Institute for Mathematics and its Applications
IMA

The Institute for Mathematics and its Applications was established by a grant from the National Science Foundation to the University of Minnesota in 1982. The IMA seeks to encourage the development and study of fresh mathematical concepts and questions of concern to the other sciences by bringing together mathematicians and scientists from diverse fields in an atmosphere that will stimulate discussion and collaboration.

The IMA Volumes are intended to involve the broader scientific community in this process.

Avner Friedman, Director
Willard Miller, Jr., Associate Director

****

IMA ANNUAL PROGRAMS

1982–1983 Statistical and Continuum Approaches to Phase Transition
1983–1984 Mathematical Models for the Economics of Decentralized Resource Allocation
1984–1985 Continuum Physics and Partial Differential Equations
1985–1986 Stochastic Differential Equations and Their Applications
1986–1987 Scientific Computation
1987–1988 Applied Combinatorics
1988–1989 Nonlinear Waves
1989–1990 Dynamical Systems and Their Applications
1990–1991 Phase Transitions and Free Boundaries
1991–1992 Applied Linear Algebra
1992–1993 Control Theory and its Applications
1993–1994 Emerging Applications of Probability
1994–1995 Waves and Scattering
1995–1996 Mathematical Methods in Material Science

IMA SUMMER PROGRAMS

1987 Robotics
1988 Signal Processing
1989 Robustness, Diagnostics, Computing and Graphics in Statistics
1990 Radar and Sonar (June 18 - June 29)
New Directions in Time Series Analysis (July 2 - July 27)
1991 Semiconductors
1992 Environmental Studies: Mathematical, Computational, and Statistical Analysis
1993 Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations
1994 Molecular Biology

****

SPRINGER LECTURE NOTES FROM THE IMA:

The Mathematics and Physics of Disordered Media
Editors: Barry Hughes and Barry Ninham
(Lecture Notes in Math., Volume 1035, 1983)
Orienting Polymers
Editor: J.L. Ericksen
(Lecture Notes in Math., Volume 1063, 1984)
New Perspectives in Thermodynamics
Editor: James Serrin
(Springer-Verlag, 1986)
Models of Economic Dynamics
Editor: Hugo Sonnenschein
(Lecture Notes in Econ., Volume 264, 1986)

Ivo Babuska
William D. Henshaw
Joseph E. Oliger
Joseph E. Flaherty
John E. Hopcroft
Tayfun Tezduyar

Editors

Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations

With 201 Illustrations

Springer-Verlag
New York Berlin Heidelberg London Paris
Tokyo Hong Kong Barcelona Budapest

Ivo Babuska
Institute for Physical Science
and Technology
University of Maryland
College Park, MD 20742 USA

Joseph E. Flaherty
Department of Computer Science and
Scientific Computation Research Ctr.
Rensselaer Polytechnic Institute
110-8th St.
Troy, NY 12180 USA

William D. Henshaw
CIC-3 MS K987
Los Alamos National Laboratory
Los Alamos, NM 87545 USA

Joseph E. Hopcroft
Joseph Silbert Dean of Engineering
Cornell University
College of Engineering
242 Carpenter Hall
Ithaca, NY 14853 USA

Joseph Oliger
Research Institute for
Advanced Computer Science
Mail Stop T20G-5
NASA Ames Research Center
Moffet Field, CA 94035 USA

Tayfun Tezduyar
Army High Performance
Computing Research Center
1100 South Washington Ave.
Suite 101
Minneapolis, MN 55415 USA

Series Editors

Avner Friedman
Willard Miller, Jr.
Institute for Mathematics and its Applications
University of Minnesota
Minneapolis, MN 55455 USA

Mathematics Subject Classifications (1991): 65M06, 65M12, 65M15, 65M20, 65M50, 65M55, 65M60, 65M70, 65N06, 65N12, 65N15, 65N22, 65N30, 65N35, 65N50, 65N55, 65Y05, 65Y10, 68Q22, 68T05, 68U05, 68U07, 73B40, 73E99, 73K20, 76D05, 76R10, 76Z05, 76S05, 92C35

Library of Congress Cataloging-in-Publication Data
Modeling, mesh generation, and adaptive numerical methods for partial differential equations / [edited by] Ivo Babuska . . . [et al.].
p. cm. — (The IMA volumes in mathematics and its applications ; v. 75)
"Based on the proceedings of the 1993 IMA summer program"--Foreword.
Includes bibliographical references.
ISBN 0-387-94542-3 (alk. paper)
1. Differential equations, Partial—Numerical solutions--Congresses. 2. Numerical grid generation (Numerical analysis)--Congresses. 1. Babuška, Ivo. II. Series.
QA377.M578 1995 95-17342
515'.353—dc20

CIP data applied for.

Printed on acid-free paper

ISBN 0-387-94542-3/1995 $5.00 + $0.20

Production managed by Laura Carlson; manufacturing coordinated by Jeffrey Taub.
Camera-ready copy prepared by the IMA.
Printed and bound by Braun-Brumfield, Ann Arbor, MI.
Printed in the United States of America

9 8 7 6 5 4 3 2 1

ISBN 0-387-94542-3 Springer-Verlag New York Berlin Heidelberg

The IMA Volumes
in Mathematics and its Applications

Current Volumes:

Volume 1: Homogenization and Effective Moduli of Materials and Media
Editors: Jerry Ericksen, David Kinderlehrer, Robert Kohn, and J.-L. Lions

Volume 2: Oscillation Theory, Computation, and Methods of Compensated Compactness
Editors: Constantine Dafermos, Jerry Ericksen, David Kinderlehrer, and Marshall Slemrod

Volume 3: Metastability and Incompletely Posed Problems
Editors: Stuart Antman, Jerry Ericksen, David Kinderlehrer, and Ingo Muller

Volume 4: Dynamical Problems in Continuum Physics
Editors: Jerry Bona, Constantine Dafermos, Jerry Ericksen, and David Kinderlehrer

Volume 5: Theory and Applications of Liquid Crystals
Editors: Jerry Ericksen and David Kinderlehrer

Volume 6: Amorphous Polymers and Non-Newtonian Fluids
Editors: Constantine Dafermos, Jerry Ericksen, and David Kinderlehrer

Volume 7: Random Media
Editor: George Papanicolaou

Volume 8: Percolation Theory and Ergodic Theory of Infinite Particle Systems
Editor: Harry Kesten

Volume 9: Hydrodynamic Behavior and Interacting Particle Systems
Editor: George Papanicolaou

Volume 10: Stochastic Differential Systems, Stochastic Control Theory, and Applications
Editors: Wendell Fleming and Pierre-Louis Lions

Volume 11: Numerical Simulation in Oil Recovery
Editor: Mary Fanett Wheeler

AUTOMATIC MESHING OF CURVED THREE—DIMENSIONAL DOMAINS: CURVING FINITE ELEMENTS AND CURVATURE-BASED MESH CONTROL

MARK S. SHEPHARD*, SAIKAT DEY* , AND MARCEL K. GEORGES*

Abstract. Specific issues associated with the automatic generation of finite element meshes for curved geometric domains are considered. A review of the definition of when a triangulation is a valid mesh, a geometric triangulation, for curved geometric domains is given. Consideration is then given to the additional operations necessary to maintain the validity of a mesh when curved finite elements are employed. A procedure to control the mesh gradations based on the curvature of the geometric model faces is also given.

Nomenclature.

- G Refers to the geometric model, or, when used as a left subscript, to indicate one or more entities associated with the geometric model
- M Refers to the mesh, or, when used as a left subscript, to indicate one or more mesh entities
- ${}_\gamma T$ Set of all topological entities associated with model $\gamma, \gamma = G$ or M
- ${}_\gamma S$ The shape information associated with the model $\gamma, \gamma = G$ or M
- ${}_\gamma T_i^d$ Topological entity i from model γ of dimension $d, d = 0$ is a vertex which represents a point in space, $d = 1$ is an edge which represents a 1-D locus of points, $d = 2$ is a face which represents a 2-D locus of points, $d = 3$ is a region which represents a 3-D locus of points (note - no right subscript indicates the set of all topological entities of dimension d)
- $\partial({}_\gamma T_i^d)$ Boundary of topological entity ${}_\gamma T_i^d$, $\gamma = G$ or M
- $\overline{{}_\gamma T_i^d}$ Closure of topological entity defined as $({}_\gamma T_i^d \cup \partial({}_\gamma T_i^d))$, $\gamma = G$ or M
- ${}_G\Lambda^n$ n-dimensional geometric triangulation of the geometric model G
- s_i^d d-dimensional element i in ${}_G\Lambda^n$
- $\sqsubset$ Classification symbol used to indicate the association of one or more entities from one model, typically M or D with a higher model, typically D or G
- $\sqcap^*$ Parametric intersection operator used to signify the application of an intersection operation of two ${}_M T_i^{d-1} \sqsubset {}_G T_k^d$

1. Introduction. The ability to develop reliable procedures that can automatically discretize arbitrary curved three-dimensional domains into valid finite element discretizations is hampered by the lack of knowledge of fundamental properties upon which to base the discretization procedure.

* Scientific Computation Research Center Rensselaer Polytechnic Institute Troy, NY 12180–3590.

Much of the current emphasis is correctly focused on the determination of the basic properties associated with the triangulation of polygonal domains. On the other hand, the demand to be able to automatically generate meshes for curved domains, as defined in solid modeling systems, pushes one to consider what can be developed to deal with these cases, even though there is some level of uncertainty of properties of the procedures. This paper discusses some specific aspects of our work on automatic three-dimensional mesh generation for curved domains.

