Feature Recognition Applications in Mesh Generation[1]

Timothy J. Tautges
Sandia National Laboratories
PO Box 5800, Albuquerque, NM, 87112

Shang-sheng Liu
Yong Lu
Jason Kraftcheck
Rajit Gadh
Mechanical Engineering Dept.
University of Wisconsin-Madison
Madison, WI 53706

1. This work sponsored by the U.S. Department of Energy under Contract No. DE-AC04-94AL85000, by ALCOA Technical Center Grant No. TC 958620 to the University of Wisconsin, and by NSF Research Initiation Grant No. DMI 9410359 to the University of Wisconsin.

ABSTRACT

The use of feature recognition as part of an overall decomposition-based hexahedral meshing approach is described in this paper. Our meshing approach consists of feature recognition, using a c-loop or hybrid c-loop method, and the use of cutting surfaces to decompose the solid model. These steps are part of an iterative process, which proceeds either until no more features can be recognized or until the model has been completely decomposed into meshable sub-volumes.

This method can greatly reduce the time required to generate an all-hexahedral mesh, either through the use of more efficient meshing algorithms on more of the geometry or by reducing the amount of manual decomposition required to mesh a volume.

INTRODUCTION

Mesh generation algorithms for meshing three-dimensional volumes can be broadly classified into two groups, those yielding tetrahedral and hexahedral-shaped elements. A fundamental difficulty for automated meshing is that a mesh is constrained in terms of how elements can share subfacets with each other. This is much less constraining for tetrahedral meshes than for hexahedral meshes. However, for reasons of accuracy, convergence (i.e. mesh locking), and application-specific requirements (e.g. boundary layers for CFD), hexahedral meshes are often preferred over tetrahedral meshes [1].

The automatic generation of all-hexahedral meshes for general 3D geometries is an unsolved problem, although there are several approaches being investigated which show promise, for example whisker weaving [2] and boundary-fitting [3]. At this time, generating all-hexahedral meshes for complex geometries, here referred to as volumes, requires the user to decompose the geometry into meshable pieces, or sub-volumes. The determination of whether a volume or sub-volume is meshable depends on the meshing algorithms available; for typical problems, a variety of meshing algorithms will be used for various sub-volumes, depending on sub-volume shape or complexity. This decomposition and algorithm selection requires a great deal of user intervention time, and can account for over 90% of the model preparation time. This time must be drastically reduced before simulation can become an integral part of the design process.

There has been a great deal of feature recognition (FR) research, but focused mainly in areas other than mesh generation. For a survey of feature recognition work, see [4]. Although several FR algorithms are based on feature information stored in the solid model, these algorithms assume that the CAD system used to construct the model stores that information, and are therefore not as generally applicable. There are several algorithms which operate on BREP information in the solid model [5][6][9]. These algorithms are preferred, since BREP information can be obtained for most types of solid models. For the purposes of mesh generation, the FR algorithm must be able to recognize a very general set of features, and ones defined by the application. Furthermore, it must operate locally, that is it must be capable of detecting features by observing local geometry, rather than inspecting the entire model. This requirement is introduced by the need to mesh very complex models, ones with hundreds or thousands of topological entities (curves, surfaces, etc.).

This paper proposes using feature recognition to find protrusion-type features of a volume, which can be sliced off and

DISCLAIMER

meshed individually. We propose to use feature recognition as a *tool* to *speed* the meshing process, rather than a single algorithm to *steer* the *entire* meshing process. In other words, feature recognition is used to separate the volume into sub-volumes of varying complexity, each of which can be meshed with the appropriate meshing algorithm which minimizes meshing time but maximizes mesh quality.

The remainder of this paper is arranged as follows. Section 2 discusses the concept of a feature for mesh generation applications. Section 3 describes the feature recognition process used in this work. Section 4 discusses control of the overall feature recognition-decomposition-mesh generation process. Section 5 gives conclusions for this paper.

