SAND2009-4021C

Embedding Features in a Cartesian Grid

Steven J. Owen and Jason F. Shepherd

[†]Sandia National Laboratories
sjowen@sandia.gov, jfsheph@sandia.gov

1 Abstract

Grid-based mesh generation methods have been available for many years and provide a relatively reliable method for meshing arbitrary geometries with hexahedral elements. One of its deficiencies, however, is the dramatically reduced mesh quality when resolving the boundaries of the geometric model. Because of this limitation, the principal use for these methods has been limited to biological-type models where sharp edges and curve definitions are not critical. While these applications have been effective, robust generation of hexahedral meshes on mechanical models impose difficulties that existing grid-based methods have not yet effectively addressed. This work introduces a set of procedures that can be used in resolving the features of a geometric model for grid-based hexahedral mesh generation for mechanical objects.

Keywords: grid-based, overlay grid, hexahedral mesh generation, topological equivalence

2 Background

The general problem of mesh generation involves discretizing a domain into simple shapes such as tetrahedra and hexahedra. In most cases the problem begins with a three-dimensional domain that is represented with topology and geometry. The topology representation can be as simple as a single surface or as complex as a mechanical assembly with hundreds of interconnected volumes, surfaces, curves and vertices related by means of a Boundary-representation (B-Rep) graph structure. In contrast the geometry representation defines the core mathematical foundation of the curves and surfaces and

[†]Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000

may be defined as a set of non-uniform rational b-splines or as a simple connected set of triangles. The B-Rep graph usually provides the frame on which the geometry is defined. Both geometry and topology should be considered when developing a mesh.

Most modern software tools that provide mesh generation capabilities enforce a requirement of *geometry-mesh* ownership. This provides a convenient method by which the user can apply physical properties and attributes to the topological features of the domain rather than dealing with the mesh itself. While attributes must ultimately be represented on the nodes and elements of the mesh for analysis, from a user's perspective, it is more convenient to assign attributes to the B-Rep entities rather than on individual mesh entities that may change over the course of a study.

To accomplish this, individual mesh entities, including, nodes, faces, edges and elements, must have a *child-parent* relationship with the B-Rep entities in the model: vertices, curves, surfaces and volumes. Likewise, B-Rep entities must have a *parent-child* relationship with the mesh entities that they own. This association, in most cases is a *one-to-many* link; that is, a single B-Rep entity may own multiple mesh entities, but a mesh entity can have only one unique parent. This one-to-many relationship has driven most of the modern meshing algorithms, where independent mesh entity groups can be generated for each individual B-Rep entity in the model and subsequently joined to form a contiguous mesh. Underlying the assumption of the B-Rep→mesh one-to-many relationship is that each B-Rep entity does indeed provide important information to the analysis and that a discrete representation of every B-Rep entity in the model will be represented in the mesh. Unfortunately, geometry creation procedures often developed in solids modeling tools frequently generate anomalous curves and surface that are not significant to the analysis. Using these models in a mesh generation system that enforces one-to-many ownership can lead to poor quality elements where the B-Rep will routinely over-constrain the resulting mesh. This work assumes that all entities in the geometric model are important to the analysis and that defeaturing or model simplification procedures such as those illustrated in the overview by Thakur et. al. [14] have already been accomplished.

The development of general-purpose unstructured hexahedral mesh generation procedures that effectively capture both geometry and topology for an arbitrary domain have been a major challenge for the research community. A wide variety of techniques and strategies have been proposed for this problem. It is convenient to classify these methods into two categories: *geometry-first* and *mesh-first*. In the former case, a topology and geometry foundation is used upon which a set of nodes and elements is developed. Historically significant methods such as plastering [1], whisker weaving [12] and the more the recent unconstrained plastering [11] can be considered *geometry-first* methods. These methods begin with a well defined boundary representation and progressively build a mesh that ensures that properties of mesh ownership are adhered to. Because these methods use the B-Rep entities themselves as the foundation

for the algorithm, the parent-child ownership of mesh entities generated is easily built in to the procedures. Most of these methods define some form of advancing front procedure that requires resolution of an interior void and have the advantage of conforming to a prescribed boundary mesh. Although work in the area is on-going, the ability to generalize these techniques for a comprehensive set of B-Rep configurations has proven a major challenge and has yet to prove successful for a broad range of models.

In contrast, the mesh-first methods start with a base mesh configuration. Procedures are then employed to extract a topology and geometry from the base mesh. These methods include grid-overlay or octree methods. In most cases these methods employ a Cartesian or octree refined grid as the base mesh. Because a complete mesh is used as a starting point, the interior mesh quality is high, however the boundary mesh produced cannot be controlled as easily as in geometry-first approaches. As a result the mesh may suffer from reduced quality at the boundary and can be highly sensitive to model orientation. In addition, grid-overlay methods may not accurately represent the topology and geometric features as defined in the geometric model. In spite of these inherent deficiencies, mesh-first methods have proven a valuable contribution to mesh generation tools for modeling and simulation. In contrast to geometry-first techniques, fully automatic mesh-first methods have been developed for some applications where boundary topology is simple or is not critical to the simulation. In particular, bio-medical models [17] [18] [3], metal forming applications [8] [4], and viscous flow [13] methods have utilized these techniques with some success. Automating and extending mesh-first methods for use with general B-Rep topologies would provide an important advance in hexahedral meshing technology.