The importance of the reliability of an automatic mesh generator to the reliability of a finite element analysis process becomes evident from a consideration of the definition of an automatic mesh generator:

Definition: Automatic mesh generator — An algorithmic procedure which can create, under program control and without user input or intervention, a valid mesh, a Geometric Triangulation, ${}_G\Lambda^n$, for geometric models, G, of arbitrary complexity.

If an automatic mesh generator is not reliable, invalid meshes can be generated. Invalid meshes lead an automated analysis process to solve the wrong problem, thus eliminating the reliability of the entire process. To address the issue of what constitutes a valid finite element mesh for curved three-dimensional domains, the next section provides a definition of a geometric triangulation which represents such a valid mesh. This definition, reviewed in section 2, has been used as the basis for procedures to mesh curved domains with straight-edge, planar-faced finite elements [12,15].

The application of higher order finite element methods to curved geometries requires consideration of finite elements which are also curved. Typically the curved finite elements are limited to those which have edges and faces on the curved boundaries of the model. The geometric shape of the curved finite elements range from simple quadratics through points on the model geometry, to that of the surface geometry itself. To maintain the computational efficiency of the meshing process, meshes including those to possess curved finite elements, are often generated by first generating the straight-sided finite elements and then curving those edges on the model faces. In many cases meshes that are valid with respect to the straight-edged geometric approximation become either invalid due to the overlap of finite elements, or unacceptable because the variations of the Jacobian within the element are too large. The third section of this paper discusses a set of local mesh modification procedures to correct such situations yielding valid and acceptable curved finite element meshes.

Another aspect of finite element mesh generation for curved geometric domains is control of the element gradations. One a priori mesh control device many users like is the ability to make the mesh finer in areas where the model is highly curved. The fourth section discusses a curvature-based mesh control procedure which meets this need.

2. Review of requirements of a valid mesh. Efforts on the development of automatic three-dimensional mesh generators have been underway for at least a decade. The developers of these procedures have often been frustrated by the inability to ensure the reliability of the procedures and to qualify exactly the conditions which would cause failure of the procedure. This section reviews a general definition of what constitutes a valid mesh and indicates how its application leads to a general algorithm to convert a triangulation of a domain into a geometric triangulation which represents a valid finite element mesh [13,10,11,14].

Background

Since mesh generation is concerned with the decomposition of a geometric domain into a union of simple, non-overlapping geometric entities, the definition of a valid mesh must be in terms of the definition of the geometric domain. The definition of a geometric domain can be considered to consist of two sets of information

$$G = \{{}_GS,\ {}_GT\} \tag{2.1}$$

where ${}_GS$ represents the geometric information defining the shape of the entities which define the domain and ${}_GT$ represents the topological types and associativities of the entities. Since individual finite elements are assumed to be a simple region bounded by simply connected faces, the topological entities associated with the 0 to n dimensional geometric entities are of interest. For the three-dimensional case ($n = 3$)

$${}_GT = \{{}_GT^0,\ {}_GT^1,\ {}_GT^2,\ {}_GT^3\} \tag{2.2}$$

where ${}_GT^d$, $d = 0, 1, 2, 3$ are respectively the set of vertices, edges, faces and regions defining the primary topological elements of the geometric domain.

Critical to the definition of a valid finite element mesh are the concepts of mesh classification and mesh compatibility [10,11,13].

Definition: Classification — The unique association of a topological mesh entity of dimension d_i, ${}_MT_i^d$, to a topological model entity of dimension d_j, ${}_GT_j^{d_j}$, where $d_i \leq d_j$, is termed classification and is denoted

$${}_MT_k^{d_i} \sqsubset {}_GT_j^{d_j} \tag{2.3}$$

where the classification symbol, $\sqsubset$, indicates that the left hand entity or set is classified on the right hand entity.

Multiple ${}_MT_i^{d_i}$ can be classified on a ${}_GT_j^{d_j}$.

Definition: Topological Compatibility — Given a non-self-intersecting mesh with all vertices in the vertex set ${}_MT^0$ classified, and the remaining sets of mesh entities ${}_MT^d$, $1 \leq d \leq n$ with boundary entity sets

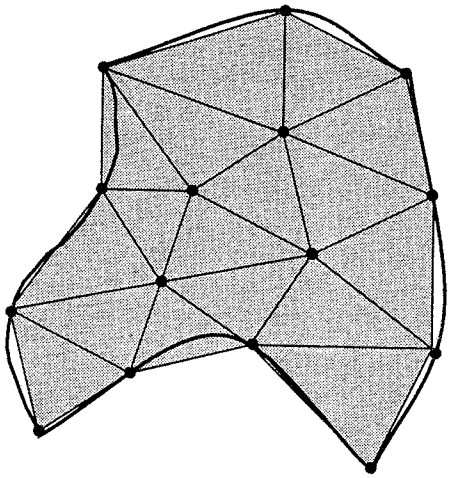


FIG. 2.1. *Topologically compatible mesh*

$\partial({}_M T^d)$, *consider a model entity* ${}_G T_j^d$ *with boundary entities* $\partial({}_G T_j^d$. *If each* $\partial({}_M T_i^d) \sqsubset {}_G T_j^d$ *is used by two* ${}_M T_i^d \sqsubset {}_G T_j^d$, *and each* $\partial({}_M T_i^d) \sqsubset \partial({}_G T_j^d)$ *is used by one* ${}_M T_k^d \sqsubset {}_G T_j^d$, *then the mesh is topologically compatible with the topological entity* ${}_G T_j^d$. *A mesh is topologically compatible if it is compatible with all topological entities.*

Consider Figures 2.1 through 2.3 for a clarification of topological compatibility. Figure 2.1 shows a ${}_G T_j^2$ covered by a compatible set of ${}_M T_k^2 \sqsubset {}_G T_j^2$. In this case all the ${}_M T_k^1 \sqsubset {}_G T_j^2$ are used by two ${}_M T_k^2 \sqsubset {}_G T_j^2$ and all ${}_M T_k^1 \sqsubset \partial({}_G T_j^2)$ are used by one ${}_M T_k^2 \sqsubset {}_G T_j^2$. Figure 2.2 contains a topological hole characterized by the fact that the three mesh edges $({}_M T_1^1,\ {}_M T_2^1,\ {}_M T_3^1) \sqsubset {}_G T_j^2$ are used by only one ${}_M T_k^2 \sqsubset {}_G T_j^2$. Figure 2.3 depicts a topological redundancy which is characterized by the four mesh edges $({}_M T_1^1,\ {}_M T_2^1,\ {}_M T_3^1,\ {}_M T_4^1) \sqsubset {}_G T_j^2$ each being used by three ${}_M T_k^2 \sqsubset {}_G T_j^2$.

Geometric triangulation

Starting with these definitions, a definition of a valid finite element mesh can be given.

Definition: Geometric Triangulation — Given a set P *of* M *unique points, each classified with respect to the geometry* G, *an* n*-dimensional geometric triangulation,* ${}_G\Lambda^n$ *is a set of* N *nondegenerate elements* $s_i^{d_i}$

$$ {}_G\Lambda^n = \left\{ s_1^{d_1}, s_2^{d_2}, s_3^{d_3}, \ldots, s_N^{d_N} \right\} \tag{2.4} $$

with $0 \le d_i \le n$, *satisfying the following properties:*

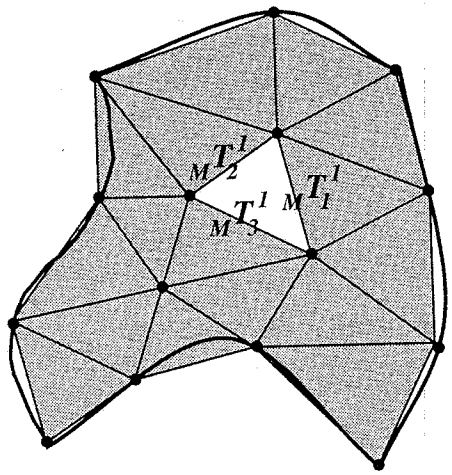


FIG. 2.2. *Topological hole*

i. All vertices of each $s_i^{d_i} \in P$

ii. For each $i \neq j$, *interior* $\left\{ s_i^{d_i} \right\} \cap$ *interior* $\left\{ s_j^{d_j} \right\} = 0$

iii. ${}_G\Lambda^n$ *is topologically compatible with* G

iv. ${}_G\Lambda^n$ *is geometrically similar to* G

The simplest explanation of a geometrically similar mesh is one that in the limit of refinement will exactly match the geometry of the domain. This simple definition is not a workable one for the development of algorithms to evaluate and correct mesh validity. In all but complex geometric cases, this requirement is satisfied if topological compatibility is satisfied. However, since there is no a priori method to ensure that topological compatibility alone will also ensure geometric similarity, it must be explicitly considered.

One method of ensuring geometric similarity introduced by Schroeder [10] employs the concept of parametric intersection. Any application of this approach requires that each of the geometric entities in the geometric model be uniquely mappable.

Definition: Uniquely Mappable [10] — *A geometric entity of dimension* d, S^d, *is uniquely mappable if for each point* $p \in S^d$, *there exist a function* $f : S^d \to H^d$, *that satisfies the following conditions:*

1. *For each neighborhood* V *of* $f(p)$, *there exist a neighborhood* U *of* p *such that* $f(U) \subset V$
2. *For each* $p \neq \acute{p}$, $f(p) \neq f(\acute{p})$
3. *Each* S^d *is mappable to the hyperplane* H^d

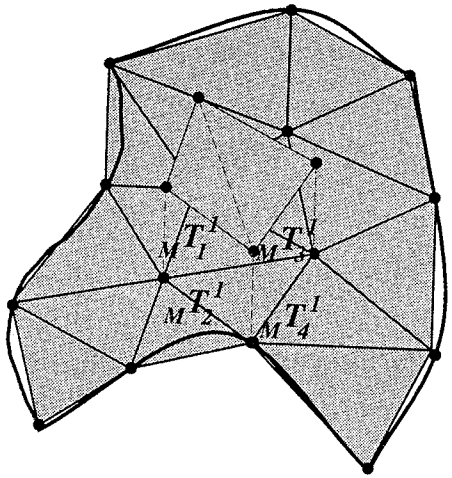


FIG. 2.3. *Topological redundancy*

4. Each S^d is of finite extent

The property of unique mappability allows the introduction of a parameterization of the individual geometric entities. The development of a practical algorithm does not require the parameterization to be explicitly defined over the entire entity. Instead it can be defined in local neighborhoods large enough to perform parametric intersections of the mesh entities under consideration.