FEATURE DEFINITION FOR MESH GENERATION

When working with feature recognition for a particular application, the first question to ask is "what is a feature?". In the feature recognition context, a feature has been defined as "*a physical constituent of a part, mappable to generic shape and having engineering significance*" [4]. In traditional applications of feature recognition, for example design for manufacturing, the set of "generic shapes" or features is pre-defined and tends to be small. Furthermore, applications often dictate that a given part must be defined completely with known features.

In fact, the definition of features is extremely application-specific, and in many cases very constrained; for example, in NC machining, the features must be ones that can be created by some sort of tool moving along a prescribed path. In contrast, the definition of features for mesh generation is quite flexible. In either case, the definition of features depends on their significance in the target application. For NC machining, features are determined by the types of machine operations possible on the target machine; for mesh generation, features are determined by the available meshing algorithms. As a rule of thumb, for mesh generation features, if a piece can be meshed atomically with a single meshing algorithm, it can be considered a feature.

The "rule of thumb" expression in the previous paragraph is necessary for the following reason. Because we assume that it is difficult or undesirable to decompose a given solid completely into known, simple features, we also assume the availability of a general 3D all-hex meshing algorithm. However, by this definition, any part could be considered a feature without the need for decomposition. We avoid this for two reasons. First, we do not yet have a robust, automatic 3D meshing algorithm, and manual decomposition can be substituted for that step. Second, even if we did, we would still want to use simpler meshing algorithms where possible, for speed and possibly mesh quality. Therefore, the notion of where to stop recognizing features depends not only on meshability but also on mesh quality and speed of meshing the given sub-volume and the algorithm applied to it.

Note that for the purposes of mesh generation, the features of interest are of the protrusion type, rather than both protrusions and depressions (although mesh generation features can contain depressions, these features are not of interest in and of themselves).

Previous attempts to apply feature recognition to mesh generation have failed largely because of the requirement that a volume be defined completely by a relatively small set features with known, deterministic shape. For example, the AMEKS effort attempted to develop algorithms which would decompose a given solid into a set of primitive shapes [7]. In another effort, the medial axis was used to decompose the volume into these primitive shapes [8]. The robustness of both these approaches was limited by the inability to treat fully general 3D geometries. In contrast, our approach combines a flexibility of feature definition with the capability to decompose a geometry only partway into (recognized) features.

FEATURE RECOGNITION PROCESS

The feature recognition approach used in this effort is based on recognizing Basic LOgical Bulk (BLOB) shapes as features using the methods described by Liu and Gadh [9]. This procedure has two stages. The first is the decomposition of the solid model into BLOBs. The meshing of the BLOBs is the second stage.

The algorithm for decomposition of the solid model is composed of three steps. The first step is the BLOB determination step, during which a BLOB is found. The second step is finding the cutting face for the removal of the determined BLOB. The third step is the actual decomposition of the solid model (by removing the BLOB). This cycle is then repeated for both of the decomposed BLOBs. The decomposition of a simple model is shown in Figure 1.

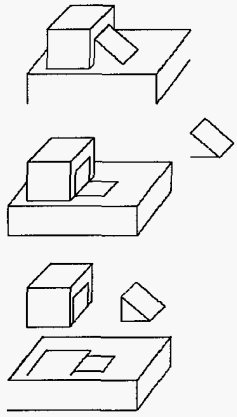

Figure 1. Decomposition of a simple model into BLOBs [9].

The BLOB determination is accomplished through the use of c-loops, as defined in Liu and Gadh [9]. A c-loop is a loop of all concave or convex edges. A protrusion can be identified as enclosed material with one concave c-loop. A blind depression consists of one convex c-loop and no enclosed material. A through depression has one more convex c-loop than the blind depression. A bridge is a feature which encloses material and has two concave c-loops [9]. Figure 2 illustrates these four cases.

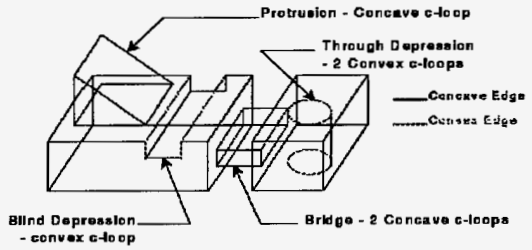

Figure 2. Examples of cloops and features defined by them.