As one of the first to propose an automatic overlay-grid method, Schneiders [8] developed techniques for refining the grid to better capture geometry. He utilized template-based refinement operations, later extended by Ito [3] and H. Zhang [16] to adapt the grid so that geometric features such as curvature, proximity and local mesh size could be incorporated. Y. Zhang [17] [18] and Yin [15] independently propose an alternate approach known as the Dual-contouring method that discovers and builds sharp features into the model as the procedure progresses. This is especially effective for meshing volumetric data where a predefined topology is unknown and must be extracted as part of the meshing procedure.

The dual contouring method for establishing a base mesh described by Y. Zhang [17] begins by computing intersections of the geometry with edges in the grid. Intersection locations are used to approximate normal and tangent information for the geometry. One point per intersected grid cell is then computed using Hermite interpolation from tangents computed at the grid edges. The base mesh in this case is defined as the dual of the Cartesian grid, using the cell centroids and interpolated node locations at the boundary. While attractive as a method for extracting features from volumetric data, it does

not guarantee capture of a pre-existing topology such as that contained in a CAD solid model.

Recent work on mesh-first approaches have focused more on the capturing of features of the geometry. A common thread among many of these methods [17] [3] [9] is the introduction of a buffer layer of hex elements to improve element quality near the boundary. This approach, while effective, still relies on a base mesh that is topologically equivalent to the features of a B-Rep. A drawback of many of these methods is that they tend to neglect the parent-child geometry-to-mesh ownership principals which are important for meshing algorithms to effectively engage with CAD-based modeling tools. With the assumption that all features of a B-Rep are indeed important in modeling the domain, the focus of this study is to propose a procedure whereby topological features can be acurately represented in a finite element mesh using a mesh-first method.

Shepherd [10] describes an approach to mesh-first hexahedral mesh generation utilizing geometric capture procedures. This work utilizes theory and assertions developed in [5] [6] . He asserts that a mesh must be topologically equivalent and geometrically similar to its geometry and B-Rep definition in order to develop a valid conformal mesh of the domain. To be topologically similar there must be a consistent correlation between the graph of the mesh and the graph of the B-Rep. This can be accomplished by establishing a one-to-many parent-child relationship between B-Rep and mesh entities. B-Rep entities of dimension r must contain a set of one or more contiguous mesh entities of dimension r, where $r = 0, 1, 2, 3$. Although not strictly required, geometric similarity between the mesh and the geometry of the domain is desired in order to maintain reasonable mesh quality. For example, aligning the base mesh with the principal orientations of curves and surfaces of the model will minimize the characteristic stair-step effect and increase mesh quality once mapped to the original geometry.

3 Feature Embedding

For convenience we have limited the base mesh for this study to a Cartesian grid. While it is often desirable to begin with an enriched octree grid or an aligned swept mesh, the Cartesian grid offers simplicity and automation that is easy to generalize for any model. For implementation purposes, a Cartesian grid is very light-weight and fast, avoiding full unstructured mesh data structures required for more general methods. While it is inevitable that mesh enrichment will be needed to more accurately capture small features and high valence vertices and curves, there is value in understanding the principles needed to embed topology through the use of a Cartesian grid. Indeed, It is expected that through careful application of topology embedding, that the need for mesh enrichment will be reduced.

Shepherd [10] outlines an algorithm which is convenient to use as the context for this work. He proposes first establishing a set of 2-manifolds on which a topology capture procedure is performed. A series of one or more buffer layers known as fundamental sheets are then inserted at the boundary, followed by a series of mesh optimization steps to improve mesh quality. This work will focus specifically on the topology capture procedure and leave the sheet insertion and mesh optimization procedures for a future study.

Establishing a 2-manifold on which topology is captured implies developing a base mesh and using the bounding set of quadrilaterals from a contiguous set of hexahedra on which vertex and curve topology is embedded. Shepherd suggests that tailoring a base mesh to the features and characteristics of the domain is advantageous. However, this procedure is not easily automated and would also be a valuable topic for future study. Instead, most current literature indicates that a base mesh is established from a Cartesian grid. In these cases individual hexes are tested for inclusion or exclusion based on common *in-out* procedures. In-out procedures classify each cell in the grid based upon its centroid's relative position with respect to the boundary of the domain. The continuous set of hexes that are completely contained within the geometry and optionally combined with hexes that are intersecting the domain are used. In many cases, mesh enrichment procedures using octree decomposition are used at this stage to ensure the geometry is effectively represented. In many cases special procedures are used to ensure non-manifold connections and non-contiguous regions within the hex mesh are eliminated.