Definition: Parametric Intersection [10] — *Given two mesh entities of order d, $_MT_i^d$ and $_MT_j^d$, classified on a topological model entity of dimension d,$_GT_k^d$, the parametric intersection of $_MT_i^d$ and $_MT_j^d$ is written as:*

$$_MT_i^d \sqcap^* \ _MT_j^d \tag{2.5}$$

With the concept of a parametric intersection the conditions of geometric similarity can be given.

Definition: Geometric Similarity [10] — A set of mesh entities of order d,$_MT^d$, is geometrically similar to a topological model entity of order d,$_GT_k^d$, when $_MT^d$ consist of N mesh entities of order d

$$_MT^d = \bigcup_{i=1}^{N} {}_MT_i^d \tag{2.6}$$

where each mesh entity $_MT_i^d$ is classified on the topological model entity $_GT_k^d$ as:

$$_MT_i^d \sqsubset \ _GT_k^d \ , \ \forall \ i = 1, \ldots, N \tag{2.7}$$

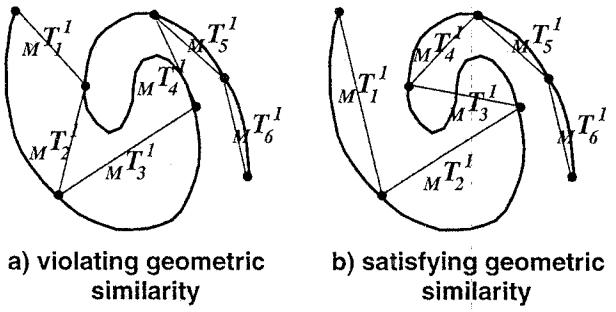


FIG. 2.4. *Geometric similarity on a model edge*

and the parametric intersection of any two $_MT_i^d \in \ _MT^d$ is:

$$_MT_i^d \sqcap^* \ _MT_j^d = 0 \tag{2.8}$$

Figure 2.4 demonstrates the concept of geometric similarity for a $_GT_j^1$. The mesh entities in 2.4a do not satisfy the geometric similarity conditions because the mesh edge $_MT_1^1$ overlaps mesh edges $_MT_2^1$ and $_MT_3^1$ in the parametric space of the model edge. The set of mesh edges in Figure 2.4b do satisfy the geometric similarity requirements. In the case of model edges the determination of satisfaction of geometric similarity is straight forward. However, the algorithms required for the determination of geometric similarity for model faces are more complex [10].

Assurance algorithm

The outline of a general assurance algorithm that can operate from a triangulation of a set of properly classified points which encompasses the convex hull of the domain being meshed is [10]:

1. Initial classification based on necessary conditions. The necessary conditions used to classify a mesh entity are based on the classification of its boundary entities. Given a model entity of dimension n, $_GT_i^n$, the set of mesh entities $_MT_j^m \sqsubset \ _GT_i^n, m \leq n$ initially classified on the model entity, H, is given by

$$H = \{ \ _MT_j^m | \partial(_MT_j^m) \sqsubset \overline{_GT_i^n} \} \tag{2.9}$$

Initial classification must be done in increasing topological order from mesh edges. An important property of the initial classification process is that all mesh entities which can be classified on an entity are identified as classified on that entity or its boundary. Therefore,

later steps in the assurance algorithm will not need to look for additional candidates.

2. Edge compatibility assurance by traversal. Employing the idea of an edge parameter, the mesh vertices classified on the model edge can be sorted in order from the start to the end. The mesh is compatible and geometrically similar if there is a single mesh edge connecting each pair of mesh vertices on the mesh edge and these edges do not intersect themselves. Mesh edges that connect to other than consecutive mesh vertices are redundant and are corrected through reclassification. If two consecutive mesh vertices are not connected by a mesh edge, a hole exist. Holes are corrected by either the creation of the correct connection or the insertion of additional points along the model edge between the mesh vertices bounding the hole followed by local re-triangulation.
3. Face compatibility by recursive boundary classification. Given the loop(s) of mesh edges bounding a model face mesh, take a mesh face that uses one or more of the bounding mesh edges once and mark it as compatible with the face. For the mesh face under consideration, remove the ${}_MT_j^1 \sqsubset \partial({}_GT_i^2)$ from the loop and insert the ${}_MT_j^1 \not\sqsubset \partial({}_GT_i^2)$ into the updated boundary. Continue this process until there are no edges remaining in the loop. If the process terminates before all edges are removed from the loop an incompatibility, in terms of either a redundancy or hole exist. Redundancies are removed by the proper reclassification. Holes are corrected by either the creation of the correct connections or the insertion of points on the face in the area of the hole and local re-triangulation [10,13]. Geometric similarity can be checked during this process through local surface parameterizations.
4. Region compatibility by inheritance. Once the mesh is compatible with the model faces all model regions will be completely bounded by valid sets of mesh faces. Starting with a single mesh region in a region, all its unclassified boundary entities inherit that region classification as does any neighboring region sharing a mesh face not classified on the boundary of the model region.

The interested reader is referred to [10,13,14] for more detail.

3. Curved finite element generation. It is common in a variety of applications to employ higher order shape functions over the individual elements to describe the behavior of the primary unknowns. The introduction of hierarchic p-version finite elements [17] have made the use of high order polynomials practical. Since higher order elements provide increased approximation power over individual elements, coarser meshes are required. Therefore, the geometric approximation introduced by the use of straight-edged finite elements can become a major contributor to the total

solution error. The geometric approximation error can be greatly reduced or even eliminated by increasing the order of the geometric shape functions from the lines associated with the straight-edged elements to shape functions which can either exactly represent the curved geometric domain, or provide a better approximation to it. The commonly used isoparametric finite element concept [18] uses the same shape functions for both the behavior functions and geometry of the element. In addition, integration to the exact or highly accurate approximate geometry has been supported through the use of blending functions [17].

A problem that can arise when finite element entities classified on curved model entities are curved to that boundary is that the requirements of a geometric triangulation can be violated and/or the shape of the element becomes so poor that numerical stiffening due to large determinants of the Jacobian variations within the element will result. In either case the elements are deemed unacceptable and corrective actions must be carried out. Assuming that leaving the edges of the problem finite elements straight is not acceptable, corrective measures which modify the finite element mesh are required. The key to determining the type of corrective action required is the determination of the cause of the unacceptability of the element.

The effect of curving mesh edges is often not limited to one element since it is typically shared by other elements in the neighborhood. Thus an unacceptable element cannot always be corrected without looking into the neighboring mesh entities which are affected due to the curving. As one or more of the edges and/or faces of an element is curved, it can cause two situations leading to the element becoming unacceptable. First, unacceptably curved elements created as a result of other mesh entities coming in proximity of the curved entities of the element are depicted in Figures 3.1a through 3.1d. Second, unacceptably curved elements can arise when entities are too far from their linear approximation as depicted in Figures 3.1e and 3.1f. For the 2D case depicted in Figure 3.1a curving of edge ${}_MT_1^1$ makes element ${}_MT_1^2$ unacceptable. The problem is not with the curved edge ${}_MT_1^1$, but with the fact that it intersects the edges ${}_MT_2^1$ and ${}_MT_3^1$ at points other than at the common vertices. In other words, the curved entities of element ${}_MT_1^2$ penetrate into the neighboring elements making it unacceptable. In 3D, bounding mesh edges, as well as faces, of an element will curve. If as a result of these curvings the edges or faces intersect other mesh entities then the element will become unacceptable. Figure 3.1b shows a case where a curved face ${}_MT_1^2$ intersects edge ${}_MT_1^1$ at points other than their common boundary implying that the curved face ${}_MT_1^2$ will penetrate one or more neighboring elements which share edge ${}_MT_1^1$.