For the object in Figure 3, there is obviously a feature, but

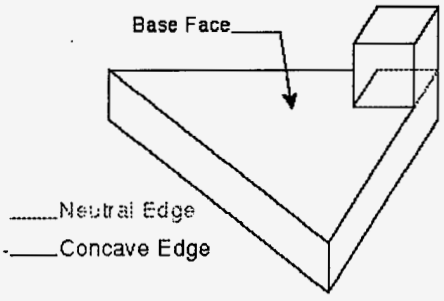

Figure 3. A hybrid cloop [9].

there is not a complete c-loop. The concept of a hybrid c-loop was developed to handle such cases in Liu and Gadh [10]. A hybrid c-loop is composed of a loop of neutral edges with either convex or concave edges. The hybrid concave c-loop is shown in Figure 3. The neutral edge is an artificial edge on a face. The purpose of the neutral edges is to complete open c-loops.

The decomposition is accomplished through the use of a cutting face, as defined in Liu and Gadh [10]. The cutting face is determined by the c-loop, which may be determined by the placement of the neutral edges. The simplest method for determining the neutral edges is by extending the base face (the face containing the largest number of concave edges in the incomplete c-loop.) The base face and resulting neutral edges are shown in Figure 3. In Figure 4, a more complicated method of determining the neutral edges is necessary. If the method used in Figure 3 were applied to Figure 4, the resulting mesh of the decomposed BLOB would be very poor. In this case the MMA (Maximum Minimum Angle) method is used to determine the neutral edges [10]. Note that the best definition for the cutting surface may not be necessary for meshability, but may be desirable for maintaining local mesh quality.

Once the hybrid c-loop is completed, the cutting face is known, and the BLOBs can be decomposed. This process is done recursively on both of the resulting BLOBs until no more concave edges can be found. The decomposed BLOBs are stored in a tree structure such that the order of meshing is known.

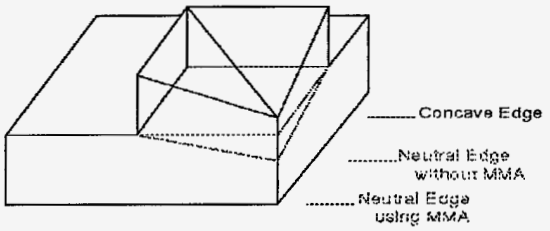

Figure 4. Neutral edges with and without the MMA approach [10].

Figure 5 illustrates the dependency of the mesh of a BLOB on

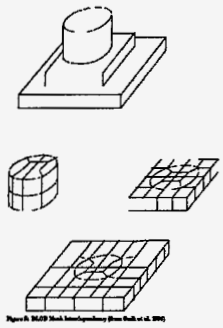

Figure 5. BLOB mesh interdependency [10].

it's neighboring BLOBs and the necessity of an appropriately organized means of storing the BLOBs.

After the recursive decomposition of the BLOBs is completed, there are several methods for meshing each BLOB. If the BLOB is a simple shape, it is mapped as a collection of hexahedral elements called a multiple block structure (MBS) that approximates the geometry of the BLOB [11]. The MBS of each BLOB is then mapped back to the actual shape of the BLOB and meshed. The MBSs for several simple shapes are shown in Figure 6,. If the BLOB is not sufficiently simple for the MBS method, a swept object method of meshing may be applicable. If the object is identified as a swept object as described in Yu, Liu, and Gadh [12], one of the end faces is meshed with quadrilateral elements and that two dimensional mesh is then swept along the object to generate the hexahedral mesh. The quadrilateral mesh and sweep path of a swept feature are illustrated in Figure 7,.