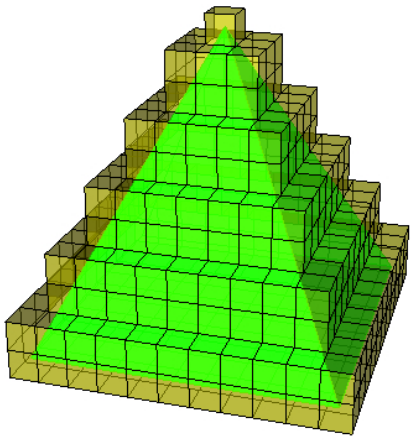

Fig. 1. Base grid defined from a pyramid geometry. Resulting 2-manifold does not provide rich enough mesh to capture 4-valent vertex at pyramid apex

Whether a base grid is defined using traditional in-out procedures or through a dual contouring procedure, topologic equivalence may not be ad-

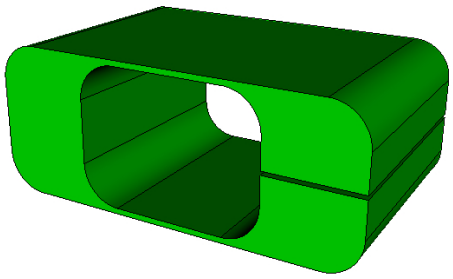

Fig. 2. B-Rep used to generate the base grids shown in figure 3

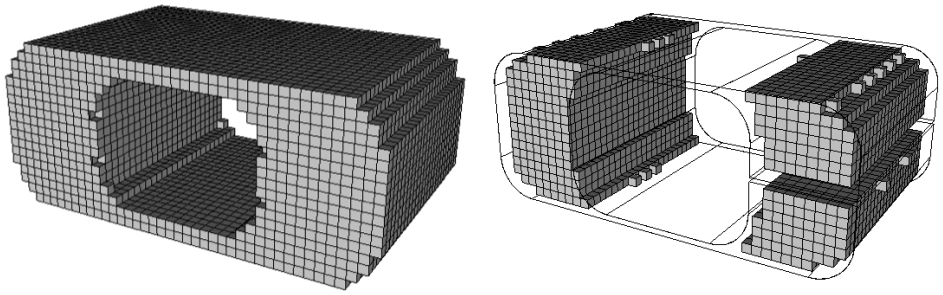

Fig. 3. Two different base grids defined from a C-shaped geometry (figure 2) using common in-out procedures. The grid on the left is defined from the inside and intersected hexes while the grid on the right is defined from only the hexes on the inside of the geometry. Neither satisfies the requirement of topologic equivalence.

equately taken into account. Figures 1 to 3 show examples where standard methods for defining a base grid may be inadequate. The single hexahedra at the apex of the pyramid in figure 1 provides a maximum of a 3-valent node on which to embed a 4-valent vertex. Without subsequent mesh enrichment, topology capture would be impossible at this location. Although Shepherd [10] proposes local pillowing operations to enrich the valence, it also requires an unstructured mesh data representation consequently reducing the efficiency of the proposed procedures and can also result in marginal quality hexahedra to achieve the required valence.

Figures 2 and 3 illustrate another potential issue with traditional base mesh definitions for mesh-first procedures. Neither the cells intersecting the geometry, nor the cells on the interior of the geometry produce a set of hexes that can build a topologically equivalent mesh without special procedures to combine the results from the two cases. In examples such as these, mesh enrichment strategies can be employed to locally refine the grid using template-based refinement techniques [3]. In order to provide complete generality, these methods typically require a *1-27* refinement strategy. That is, each hex in the

refinement region is decomposed into 27 hexahedra. In addition to introducing artificially high gradients on the local mesh size, transition elements, typically of marginal quality, are needed to ensure a conformal mesh between coarse and fine regions.

In an attempt to better control some of these issues, this work proposes using the full three dimensional Cartesian grid on which the B-Rep topology is progressively extracted. Initially limiting the topology extraction problem to a 2-manifold may over constrain the problem such that specialized procedures are required to enrich the grid where otherwise a 3-manifold approach would naturally extract topologic equivalence from the grid. However, it is clear that mesh enrichment strategies will still need to be employed, particularly for features significantly smaller than the grid cell size and for vertex valence greater than 6. We contend that the use of such procedures may be reduced by limiting specialized mesh enrichment procedures that impose high valent or marginal transition elements, ultimately improving overall mesh quality.

4 Embedding Procedures

Beginning from the solid model boundary representation of the model, the current implementation may first employ defeaturing as described by Quadros et. al. [7] and then extract the facets from the model to use as an approximate geometry representation. Geometry evaluation during the embedding procedures is then limited to evaluation of planar triangles and linear edges. A Cartesian grid that completely encloses the geometry with a user-defined resolution is then established. While there is no explicit requirement on grid cell size, for practical purposes the cells should be approximately smaller than the smallest feature size in the Geometry. The grid may also be optionally oriented so that a tight fitting bounding box is established to help align curves and surfaces with the principal axes of the grid. To maintain the advantages of a Cartesian grid, rather than transforming the grid itself, the geometry can be transformed into the Cartesian space during the embedding procedures and transformed back when complete.

The approach taken for embedding features follows roughly the bottom-up method of mesh generation, where successive dimensions are embedded starting from vertices and continuing through curves surface and volumes. If we define a Cartesian grid $\Omega_M = \{M_i^r | r = 0, 1, 2, 3\}$ and a B-Rep $\Omega_G = \{G_i^r | r = 0, 1, 2, 3\}$, our objective is to find groups of grid entities, M_i^r of dimension r that will be owned by corresponding B-Rep entities, G_i^r of the same dimension. Corresponding grid and B-Rep entities are enumerated in table 1. The embedding $E_{G \to M} \subseteq \Omega_M$ is defined as $E_{G \to M} = \{M_i^r : M_i^r \mapsto G_i^r\}$, where the non-unique mapping, $M_i^r \mapsto G_i^r$, is a set of procedures for each dimension that assigns grid entities to individual B-Rep entities. This mapping will necessarily be *collision-free*; that is, a grid entity, M_i^r may be mapped to one and only one B-Rep entity, G_i^r.