In the cases where mesh entities intersect or are in close proximity to the curved finite element entities the goal of any mesh modification operation is to properly modify or eliminate the entity that intersects or

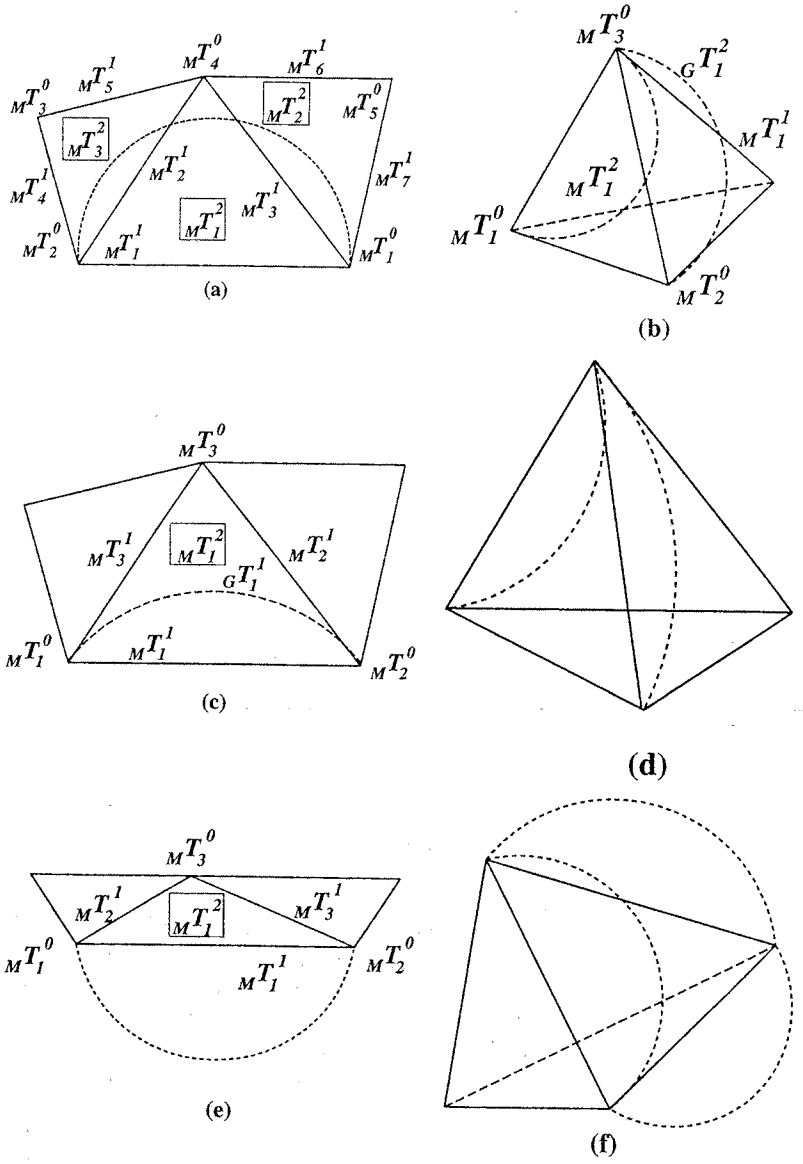


FIG. 3.1. *Unacceptable curvings in 2D and 3D, Figures 3.1a and 3.1b have penetration into neighboring entities, Figures 3.1c and 3.1d have entities "too close" to neighboring entities, and Figures 3.1e and 3.1f have entities that curve too far from linear representation.*

is too close to the curved entity. This point is clear in the case where the curved finite element entities exactly match the curved geometry of the model. Unless the entity that is intersecting or is too close to the curved entity is modified or eliminated, they will continue to be too close and always lead to unacceptable elements.

The steps involved with the development of a procedure for producing acceptable curved elements consists of the following steps:

1. Identification of the unacceptable elements.
2. Identification of the mesh entities that must be modified or eliminated.
3. Determination and execution of corrective actions necessary to modify or eliminate the problem mesh entities through local mesh modifications.

The next subsection discusses the metrics used to determine unacceptable elements and the mesh entities causing the elements to be unacceptable. The following subsection presents an incremental approach for eliminating the problem entities through a hierarchy of local mesh modifications. The last two subsections provide some specifics of the implementation of the mesh modification procedures and present the results obtained with those procedures implemented to date.

Determination of unacceptable elements and the mesh entities to be modified or eliminated

As indicated previously, elements become unacceptable when curved either because they self intersect, causing the triangulation of the mesh to violate the definition of a geometric triangulation, or the element shape becomes so poor that it will lead to numerical stiffening due to large variations in the determinant of the Jacobian. The two causes of unacceptable elements immediately lead to consideration of two metrics to determine unacceptable elements, intersection calculations and variations in the determinant of the Jacobian.

The use of intersection calculations has two advantages. The first is that determination of the intersection directly indicates which mesh entities of the current element must be modified or eliminated. The second advantage is its ability to also determine problem mesh entities of other elements in the neighborhood. To see this, consider the 2D case shown in Figure 3.2 where the curved edge of element ${}_MT_1^2$ intersects mesh entities in elements ${}_MT_1^2$ through ${}_MT_4^2$. Clearly the mesh modifications will need to propagate into the mesh far enough to modify or eliminate all the intersected mesh edges. There are, however, two disadvantages to intersection calculations. The most critical is intersections only identify the invalid element situations, they do not identify the elements that are valid but are unacceptably shaped. The second disadvantage is the computational

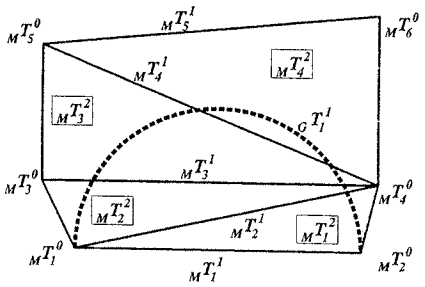


FIG. 3.2. *Interior elements affected due to curving of boundary entity*

expense. It is possible to supplement the intersection test with a closest distance check to determine the situations where the mesh entities come too close to the curved entity. The addition of such calculations greatly increases the computational cost past the already computationally expensive intersection calculations.

The second metric of evaluating the variation and minimum value of the determinant of the Jacobian can identify both invalid and poorly shaped elements. Since it considers only the influence of the mesh entities of the element itself, this procedure will not identify additional problems with neighboring elements in the way a set of intersection checks can. Again consider Figure 3.2 where entities belonging to elements $_MT_2^2$ through $_MT_4^3$ intersect the curved edge of element $_MT_1^2$. Since all the edges of elements $_MT_2^2$ through $_MT_4^2$ are straight and the elements have positive area, the Jacobian is constant through the element and positive. Other potential drawbacks of the determinant of the Jacobian variation evaluation is the ability to identify the problem entities in the element and the computational effort required to determine the location of the maximum and minimum determinants of the Jacobian. As discussed in the subsections that follow, the use of an incremental approach focused on elimination of the problem mesh entities is capable of incrementally determining all the entities in the neighborhood of the curved mesh entities for eventual elimination. The use of only a limited number of pointwise determinants of the Jacobian evaluations can greatly reduce the computational cost, but does introduce some level of approximation into the process. This approximation is typically acceptable since the common method used by analysis codes to determine unacceptable elements is to examine the positiveness of, and maximum difference between, the determinants of the Jacobian evaluated at the numerical integration points used in the calculation of the stiffness matrix. In addition, pointwise determinants of the Jacobian evaluations should be able to identify the problem mesh entities which must be modified or eliminated.

A curved finite element is valid with respect to self intersection if the determinant of the Jacobian of the element mapping is positive at all points within the element. If $\mathbf{X}(\xi)$ defines the geometric mapping of the element where ξ represents the natural coordinates of the element then the Jacobian [7,18] of the mapping is given by

$$\mathbf{J}(\xi) = \frac{\partial \mathbf{X}}{\partial \xi} \tag{3.1}$$

and a valid element has

$$\det(\mathbf{J}(\xi)) > 0 \quad \forall \quad \xi \tag{3.2}$$

Numerical stiffening due to large variations in the determinant of the Jacobian variation is related to the numerical integration of the stiffness matrix over the element. Consider the most basic integral of evaluating the volume of a tetrahedral finite element with geometry shape functions written in volume coordinates $\{\xi_1, \xi_2, \xi_3, \xi_4\}$. The exact volume, V_{exact}, of the element is given by

$$V_{\text{exact}} = \frac{1}{6} \int \int_{\Omega^e} \int \det(\mathbf{J}(\xi)) d\xi_1 d\xi_2 d\xi_3 \tag{3.3}$$

When numerical integration is used as an approximate volume, V, is calculated as

$$V_{\text{exact}} \simeq V = \frac{1}{6} \sum_{i=1}^{N_{int}} \det(\mathbf{J}(\xi^i)) w^i \tag{3.4}$$

where N_{int} is the number of integration points, ξ^i and w^i are the integration point coordinates and the corresponding weights respectively. Based on this a common measure of element shape distortion is given by [9]

$$I = \frac{N_{int} \min_i (\det(\mathbf{J}(\xi^i)) w^i)}{V} \tag{3.5}$$

An element is considered acceptable when $\min_i(\det(\mathbf{J}(\xi^i))w^i) > 0$ and $I > I_{\min} > 0$.

Since other expressions can equally be used as a metric of the element acceptability, care is taken in the implementation to allow the details of the metric to be changed by changing one procedure.

Although equation (3.5) indicates the unacceptability of an element, it does not indicate which of the mesh entities need to be modified or eliminated. However, the location of negative, or small, relative to the other, determinants of the Jacobian should to able to identify the problem

entities. For example, evaluation of the determinant of the Jacobian along the specific entities should indicate which are problems. Current studies are underway to more carefully qualify the use of this type of information.

Incremental approach to eliminate problem entities

Given a list of unacceptable finite elements, and an indication for each such element which mesh entities must be modified or eliminated to potentially produce acceptable elements, a variety of approaches to the development of the mesh improvement procedures are possible. Since the number of unacceptable elements is typically small, it is appropriate to focus attention on local mesh modifications to produce the acceptable elements. This has the advantage that the individual components of the procedure are based on well qualified operations. However, the overall procedure is still based on a heuristic combination of these operations due to the large number of constraints on the operations possible in any specific circumstance, and the tendency for propagation of unacceptable elements past the current element (see the simple 2–D example of Figure 3.2).

Because of the number and complexity of the local mesh modifications possible and required to make the elements in the neighborhood of an unacceptable element acceptable, an incremental approach is used which focuses on the specific mesh entities of an unacceptable element that must be modified or eliminated. The basic justification for this is that, under the assumption that an acceptable set of elements can be created through mesh modifications in the local neighborhood, any final mesh modification can be obtained through a series of basic mesh modification operations. It is also assumed that it is possible to determine the final modifications needed by the incremental application of a set of operations.