OVERALL PROCESS CONTROL

The ultimate goal of this research is to reduce the amount of time required to generate an analysis model from a design model, both in terms of interactive user time and cpu time. Separating the volume into sub-volumes of varying complexity and target meshing algorithm allows the optimal combination of algorithm

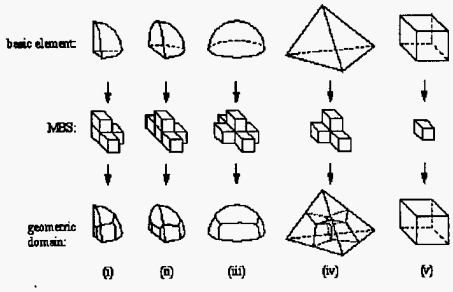

Figure 6. Basic elements and their corresponding MBSs and meshes [11].

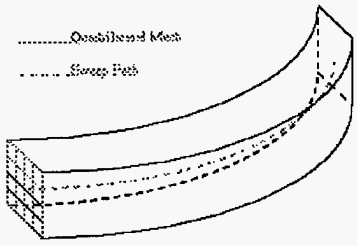

Figure 7. Swept feature meshing [13].

speed and mesh quality for each sub-volume. This minimizes the overall mesh generation time because it uses expensive algorithms only when necessary. In fact, this approach is already used to manually decompose and mesh complex geometries. We seek to automate both the decomposition and the meshing algorithm selection processes.

At a broad level, feature-based decomposition for mesh generation can be viewed as a repeated series of detect-decompose-evaluate operations (each described in a previous section), where features are detected on a volume, the volume is decomposed according to those features, and sub-volumes are evaluated for meshability. This process can be either fully or partially recursive; the pseudo code for a fully recursive algorithm is shown in Figure 8. In this version, a single feature is

- For a volume N:
  - find a feature F on volume N
  - slice off feature F from N, leaving sub-volumes N', N"
  - For volume n in N, N":
  - if n is not meshable with a simple algorithm, repeat overall process for n

Figure 8. Pseudo code for fully recursive feature-based decomposition algorithm.

recognized and the decomposition always separates one sub-volume into two sub-volumes. In a partially recursive version, multiple features are recognized and removed as part of the same iteration through the process (Figure 9). In both versions, the

- For a volume N:
  - find a set of non-interfering features Fi on volume N
  - slice off all features Fi from N, leaving sub-volumes N' and Fi
  - For volume n in N', Fi:
  - if n is not meshable with a simple algorithm, repeat overall process for n

Figure 9. Pseudo code for partially recursive feature-based decomposition algorithm.

process is repeated for sub-volumes, and terminates for a sub-volume when either no more features can be found or when the sub-volume can be meshed with a single algorithm. While the partially recursive method is more efficient, it may not be possible to detect all features on a volume in one pass due to feature interaction.

The determination of when to terminate the recursive sequence is based on a trade-off between time taken to recognize features and decompose versus time saved by using simpler meshing algorithms on a larger portion of the volume. This depends heavily on the relative speed and complexity of the available meshing algorithms. The meshing algorithms available in the CUBIT mesh generation code are classified in Table 1[14].

Table 1: CUBIT meshing algorithms and associated complexity of volumes meshed and approximate meshing speed [14].

Algorithm	Complexity[a]	Meshing speed (# hexes/second)[b]
Mapping	Low	10,000
Submapping	Medium	10,000
Translate	Medium	5000
Project	Medium-high	3000
Whisker weaving	High	100

a. In this context, complexity is inversely proportional to constraints on the volume required by the algorithm. For example, mapping requires a volume to have 6 sides with matching meshes on each pair of opposite faces. Algorithms with high complexity have no constraints.
b. Execution times approximate; measurements made on an HP 735/125 workstation.

Currently, the implementation of feature detection and decomposition is separated from the CUBIT code; therefore, feature detection continues until either no more concave edges remain or no more features can be detected. The resulting sub-volumes are written to an ACIS file, which is read by CUBIT. Meshing can then proceed on the sub-volumes, each using one of

the algorithms listed in Table 1.

CONCLUSIONS

This paper describes the use of feature recognition as part of an overall decomposition-based hexahedral meshing approach. Feature recognition is accomplished using a c-loop or hybrid c-loop method. From the c-loops, cutting surfaces are constructed; these cutting surfaces are used to decompose the solid model. Using an iterative process, decomposition proceeds either until no more features can be recognized or until the model has been completely decomposed into meshable sub-volumes.