Table 1. Corresponding B-Rep and Cartesian grid entities

B-Rep Entity	Symbol	Cartesian Grid Entity	Symbol
Vertex	G^0	Node	M^0
Curve	G^1	Edge	M^1
Surface	G^2	Face	M^2
Volume	G^3	Cell	M^3

The embedding procedures $M_i^r \mapsto G_i^r$ for each dimension have similar characteristics. Each uses a combination of local geometric and topologic information to determine grid entities that will map to a given B-Rep entity. Traditional methods that may utilize in-out procedures followed by projections to nearby vertices and curves often fail as a result of not sufficiently taking into account the local B-Rep topology. Issues such as mismatched vertex valence, non-manifold connections or disjoint cell groupings may be eliminated by controlling the embedding procedures. The procedures generally proceed by looping through each B-Rep entity G_i^r for dimension r and capturing grid entities M_i^s for dimensions $s \geq r$. For example, the vertex embedding procedure, while incorporating node capture, also includes methods to capture nearby edge, face and cell entities. Similarly, the curve embedding procedure includes the capture of faces and cells. This has proven effective in avoiding or controlling collision conditions as the algorithm proceeds.

4.1 Embedding Vertices

Vertices can be embedded in most cases by simply finding the closest node in the grid to each vertex in the B-Rep. Collisions may occur where multiple vertices may claim the same grid node. Collision resolution in this case is handled by selecting the closest vertex to the node in question and successively assigning ownership to nearby grid nodes based on their relative distance.

Once vertex locations are assigned, grid edges at the selected grid node must be matched with appropriate curves. Figure 4 shows an example where the vertex G_0^0 at the apex of a pyramid maps grid node M_0^0. For this vertex, four grid edges must be selected that match the four curves that share the vertex. At this location, any of the curves $G^1_{i=0,1,2,3}$ may choose any one of the six grid edges $M^1_{j=0,1,..5}$ sharing grid node M_0^0. Clearly vertices with valence greater than six would not be permitted when utilizing a Cartesian grid. To facilitate edge selection at a selected grid node, the metric μ_{ij} is computed as follows:

$$\mu_{ij} = (\mathbf{V}_{M_j} \cdot \mathbf{V}_{G_i}) \left(\frac{\left|\mathbf{P}_{M_j} - \mathbf{P}_{G_i}\right|^{-2}}{\sum_{i=0}^{6} \left|\mathbf{P}_{M_j} - \mathbf{P}_{G_i}\right|^{-2}} \right) \tag{1}$$

where μ_{ij} is a value $-1 \leq \mu_{ij} \leq 1$ that represents how well curve G_i^1 matches geometrically with edge M_j^1 where $\mu_{ij} = 1$ is a perfect match and $\mu_{ij} = -1$

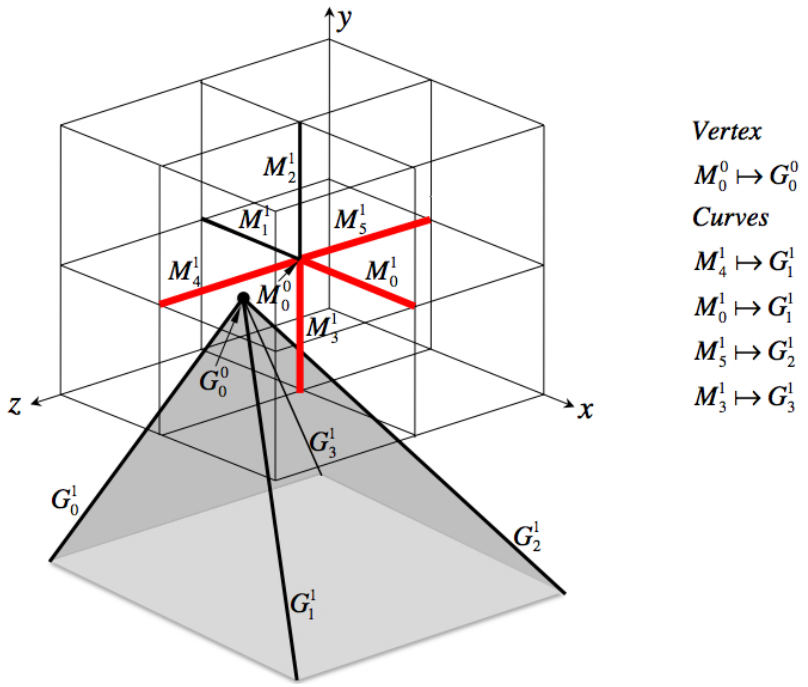

Fig. 4. A representation of the local grid near the apex of a pyramid. Four of the grid edges selected from $M^1_{j=0,1,\ldots 5}$ that are connected to the node M^0_0 must be paired to the four curves in the B-Rep $G^1_{i=0,1,2,3}$