The classes of local mesh modification operators that can be used in the process of producing a mesh of acceptable elements includes:

1. Altering the shape of mesh edges and mesh faces classified interior to the domain.
2. Repositioning of mesh vertices without altering their classification.
3. Performing local retriangulation, where retriangulation is defined as an operation where the mesh topology is modified without changing the number of mesh vertices and without changing their position.
4. Deletion of elements with reclassification of interior mesh entities to the boundary. This operation includes the repositioning of reclassified mesh vertices onto the boundary entity they are reclassified onto. As with the previous operations, this one does not change the number of mesh vertices.
5. Performing local remeshing, where remeshing is defined as an operation that adds or deletes one or more mesh vertices.

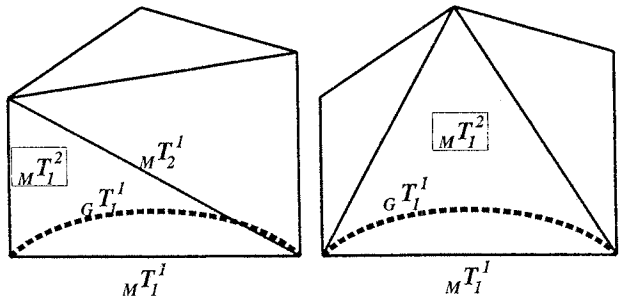


FIG. 3.3. *Example of creation of acceptable elements through retriangulation*

Although the altering of the shape of mesh edges and faces classified interior to the domain can often yield acceptably shaped elements with little effort, it is considered an undesirable operation since it increases the number of curved finite elements. Curved elements require higher order numerical integration rules to ensure the convergence rate and are typically considered to be less desirable than straight sided elements due to potential stiffening due to Jacobian variation.

Mesh vertex repositioning is a simple operation which is useful in some cases. However, it is of limited applicability as is easily seen in the example in Figure 3.2 where any reasonable repositioning of mesh vertices $_MT_3^0$ and $_MT_4^0$ under the constraint that the other mesh vertices are constrained will not yield acceptable elements.

Under the right conditions the application of a retriangulation operation can effectively yield an acceptable mesh. Consider the example shown on the left side of Figure 3.3 where the element $_MT_1^2$ becomes unacceptable when $_MT_1^1$ is curved to the model edge $_GT_1^1$ since it intersects mesh edge $_MT_2^1$. However, if the same set of mesh vertices is retriangulated as shown on the right of Figure 3.3, element $_MT_1^2$ is fully acceptable when $_MT_1^1$ is curved to the model edge $_GT_1^1$. Retriangulation has the basic advantage of closely maintaining the original mesh gradation since it used the original set of mesh vertices in their original positions. Retriangulation can not, however, yield acceptable elements in all situations. Again consider Figure 3.2 where no possible retriangulation will yield an acceptable set of elements.

Deletion of elements with reclassification of mesh entities is a useful local mesh modification operation for eliminating unacceptable elements. Figure 3.4 shows a simple 3D situation where all the mesh vertices of the element $_MT_1^3$ are classified on $_GT_1^2$. In this case all the mesh edges except $_MT_2^1$ are also classified on $_GT_1^2$. When the element $_MT_1^3$ is deleted so is the mesh edge $_MT_1^1$ and the two mesh faces it bounds. The deletion operation is completed by reclassifying the mesh edge $_MT_2^1$ and the two mesh faces of

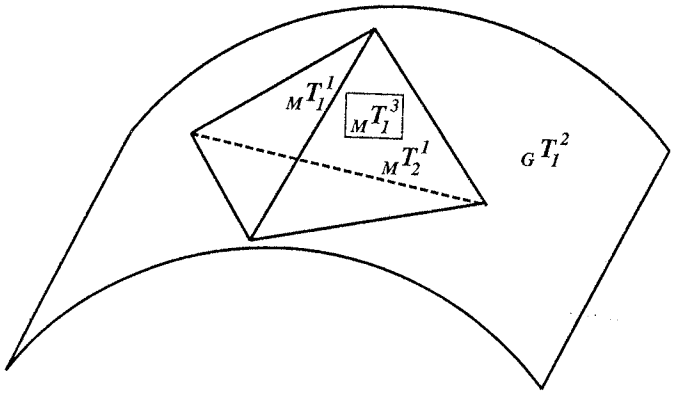


FIG. 3.4. *Deletion of a 3D element with reclassification of mesh entities*

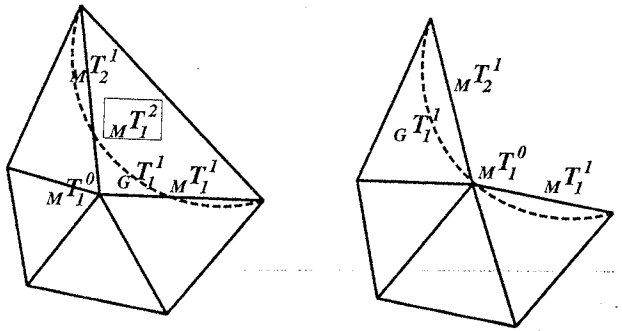


FIG. 3.5. *Deletion of a 2D element with reclassification of mesh entities and repositioning of a mesh vertex*

the original element ${}_M T_1^3$ that it bounds from the model region to the model face ${}_G T_1^2$. The most useful version of the element deletion operation is one which also allows the reclassification of mesh vertices from a model region to one of its boundary entities. Such reclassification of mesh vertices is only valid if the coordinates of the mesh vertex are modified so it is positioned on the model boundary entity upon which it is classified. Figure 3.5 shows a 2D example of this operation in which the element ${}_M T_1^2$ is deleted. In this case the mesh vertex ${}_M T_1^0$, and mesh edges ${}_M T_1^1$ and ${}_M T_2^1$ are reclassified on the model edge ${}_G T_1^1$. In the process the mesh vertex ${}_M T_1^0$ was repositioned to lie on the model edge ${}_G T_1^1$.

The ability of remeshing procedures to add and delete mesh vertices makes it the most flexible of procedures. At the cost of at least some variation in the local mesh gradation, it can always yield an acceptable set of elements. Figure 3.6 shows two possible retriangulations starting

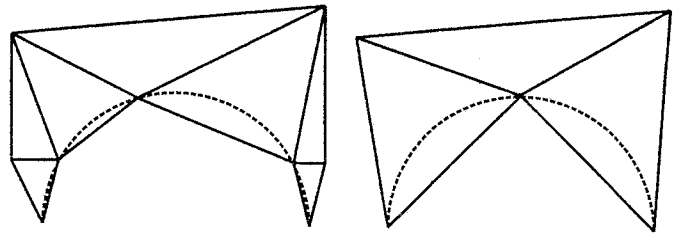


FIG. 3.6. *Two retriangulations of the mesh with unacceptable elements shown in Figure 3.2*

from the mesh containing unacceptable elements shown in Figure 3.2. The retriangulation on the left maintained all the original mesh vertices and created three new ones on the model edge, while the retriangulation on the right eliminated two interior mesh vertices and added one on the model edge.

The power of remeshing coupled with the ability of retriangulation and element deletion to effectively convert a set of unacceptable elements into acceptable elements makes them the primary tools for performing the required local mesh modifications. The current mesh modification algorithm uses only these three classes of mesh modification. The final procedure will include mesh vertex motion, and, if further study indicates a necessary, or useful, role for the curving of mesh entities classified interior to the domain, it will also be included. The current algorithm begins by creating a list of unacceptable shaped elements and an indication of the mesh entities which must be modified or eliminated. With this input, the steps in the incremental procedure are:

1. Select the next unacceptable element from the list. Terminate when the list is empty.
2. Considering the mesh entities that must be modified or eliminated, evaluate the best, if any, retriangulation operation which either yields an acceptable set of elements, or improves the situation by reducing the number of mesh entities to be modified or eliminated. If the best retriangulation operation yields an acceptable set of elements, it is performed, the list of unacceptable elements updated, and control returned to Step 1.
3. Considering the mesh entities that must be modified or eliminated, evaluate the best, if any, deletion operation which either yields an acceptable set of elements, or improves the situation by reducing the number of mesh entities to be modified or eliminated. If the best deletion, or combined deletion and retriangulation operation,

that yields an acceptable set of elements is performed, the list of unacceptable elements updated, and control returned to Step 1. If there is a deletion, or combined deletion and retriangulation operation, that reduces the number of problem entities, perform the operation, update the list of unacceptable elements, and continue.

4. Considering the mesh entities that must be modified or eliminated, evaluate the best remeshing operation which either yields an acceptable set of elements, or improves the situation by reducing the number of mesh entities to be modified or eliminated. It is always possible to perform a local remeshing that will improve the situation. After the remesh is performed, update the list of unacceptable elements and return to Step 1.

One reason triangulation and deletion are attempted before mesh modification is they generally maintain the mesh gradation closer to that of the original mesh because they do not change the number of mesh vertices. An open issue however is the selection and application of appropriate retriangulation or deletion operations when they appear to improve the situation, but must still be followed by remeshing operations to yield acceptable elements locally. Since the remeshing operations alter the triangulation as well as introduce new mesh entities, it is not critical to exhaustively evaluate all the retriangulation and deletion procedures that may lead only to an improved situation before proceeding to attempting a remeshing. The number of options evaluated is driven more by the desire to create acceptable elements without remeshing. On the other hand, it is critical when a remeshing operation is required to determine the appropriate operation. Experience indicates that the constraints on the type of retriangulation and remeshing operations allowed, and the basic properties of the various specific retriangulation and remeshing operations, usually make it clear which mesh modifications should be performed. However, the lack of a proof of a convergent sequence of operations for all situations indicates additional effort is required to ensure the overall reliability of the procedure.

Application mesh modification procedures

The decision of which retriangulation, deletion or mesh modification procedure to apply depends on the entities causing the element to be unacceptable. The operations required when the curved mesh entities are too far from their linear approximation (Figures 3.1e and 3.1f) are fairly obvious. In these cases remeshing operations which add mesh vertices classified on the geometric entities of concern are needed. In those cases where the curved mesh entities intersect or come too close to other mesh entities (Figures 3.1a through 3.1d) it is not obvious which mesh modification is the most appropriate. As indicated above, the retriangulation and deletion operations are considered first followed by consideration of remeshing.