While parts are rarely completely 2.5D (i.e. sweepable), often a substantial volume of the part appears in 2.5D features or sub-volumes. Using feature recognition, these sub-volumes can be separated from the base part and meshed individually, often with a faster algorithm. The remaining object can be meshed with a 3D free-form all-hexahedral mesher, or alternatively can be meshed using manual decomposition. In the case of an automatic 3D mesher, this approach reduces the overall meshing time because it uses the relatively expensive meshing algorithms only when necessary. On the other hand, if such a meshing algorithm is not available, the relative reduction in meshing time is even greater, because a great deal of user-driven decomposition is performed automatically.

REFERENCES

1. Steven E. Benzley, Ernest Perry, Karl Merkley, Brett Clark, Greg Sjaardema, "A Comparison of All Hexagonal and All Tetrahedral Finite Element Meshes for Elastic and Elasto-plastic Analysis", Proceedings of the 4th International Meshing Roundtable, Sandia National Laboratories report SAND95-2130, Oct. 16-17, 1995, Albuquerque, NM.

2. Timothy J. Tautges, Ted D. Blacker, Scott Mitchell, "The Whisker Weaving Algorithm: a Connectivity-Based Method for Constructing All-Hexahedral Finite Element Meshes", *Int. j. numer. methods eng.*, 39, 3327-3349 (1996).

3. R. Taghavi, "Automatic, Parallel and Fault Tolerant Mesh Generation from CAD", Engineering with Computers, 12, 178-185 (1996).

4. Somashekar Subrahmanyam, Michael Wozny, "An overview of automatic feature recognition techniques for computer-aided process planning", Computers in Industry 26 (1995) 1-21.

5. L. Kyprianou, "Shape classification in computer aided design", PhD Thesis, University of Cambridge, 1980.

6. S. Joshi, T.C. Chang, "Graph-based heuristics for recognition of machined features from a 3D solid model", Computer-Aided Design 20 (2) (March 1988) 58-66.

7. T.D. Blacker, M.B. Stephenson, J.L. Mitchiner, L.R. Phillips, Y.T. Lin, "Automated quadrilateral mesh generation: A knowledge system approach", ASME Paper No. 88-WA/CIE-4, 1988.

8. M.A. Price, C.G. Armstrong, M.A. Sabin, "Hexahedral Mesh Generation By Medial Surface Subdivision: Part I. Solids with Convex Edges", *Int. j. numer. methods eng.*, 38, 3335-3359 (1995).

9. Liu, S.-S. And Gadh, R. 'Basic LOgical Bulk Shapes (BLOBs) for Finite Element Hexahedral Mesh Generation' 5th International Meshing Roundtable, pp. 291-304 (Pittsburgh, PA, Oct. 10-11, 1996).

10. Liu, S.-S. And Gadh, R. 'Recursive Volume Decomposition to Support Hexahedral Mesh Generation' Submitted to Journal of Mechanical Design, ASME Transactions, area of Design Automation, November 1996.

11. Liu, S.-S. And Gadh, R. 'A Maximized Minimum Angle Approach for Quadrilateral Mesh Generation' technical report # I-CARVE-1996-18, December 1996.

12. Yu, X., Liu, S.-S., and Gadh, R. 'The Determination of Swept Objects to Support Hexahedral Mesh Generation' technical report # I-CARVE-1996-19, December 1996.

13. Gadh, R., Liu, S.-S., Lu, Y., and Kraftcheck, J. 'Finite Element Hexahedral Mesh Generation' technical report # I-CARVE-1996-20, December 1996.

14. Ted D. Blacker et al., "CUBIT Mesh Generation Environment, Volume 1: User's Manual", SAND94-1100, Sandia National Laboratories, Albuquerque, New Mexico, May 1994.

Report Number (14) SAND--97-2126C
CONF-9706106--

Publ. Date (11) 199709
Sponsor Code (18) DOE/DP, XF
UC Category (19) UC-700, DOE/ER

DOE