is very poor. In equation 1 the variables $\mathbf{V}_{M_j}$ and $\mathbf{V}_{G_i}$ are the normalized outward pointing vectors of the grid edges M^1_j and curves G^1_i at the node M^0_0 and vertex G^0_0 respectively. The first term of this equation is simply the dot product of edge and curve vectors to indicate how well each grid edge is oriented with respect to a particular curve and is a value between -1 and 1. In practice, since the B-Rep is comprised of facets, the curve vector $\mathbf{V}_{G_i}$ can be approximated from the first facet edge that shares the vertex G^0_0 and curve G^j_i. The second term in equation 1 incorporates proximity information, where a normalized inverse distance weighted value between 0 and 1 is computed so that edges that are close are given a larger weight than those farther away. The vector $\mathbf{P}_{M_j}$ in this case is defined as the midpoint of edge M^1_j and the vector $\mathbf{P}_{G_i}$ is a point on curve G^0_i a distance $\frac{1}{2}length(M^1_j)$ from vertex G^0_0.

Once the metric μ_{ij} has been computed for each edge with respect to each curve at the vertex, all permutations of pairings between edges and curves are computed and the sum of μ_{ij} for each permutation is determined. The pairings can then be ranked from best to worst based on their μ_{ij} sums. For example, in figure 4 a ranking of all possible curve-edge pairings are compiled. Given 6 possible edges assigned to 4 curves, the total number of permutations

of edge pairings would be $6 \cdot 5 \cdot 4 \cdot 3 = 360$ permutations. For the worst case of a 5 or 6 valent vertex, the total number of permutations would be 6! or 720. The curve-edge pairing with the highest μ_{ij} sum is used as the candidate set of edges to be used to represent the curves at the vertex. For the example in figure 4 the selected candidate edge-curve pairings are shown at right. Note that multiple edge-curve permutations may yield identical μ_{ij} sums as in this simple example.

Even after selecting the best candidate edge-curve pairing for a given vertex, there is no guarantee that a valid topology that matches the local surface configuration can be established. For this reason a face-surface pairing procedure is also used. The best candidate edge-curve pairing is used as the starting point for this procedure. For any B-Rep graph of a three-dimensional domain, each vertex will have an equal number of edges and surfaces sharing the common vertex. Having defined edge-curve pairings, it remains to find a path of faces at the vertex between existing edge-curve pairs such that each surface is represented by at least one face.

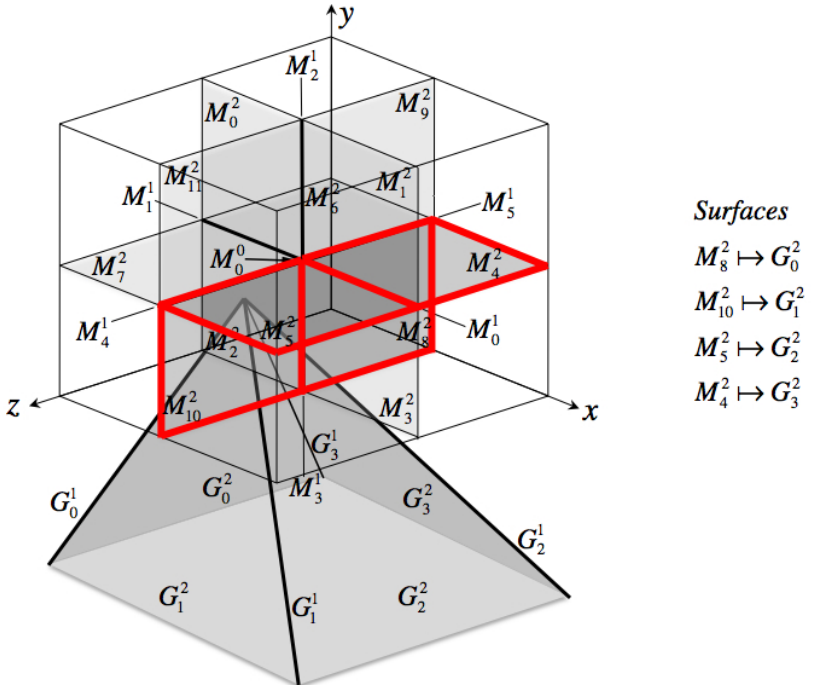

Fig. 5. A representation of the local grid near the apex of a pyramid. Four of the grid faces selected from $M^2_{j=0,1,\ldots 11}$ that are connected to the node M_0^0 must be paired to the four surfaces in the B-Rep $G^2_{i=0,1,2,3}$

Figure 5 shows the same example of a pyramid apex and the local grid topology at node M_0^0. Also shown are the faces $M^2_{j=0,1,...11}$ sharing the node. For this problem we use the cyclic ordering of curves and surfaces at the vertex to guide a traversal. Staring with curve G_0^1, its corresponding grid edge (in this case M_4^1) is selected to begin the traversal. Using the B-Rep topology, we know that surface G_1^2 is the next counter-clockwise surface adjacent curve G_0^1; which in turn shares the curve G_1^1 that is associated with grid edge M_0^1.