Unlike the 2D case where the basic retriangulation operation is a diagonal swap, there are a large number of retriangulation operations in 3D. The two most powerful that have been qualified into useful tools are edge swapping [1,5,2] and multi-face removal [1]. Edge swapping considers a mesh edge that is to be eliminated from the triangulation. Deleting all the elements that are bounded by that edge creates a polyhedral domain that can be remeshed by creating edges between selected mesh vertices and defining the appropriate mesh faces and mesh regions. The number of possible retriangulations grows as the square of the number of mesh vertices on the bounding polygon [5,2], therefore, an effective implementation must quickly determine a limited number of possibilities to be evaluated [1,5,2]. Although edge swapping has the advantage of lending itself to an effective algorithmic implementation, the total number of mesh regions in the retriangulated region is higher when the total number of mesh vertices in the local polygon is seven and greater [1,5,2]. Since retriangulations which reduce the number of mesh regions are typically superior to those that increase the number of mesh regions, it is desirable to consider the development of operators that do this. The recently developed multi-face removal operator, which is the reverse of the edge swap operator addresses this issue.

The decision as to which retriangulation option to apply is driven by the entities which must be modified or eliminated. If the undesirable entity is a mesh edge bounding the unacceptable element, the edge swap configurations based on that edge are considered. In addition, if the unacceptable edge is classified interior to the model then multi-face swaps which eliminate that edge are also considered. If the undesirable entity is an interior face then a multi-face swap is considered along with edge swap based on the bounding edges of the undesirable face.

The swapping possibilities are evaluated based on the number of undesirable connections that are produced in elements of the resulting mesh compared to the original mesh sharing the entity on which retriangulation is based. If a retriangulation exists that results in all affected elements being acceptable, it is applied. If no retriangulation could be found that makes all the elements acceptable in the resulting mesh the deletion options are considered. If a deletion option is found that results in a mesh that yields no unacceptable elements it is applied. If this does not yield all acceptable elements, the best possible retriangulation and deletion options are compared based on the number of undesirable mesh entities in the affected mesh and the one which results in fewer undesirable mesh entities is selected to be applied and the process repeated with the next unacceptable element.

If unacceptable elements remain after the application of retriangulation and deletion, remeshing procedures are then applied. Again, there are a wide variety of possible tools available for this process. In the current implementation the primary remeshing tool for creating additional mesh

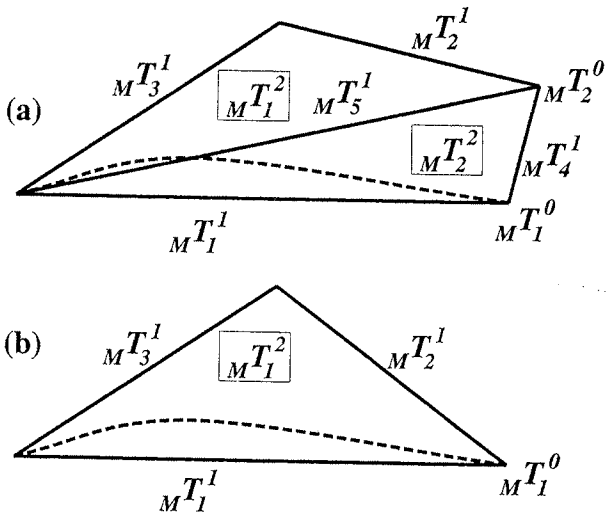


FIG. 3.7. *Example of a 2D edge collapse operation to eliminate an unacceptable element*

vertices is splitting of mesh entities, while the primary tool for reducing the number of mesh vertices is collapsing. The focus of the application of these two operations are mesh edges [1] since their splitting and collapsing operations typically produce the best results.

The edge collapse operation is typically applied to edges of unacceptable elements which have one mesh vertex classified on the boundary and one classified interior to the domain The collapse of such an edge typically eliminates the problem mesh entities by collasping them onto mesh entities classified on the boundary. Figure 3.7 demonstrates this process for a simple 2D example. The original mesh (Figure 3.7a) is modified by collapsing $_MT_4^1$ which pulls $_MT_2^0$ onto $_MT_1^0$ eliminating the mesh entities $_MT_2^0$,$_MT_4^1$, $_MT_5^1$, and $_MT_2^2$. This collapsing process yields the acceptable element $M^{T_1^2}$ shown in Figure 3.7b.

When the curved mesh entities are too far from their linear approximation edge splitting is applied. In edge splitting a new mesh vertex is introduced along the mesh edge the appropriate mesh entities connecting to it and the entities it bounds are created. Since the new mesh vertex inherits the boundary classification of the mesh edge, it is positioned at an appropriate location on the model boundary entity. As demonstrated in the left hand example of Figure 3.6 , edge splitting plus retriangulation can also be used to eliminate the unacceptable elements which intersect the curved boundary shown in Figure 3.2. The actual implementation in this case is best performed by:

1. Deleting the unacceptable elements creating an empty polygon of mesh entities.
2. Splitting the mesh edges classified on the model boundary the appropriate number of times.
3. Using the new mesh edges retriangulate (creation of surface triangulations) the appropriate portions of model faces, yielding an updated empty polygon.
4. Fill the updated empty polygon by tetrahedral element triangulation. This step may or may not introduce new mesh vertices into the polygon.

Examples of local mesh modifications to produce curved meshes of acceptable elements

Two images are given for each of the examples discussed in this subsection. The first shows the original mesh in which the element edges for elements that would become unacceptable if curved are displayed as straight edges. The second image shows the mesh after performing the appropriate mesh modifications where all the mesh edges classified on curved boundary entities are curved.

The first curved mesh example is depicted in Figures 3.8a and 3.8b. In the original mesh the mesh edge on the circular model edge that was left straight (Figure 3.8a) would cause an invalid element if curved. A simple retriangulation (an edge swap in this case) resulted in an acceptable curved mesh (Figure 3.8b).

The second curved mesh example also had an invalid element as its worst case where one of the curved edges classified on a model face was intersecting a bounding mesh face classified in the interior of the model (Figure 3.8c). Two local retriangulations (edge swaps of the bounding edges of the undesirable face) lead to an acceptable curved mesh (Figure 3.8d).

The original mesh for the third example also had invalid elements resulting from the curving of some of the edges. The top mesh in Figure 3.9 shows the mesh edges left straight in the original mesh. The corrective steps required to get to the acceptable curved mesh (bottom figure) consisted of local retriangulation, element deletion and edge collapsing. The edge collapsing procedures lead to a slight coarsening of the final mesh.

4. Curvature-based refinement. Curvature-based refinement provides a convenient method to control the gradations of the mesh on objects with curved faces. It has been observed that given a variety of a priori mesh control devices, users find curvature-based refinement options among the most useful. The primary reasons for this is that it provides a direct means to control the geometric approximation of straight-sided finite elements yielding esthetically pleasing meshes. More importantly, in a variety

