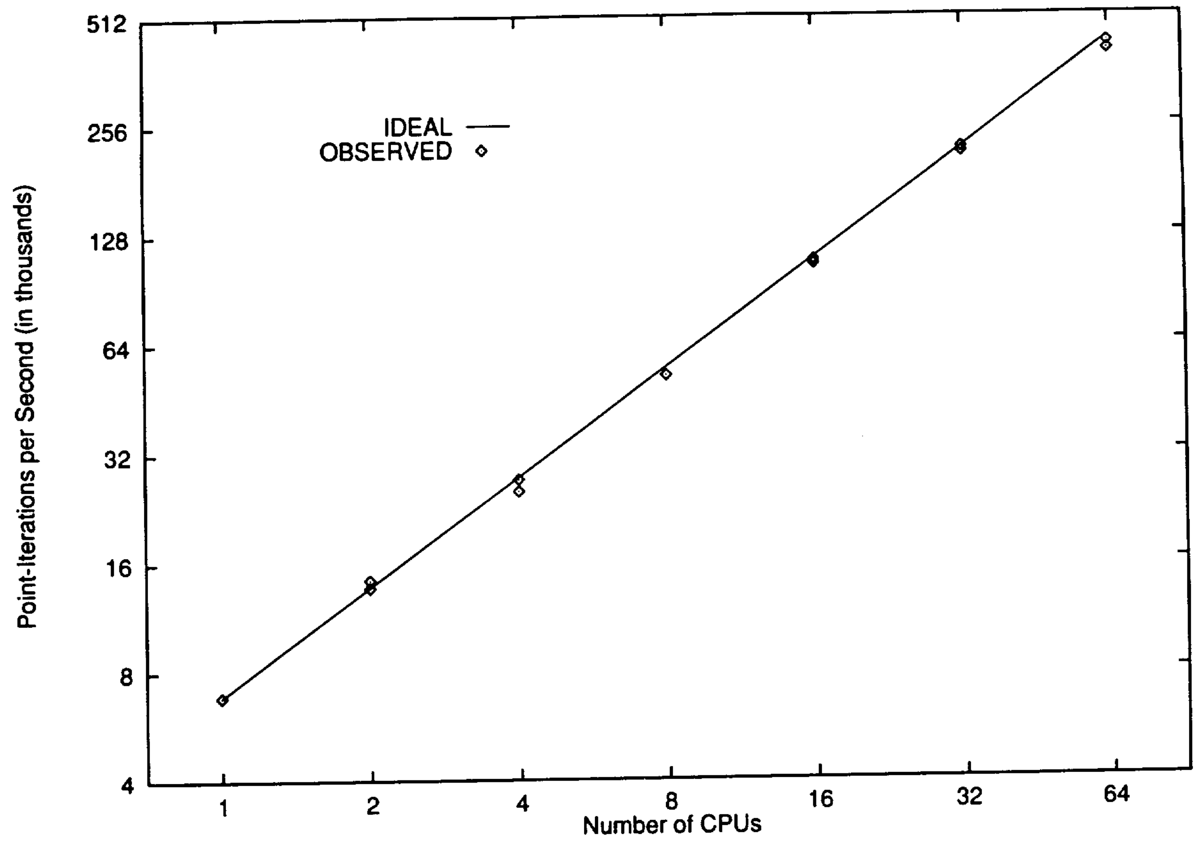


Figure 4.-Parallel Efficiency of an Airplane Code on the IBM SP2 Parallel System

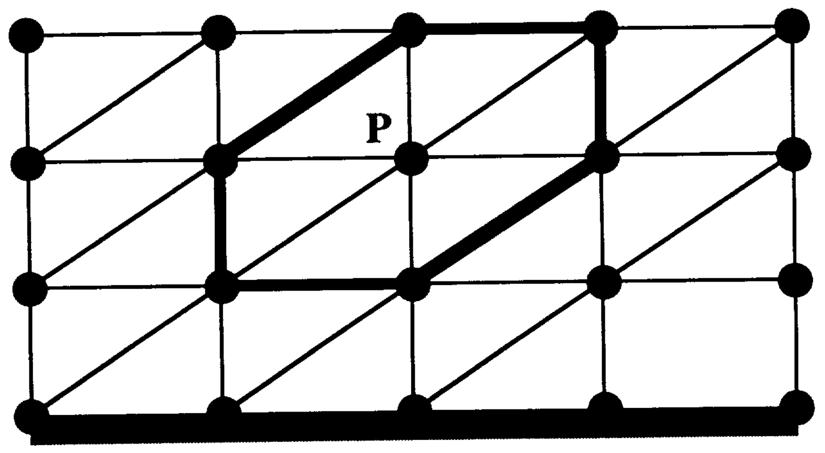


Figure 5.-Control Volume for Point P has Central Symmetry