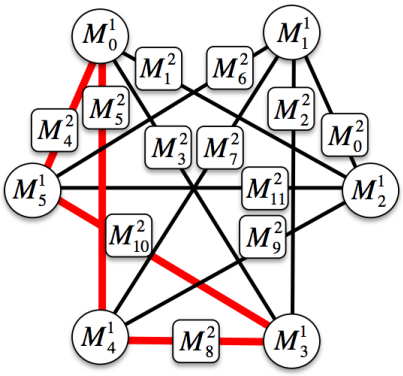

Fig. 6. The graph of local faces and edges at a node illustrating all possible paths of faces that can be selected representing the set of surfaces around a vertex. The path used in figure 5 is highlighted.

To find the face(s) at node M_0^0 that will map to surface G_1^2 we must find a set of faces bounded by edges M_4^1 and M_0^1. For this case, the choice is relatively trivial, as face M_5^2 appears to satisfy our criteria. For the general case, however there are several solutions that will satisfy this criteria. For example the set of faces $\{M_7^2, M_6^2, M_4^2\}$, $\{M_{11}^2, M_1^2\}$ and $\{M_{10}^2, M_3^2\}$ are each bounded by edges M_4^1 and M_0^1. Figure 6 is a representation of the edge-face graph at any node in a Cartesian grid. Using this connectivity information, a complete set of unique paths can be derived between any two edges. A metric m_{ij}, where $0 \leq m_{ij} \leq 1$, can be computed for each possible path $P_{ij}^2 \subseteq M^2_{j=0,1,...11}$ based on the following criteria:

$$m_{ij} = \frac{1 + 0.5(\mathbf{T}_{G_i^2} \cdot \mathbf{T}_{P_{ij}^2})}{1 + \left| J_i - N_{P_{ij}^2} \right|} \tag{2}$$

where $\mathbf{T}_{G_i^2}$ is the normalized outward pointing tangent vector at the common vertex G_i^0 and in the plane of the surface G_i^2 that bisects curves G_i^1 and G_{i+1}^1; $\mathbf{T}_{P_{ij}^2}$ is the average normalized outward pointing tangent vector at node M_i^0 to the faces in P_{ij}^2; $N_{P_{ij}^2}$ is the number of faces in P_{ij}^2 and J_i is defined as:

$$J_i = \begin{cases} 1, \theta_i \leq \frac{3\pi}{4} \\ 2, \frac{3\pi}{4} \leq \theta_i \leq \frac{5\pi}{4} \\ 3, \theta_i > \frac{5\pi}{4} \end{cases} \tag{3}$$

where J_i is an integer that represents the ideal number of faces that should represent surface G_i^2 given the angle θ_i between curves G_i^1 and G_{i+1}^1 on the surface. Equation 2 computes the relative alignment between the path P_{ij}^2 and the surface G_i^2 and assigns a penalty for paths that are longer or shorter than the ideal path defined by J_i.

A suitable path P_i^2 can now be selected that effectively matches surface G_i^2 by selecting the path P_{ij}^2 with the maximum m_{ij}. Subsequent paths can also be computed in a similar manner. The fact that edges and faces can only be used once, may block an optimal path between any given pair of edges, however alternate paths can generally be found. The final set of paths are represented in figure 6 as thicker lines and are illustrated as faces in figure 5.

Once a valid set of faces and edges have been determined using the preceding algorithm, there is still no guarantee that an optimal solution has been found. Given the fact that equation 1 may yield identical or similar metrics μ_{ij} for different permutations of edge-curve pairings, it may be necessary to check multiple edge-curve pairings by determining their associated face-surface pairings. This can be done by keeping track of the sum of m_{ij} for the face-surface pairings associated with each edge-curve pairing. For most cases, a limited number of edge-curve pairings will need to be tested before an optimal is determined.

The final step in matching grid topology at a vertex is to capture the grid cells that will be inside the volume. In practice this is done by finding the face that is well-aligned with its associated face. We can determine alignment based on the dot product of the face with the surface normal. The surface normal can then determine which side of the face is defined as inside and which is defined as outside. By selecting the inside cell and traversing to neighboring cells sharing the node M_i^0 that are on the same side as the inside cell, all cells at the node inside the volume can be selected.

An example of topology captured at the vertices of the simple pyramid problem is shown in figure 7.

4.2 Embedding Curves

The next step of the procedure involves embedding curves into the grid topology, or specifically the set $E_{G \to M} = \{M_i^1 : M_i^1 \mapsto G_i^1\}$. We can gather a set of grid edges for each curve by starting from the grid edge associated with the curve at each vertex that we determined in the previous step and finding a collision free path between the start and ending grid edges on the curve. For this procedure, knowing the edge at position k on the curve, we can determine the next grid edge at position $k+1$ by examining the 5 connected edges at its end. Equation 1 can then be used to select the best next edge in the path.

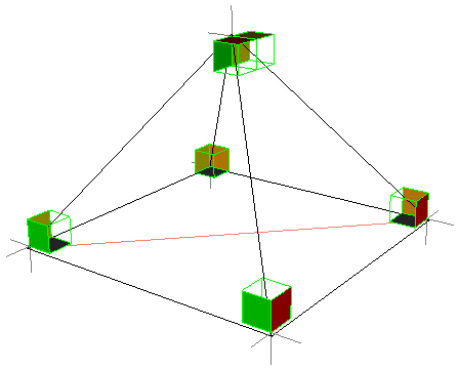

Fig. 7. The grid topology captured for vertices of a simple pyramid model is illustrated. The procedure described here has been used to select the appropriate grid topology.