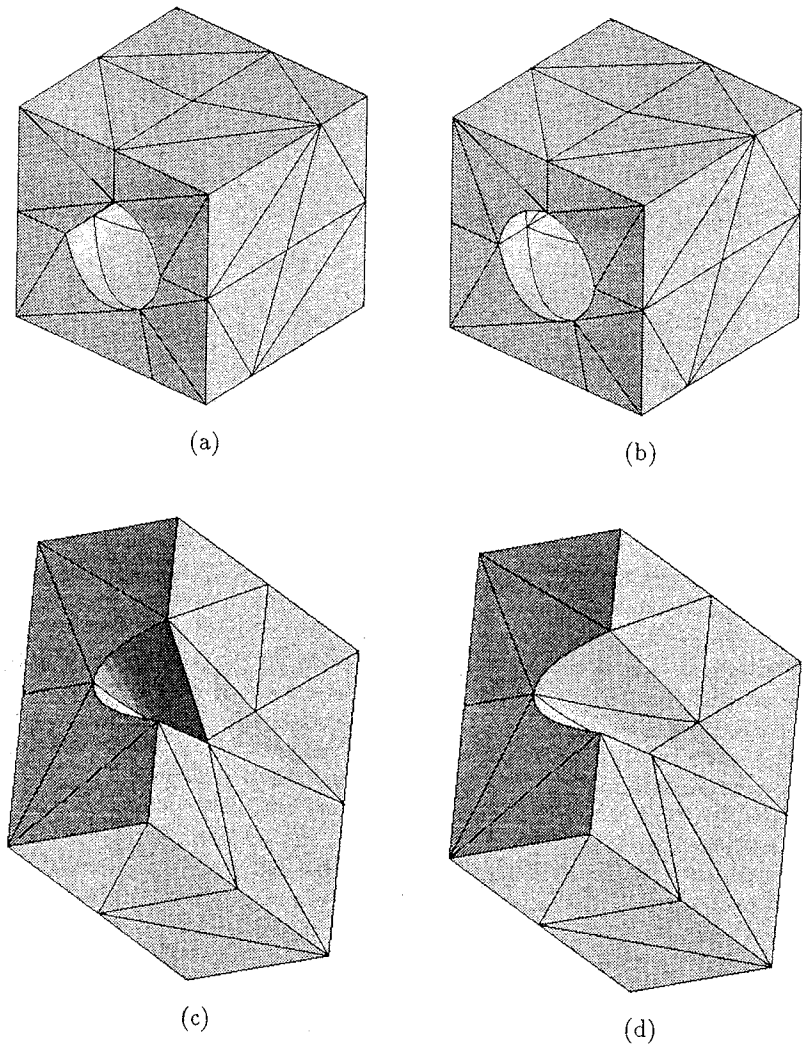


FIG. 3.8. *Mesh examples 1 and 2*

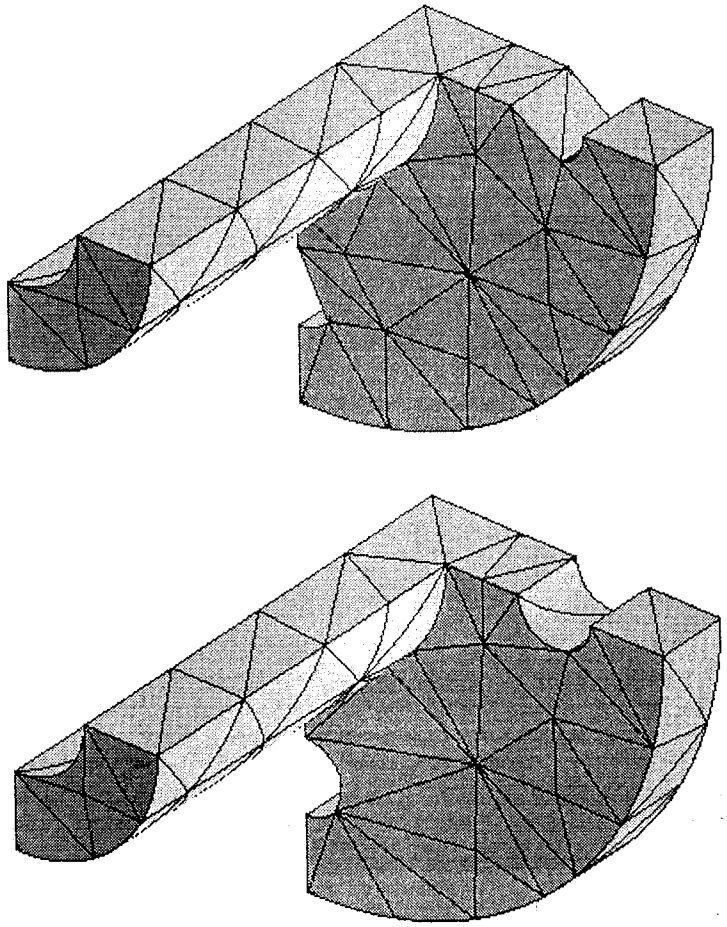


FIG. 3.9. *Curved mesh example 3*

of analysis types, ranging from the stress in solid parts, to the flow around an aircraft, the areas of high curvature, which will receive a finer mesh, tend to be among the critical areas in the analysis. Therefore, mesh refinements based on curvature are typically effective initial meshes.

Approach

The application of curvature-based refinement first requires an ability to evaluate the curvature of the surface. Maintaining a functional link to the geometric model representation which employs only pointwise geometric interrogations [15,16], the options to obtain curvature information are to directly request it of the geometric modeler, or to obtain an approximate measure in terms of the distance between the centers of straight finite element entities and curved model faces. Two drawbacks of the approximate measure are (i) the interrogation to find, even approximately, the distance from the point to the face is an expensive operation, and (ii) this distance only provides a scalar value, not allowing any potential use of the vectorial nature of the curvature[1] Although not all modeling systems provide a direct measure of the surface curvatures, enough derivative information is available to calculate the curvature information [3,4,6,8]. The specific curvature information determined by the geometric interrogation is to obtain the principal curvatures and directions at given points on the surface.

One approach to apply curvature-based refinement is to evaluate the curvature at node points on the model face as they are generated and to use this information to set the appropriate parameters in the mesh generator to control the element sizes in that area. For example, following this approach within the Finite Octree mesh generator [15], where the nodes on the model face are obtained by intersecting octant edges with the model face, the curvature would be evaluated at those intersection points. If the curvature is high enough to indicate additional refinement is required, the tree is locally refined and new node points obtained at the intersections of the new octant edges with the model face. Of course the curvature at these new nodes would also be evaluated. A danger of this approach is that when the curvature varies over the face, and is locally high with respect to the basic finite element mesh control parameters, areas of high curvature can be missed. Therefore, it is appropriate to employ additional sampling points in the process. Although it is possible to develop an adaptive sampling approach that minimizes the number of additional sampling points that may be required, the computational effort required does not justify its use over a simple sampling process over a more uniform structure.

The uniform structure used in this work is a grid in parametric space. A comparison of sampling on the corners of a uniform grid to that of sampling

[1] Although not demonstrated here, the vector nature of curvature can be effectively used to perform directionally-based refinement. This can be highly desirable in cases such as obtaining the best surface approximation with the minimum number of discrete facets.

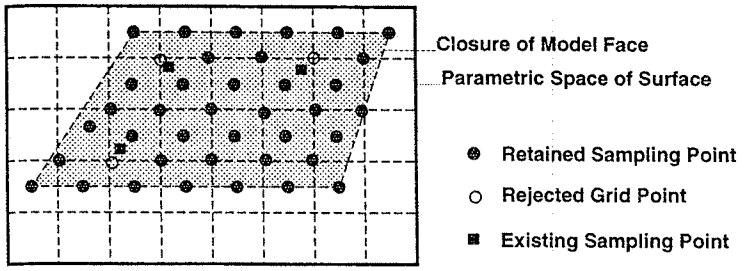


FIG. 4.1. *Sampling points for use in curvature-based mesh refinement*

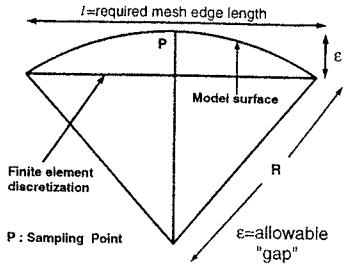
at both the corners and centroid of each cell indicates that the use of both the corners and centroid yields a smaller distance between points for the same total number of sampling points. To account for the fact that faces can be trimmed, only those sampling points on the face are considered. In addition, sampling points that are close to points sampled during any previous meshing steps are not considered. Figure 4.1 shows the various sampling points just discussed.

Given the above considerations, a basic curvature based refinement procedure applied during the meshing of a model face consist of:

1. Obtain principal curvature at each node point generated on the model face.
2. If the local element sizes are too large, locally alter the mesh control functions to obtain the correct size elements.
3. Sample at all newly created nodes and refine as necessary.
4. Determine additional sampling points based on the grid in parametric space.
5. Evaluate the curvature at the additional sampling points and refine the mesh as required.

An alternative approach would be to first evaluate at all the grid sampling points within the face and locally alter the mesh control information so the correct mesh gradation is obtained. The selection between the two approaches should be based on the computational efficiency of the approaches with the particular mesh generation algorithm.

Accounting for the vector nature of the curvature to control directional refinement would follow a similar process. The main difference is that the vector information on the curvature would need to be returned by the modeler, and the mesh generator must be able to account for anisotropic mesh control information. For example, the interrogation for curvature information at a point could return the principal curvatures and principal directions. This information could then be used to set the element



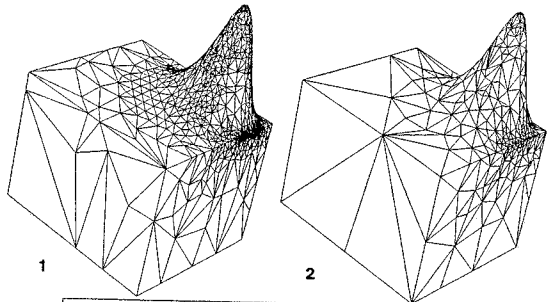
$$R^2 = (R-\varepsilon)^2 + \left(\frac{1}{2}\right)^2$$

$$\ell = (8\varepsilon R - 4\varepsilon^2)^{\frac{1}{2}} \tag{4.1}$$

$$R = \frac{1}{\kappa}$$

Constraint : $\varepsilon < 2R$

FIG. 4.2. *Relating curvature to element edge length*



Approach	Nodes	Tetrahedrons
1	2417	10620
2	788	3219

FIG. 4.3. *Example comparing two methods of controlling element length*

size directions in the principal directions for use by the mesh generation procedure.

It is still necessary to determine the correspondence between the values of the curvature and the mesh control information. A straight forward means to do this is to convert a curvature value into the requested element length at that point in the direction to which the curvature was measured. One way to do this is to consider the geometric approximation of a straight element edge to the curved geometry. Assuming a constant curvature, an acceptable local approximation, the curvature, κ, can be related to the distance from a straight edge to the curve, ε, to the length of that straight edge, ℓ, as shown in Figure 4.2.