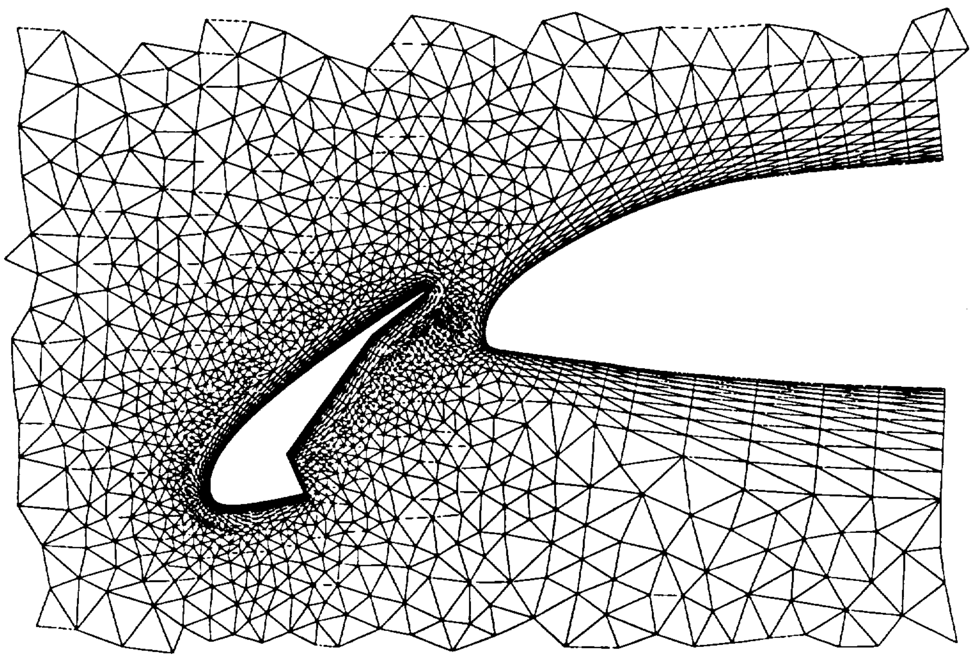


Figure 6.-Triangulation of an Airfoil/Slat Region Suitable for Viscous Computation

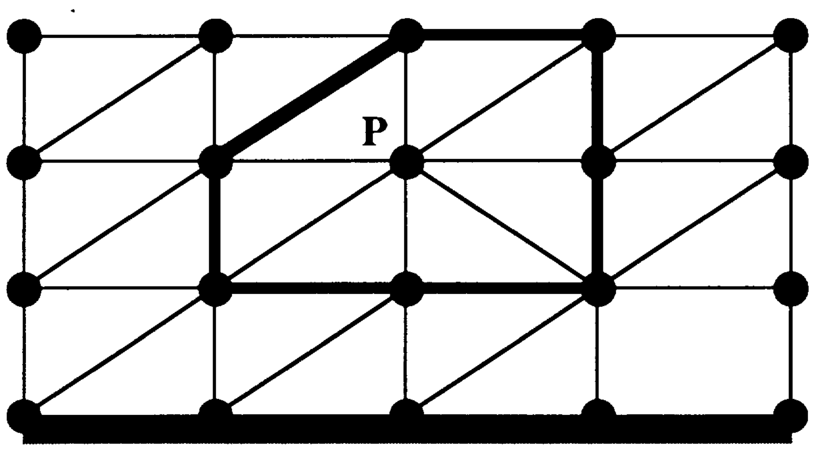


Figure 7.-Control Volume for Point P does not have Central Symmetry