For this case the point $\mathbf{P}_{G_i}$ is determined by projecting the mid point of the edge to the curve. The vector $\mathbf{V}_{G_i}$ is the tangent vector of the curve at $\mathbf{P}_{G_i}$.

Some of the possible edge selections may be eliminated by examining the end node of each of the edges to see if they are already in use. While this will avoid collisions, in some cases a non-optimal path may be selected. Possible alternative paths may be generated by starting from opposite ends of the curve or by changing the order in which curves are processed. Optimal curve paths may be generated by minimizing the direction changes between grid edges representing the curve, and attempting to modify the edge-curve pairing of those curves where μ_{ij} is small while attempting to maximize the average μ_{ij}. This may require multiple iterations to define an optimal curve path.

Figure 8 left shows edge paths representing the curves for the pyramid problem. Where curves are not naturally aligned with one of the coordinate axis, multiple direction changes between grid edges are unavoidable, which will ultimately reduce final mesh quality.

Another important aspect of the curve capture problem is selecting the cells at the curve that will be interior to the volume. Knowing the orientation and angle between surfaces at the curve, we can attempt to control the number and position of each grid cell so that final mesh topology will match as close as possible to the B-Rep. Figure 9(a) illustrates two edges M_k^1 and M_k^1 that have been selected to represent a segment on a curve G_i^1. The four shaded cells adjacent edge M_k^1 represent the possible choices for cells where at least one and no more than three cells must be selected to represent the interior of the volume at the curve. Figure 9(b) shows the 4 cells adjacent edge M_k^1. J_i from equation 3 may be used to determine the number of cells to select where θ_i is the interior angle formed by adjacent surfaces G_i^2 and G_{i+1}^2 at the curve. If we compute vector $\mathbf{B}_i$ as the bisecting vector at edge G_i^1 projected onto the plane normal to edge M_k^1, then the quadrant in which $\mathbf{B}_i$ falls with

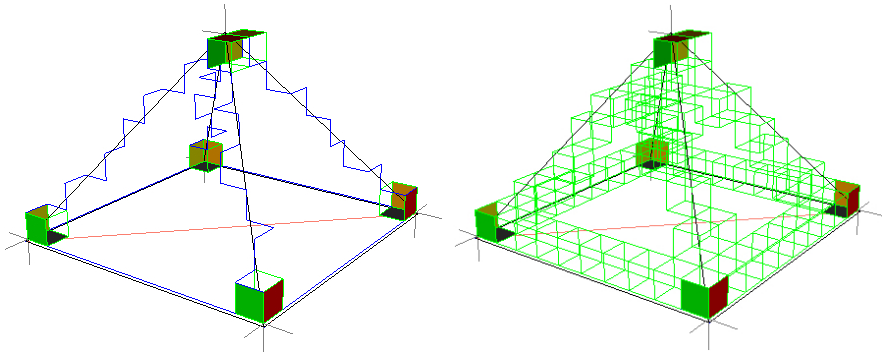

Fig. 8. Continuing with the embedding procedures, the figure on the left illustrates the edges selected to represent each of the curves in the model. On the right, the cells adjacent each of these edges has been selected and illustrated here.

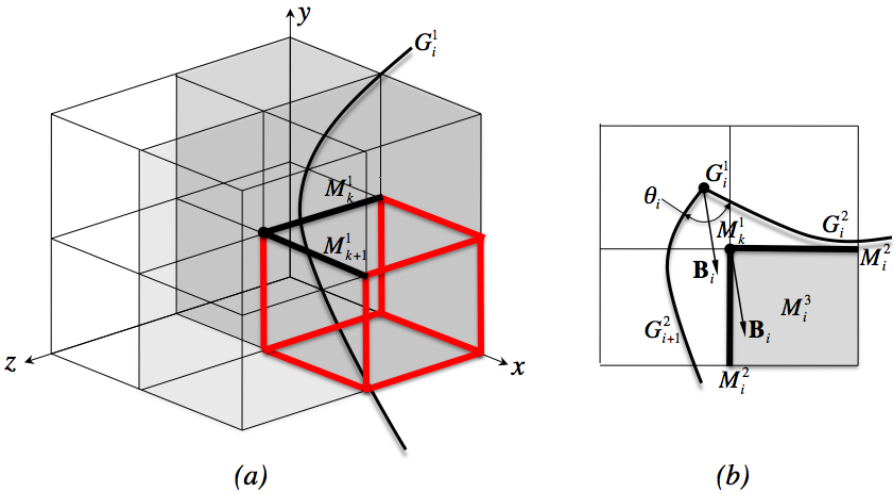

Fig. 9. (a) Representation of the curve G_k^1 in the geometry and two edges M_k^1 and M_{k+1}^1 that have been associated with the curve. The cell selected at the edges representing the interior of the volume is highlighted. (b) The side view of the four cells adjacent the same edge M_k^1 along with the curve G_i^1 and its adjacent surfaces. The angle θ_i between the surfaces and its bisecting vector $\mathbf{B}_i$ are used to determine which of the four cells at the edge will be captured.