One method to employ the distance between the element edge and a model face is to force a uniform geometric approximation by setting

$$\varepsilon = \varepsilon_c \tag{4.2}$$

where ε_c is a constant. Under these conditions the curvature dependent

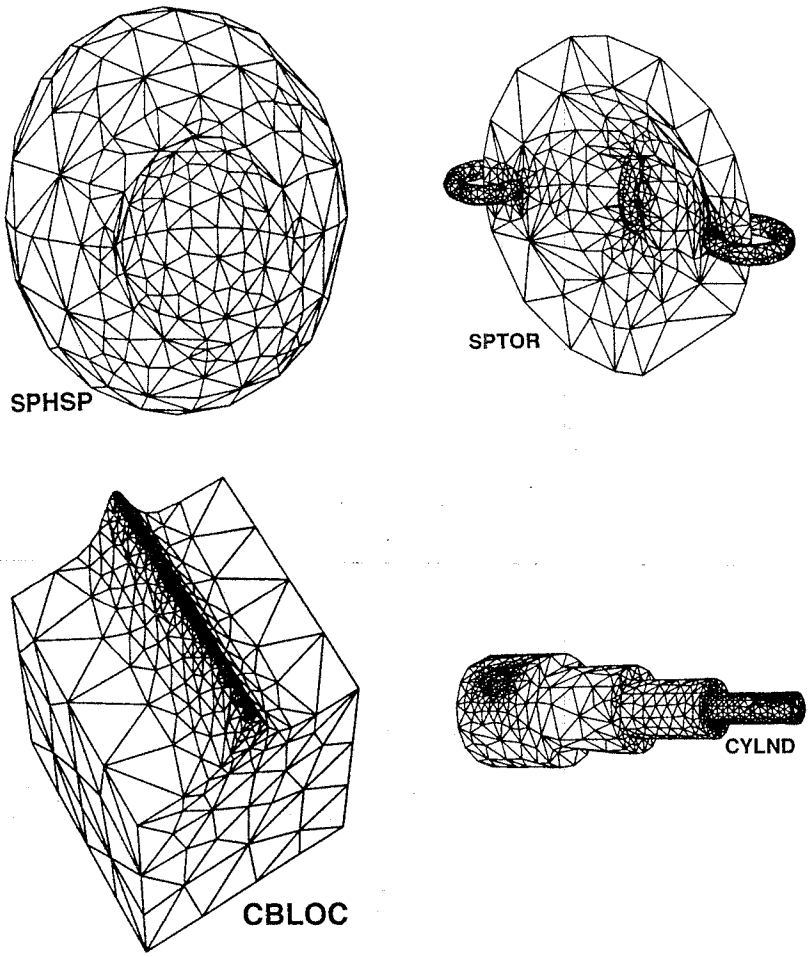


FIG. 4.4. *Examples of curvature based refinement*

value of the element edge length is

$$\ell = (8R_m\varepsilon_c - 4\varepsilon_c^2)^{\frac{1}{2}} \tag{4.3}$$

where $R_m = \dfrac{1}{\max(|\kappa_1|, |\kappa_2|)}$ and κ_1 and κ_2 are the maximum and minimum principal curvatures respectively. An alternative method to the specification of the required element edge length that users tend to favor is to make the distance between the element edge and model face proportional to the element edge length as

$$\varepsilon = \alpha\ell \tag{4.4}$$

where α is the proportionality constant for that model face. Under these conditions the curvature dependent value of the element edge length is

$$\ell = \frac{8R\alpha}{1 + 4\alpha^2} \tag{4.5}$$

Implementation in finite octree

To demonstrate the application of curvature-based refinement, it was implemented in the Finite Octree mesh generator [15]. Within Finite Octree the element sizes are controlled by the size of the octant, which is dictated by the level of the tree at that location of that octant. Assuming that the average edge length is equal to the octant edge length, the level of tree level, N required at a particular location is defined by

$$N = \log_2\left(\frac{L}{\ell}\right) \tag{4.6}$$

where L is the side length of the root octant which is the cube enclosing the object being meshed. In this case the application of the constant distance between the element edges and model faces will yield octant levels defined by

$$N = \frac{1}{2}\left\{\log_2\left(\frac{L^2}{8R\varepsilon_c - 4\varepsilon_c^2}\right)\right\} \tag{4.7}$$

while making the distance proportional to the element edge length yields

$$N = \left\{\log_2\left(\frac{(1 + 4\alpha^2)L}{8R\alpha}\right)\right\} \tag{4.8}$$

Figure 4.3 shows the meshes comparing the two methods for a simple object with one curved face with variable curvature where the shortest element lengths are the same.

To minimize the computational impact of the curvature-based refinement on the meshing process, its implementation evaluates the curvature as each node point is defined by the intersection of an octant edge with the model face. If the value of the curvature dictates octant level refinement, it is carried out immediately. The additional sampling points, and any refinements required by them, are then carried out after the face was inserted. When implemented in this manner the average increase in mesh generation time caused by the curvature evaluations was 1.35%.

Figure 4.4 shows a set of examples generated using curvature-based mesh control only.

5. Closing remarks. The generation of valid, controlled finite element meshes for general curvilinear 3–D geometries introduces a number of additional complexities past that of planar geometries. This paper has considered aspects of these complexities including (i) ensuring the validity of the finite element mesh, (ii) the ability to maintain validity and acceptability of element shapes when curved finite elements are introduced to reduce the geometric approximation and (iii) applying mesh gradation accounting for the curvature of the geometry. Of the techniques presented, those that deal with the mesh modifications to produce acceptable curved finite elements would benefit from further qualification to ensure their reliability.

6. Acknowledgment. The authors would like to acknowledge the support of the Industrial Affiliates of the Scientific Computation Research Center.

REFERENCES

[1] H.L. DE COUGNY AND M.S. SHEPHARD, *Refinement, derefinement and local retriangulations in three-dimensions*, 1994, in preparation for submission.

[2] E.B. DE I'ISLE AND P.L. GEORGE, *Optimization of tetrahedral meshes.* Technical report, INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex, France.

[3] S. DEY, *Curvature sensitive refinements in 3D automatic mesh generation.* Master's thesis, Scientific Computation Research Center, Rensselaer Polytechnic Institute, Troy, NY 12180–3590, May 1993.

[4] G. FARIN, *Curves and Surface for Computer Aided Geometric Design.* Academic Press Inc., 1990.

[5] P.L. GEORGE, *Generation de maillages par une methode de type Voronoi, Partie 2: le cas tridimensionnel.* Technical Report RR INRIA n 1664, INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex, France, 1993.

[6] D.H. HOITSMA, *Surface curvature analysis.* In *Geometric Modeling for Product Engineering.* North-Holland, Amsterdam, 1990.

[7] T.J.R. HUGHES, *The Finite Element Method: Linear Static and Dynamic Finite Element.* Prentice Hall, Englewood Cliffs, NJ, 1987.

[8] M. MORTENSON, *Geometric Modeling.* J. Wiley and Sons, New York, 1985.

[9] PDA Engineering, PATRAN Division, 2975 Redhill Avenue, Costa Mesa, CA 92626. *Patran Plus User Manual*, Release 2.5, October 1990.

[10] W.J. SCHROEDER, *Geometric Triangulations: with Application to Fully Automatic 3D Mesh Generation*. PhD thesis, Rensselaer Polytechnic Institute, Scientific Computation Research Center, RPI, Troy, NY 12180–3590, May 1991.

[11] W.J. SCHROEDER AND M.S. SHEPHARD, *An $O(N)$ algorithm to automatically generate geometric triangulation satisfying the Delaunay circumsphere criteria*. Engng. with Computers, (5(3/4):177–194, 1989.

[12] W.J. SCHROEDER AND M.S. SHEPHARD, *A combined octree/Delaunay method for fully automatic 3-D mesh generation*. Int. J. Numer. Meth. Engng., 29:37–55, 1990.

[13] W.J. SCHROEDER AND M.S. SHEPHARD, *On rigorous conditions for automatically generated finite element meshes*. In J. Turner, J. Pegna, and M. Wozny, editors, *Product Modeling for Computer-Aided Design and Manufacturing*, pages 267–281, North Holland, 1991.

[14] M.S. SHEPHARD, *Automatic generation of finite element models*. In M. Papadrakais, editor, *Solving Large Scale Problems in Mechanics: The Development and Applications of Computational Solution Methods*, to appear 1992.

[15] M.S. SHEPHARD AND M.K. GEORGES, *Automatic three-dimensional mesh generation by the Finite Octree technique*. Int. J. Numer. Meth. Engng., 32(4):709–749, 1991.

[16] M.S. SHEPHARD AND M.K. GEORGES, *Reliability of automatic 3-D mesh generation*. Comp. Meth. Appl. Mech. Engng., 101:443–462, 1992.

[17] B.A. SZABO AND I. BABUSKA, *Finite Element Analysis*. Wiley Interscience, New York, 1991.

[18] O.C. ZIENKIEWICS AND R.L. TAYLOR, *The Finite Element Method - Volume 1*. McGraw-Hill Book Co., New York, 4th edition, 1987.

OPTIMIZATION OF TETRAHEDRAL MESHES

ERIC BRIÈRE DE L'ISLE* AND PAUL LOUIS GEORGE*

Abstract. Finite element computations are all the more exact if we start from "good" elements. We are interested in meshes where the elements are tetrahedra and we shall develop utilities allowing us to improve the quality of these meshes.

1. Introduction. The aim of this paper is to propose a method that enables us to improve the quality of a mesh. The meshes we consider are only composed of tetrahedra and are the result of mesh generation algorithm such as Delaunay-Voronoï, advancing front, octree, etc. ...(see, for example, [2]). The paper covers three cases:

- **Isotropic case**: In this case, the aim is to improve the shape of the elements in the mesh.
- **Isotropic case with size specification**: In addition to the initial mesh, we have, in this case, a constraint regarding the desired size, i.e. a function that enables us to know for each location in the mesh the desired size. Thus, the aim is to satisfy this property (i.e., for example, to obtain the correct edge size) by locally modifying the mesh while preserving the element quality in terms of shape.
- **Anisotropic case**: This case is quite similar to the previous one but the constraint is replaced by a tensor field that can be seen as a size specification along the three directions. The purpose of the method is then as above (obtain a good quality mesh (in terms of shape) such that the specification is satisfied).

After clarifying the notion of a mesh quality, four local tools are given:

- point relocation,
- edge (or face) removal resulting in a local re-meshing,
- point creation to suppress an edge,
- point removal.

The proposed method for optimizing the meshes consists in the adequate use of the previous tools. The summary of the paper is as follows: after giving some useful definitions, each of the three cases of optimization is discussed and illustrated by various application examples, after which a conclusion is given.

2. Definitions. In this section, some useful definitions are given. First, the mesh quality regarding the three cases is established, then two local set of elements are introduced which will be used in the different steps of the method: a *shell* and a *ball*.

* INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cédex, France.