respect to the four cells adjacent edge M_k^1 will determine which initial cell M_i^3 to select. Where $J_i > 1$, then one or two additional cells adjacent the cell M_i^3 at edge M_k^1 may be selected based on the relative orientation of $\mathbf{B}_i$ within the cell. Once cells at the edge have been selected, then faces M_i^2, also shown in figure 9(b), can be selected and associated with their respective surfaces G_i^2 and G_{i+1}^2

Another issue which must be resolved are potential collisions between cells as the algorithm proceeds. This is common especially at adjacent edges where a 90 degree turn is required to better capture geometry. This is illustrated in figure 9(a), where edge M_k^1 is oriented 90 degrees to M_{k+1}^1. In this case it is not uncommon for $\mathbf{B}_i$ and J_i to be computed such that the cells selected adjacent M_{k+1}^1, conflict with those selected adjacent M_k^1. Where conflicts arise, they are resolved by using the cell selection from the edge whose tangent most closely aligns with that of the curve at its closest point.

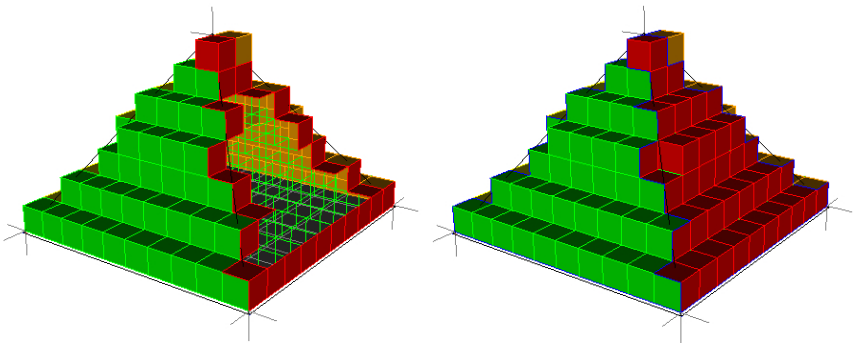

Fig. 10. Continuing with the same pyramid example, the figure at left illustrates the faces that have been captured near the curves of one of the surfaces. The figure on the right illustrates the final grid topology with all features embedded.

4.3 Embedding Surfaces

The next stage of the algorithm is to ensure that a continuous set of faces M_i^2 is associated with their respective surfaces G_i^2. For this we seek the set of faces $E_{G\to M} = \{M_i^2 : M_i^2 \mapsto G_i^2\}$. Since both vertex and curve embedding procedures also included selection of adjacent faces, we can begin with the assumption that at least one continuous layer of faces has been captured at each surface. This is illustrated in figure 10 where the image on the left shows one of the surfaces with only the faces near the curves that have been captured.

This procedure can be compared to an advancing front algorithm where boundary faces are first captured and interior faces to the surfaces are progressively discovered and added to the surface until a continuous set of grid faces have been established that represent the surface. It is advantageous to

order the procedure such that loops of faces progressively advance towards the interior of the surface until they close on themselves. Figure 11 illustrates the procedure for advancing a single front where a current front edge defined by M_i^1 has adjacent face M_i^2 which has already been associated with its parent surface G_i^2. The objective here is to determine which of the three faces, illustrated as $M^2_{i\rightarrow up}$, $M^2_{i\rightarrow forward}$, or $M^2_{i\rightarrow down}$, should be selected as the next face onto which the front will advance. The candidate face can be selected by computing μ_{ij} using equation 1 as we did for edges. In this case the vectors $\mathbf{V}_{M_j}$ and $\mathbf{V}_{G_i}$ are the normal vectors of the grid face and surface, and the points $\mathbf{P}_{M_j}$ and $\mathbf{P}_{G_j}$ is the midpoint of the face and its projection to the surface G_i^2 respectively.

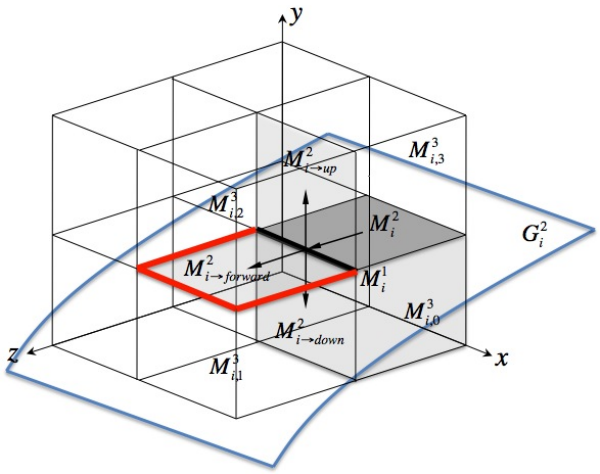

Fig. 11. The cells adjacent a grid edge and face M_i^2 that has been associated with the surface G_i^2 is illustrated to represent the current front of the advancing front procedure. The next face in the surface will be selected from the three faces $M^2_{i\rightarrow up}$, $M^2_{i\rightarrow forward}$ or $M^2_{i\rightarrow down}$

The selection of the next front is also effected by the placement of the existing cells associated with the volume. From our definition of the cells captured at curves, we will always have a volume cell $M^3_{i,0}$ immediately adjacent the face M_i^2. It is however conceivable, that the cell $M^3_{i,1}$ has already been selected for the volume. Should this case arise, then the face $M^2_{i\rightarrow down}$ would be eliminated as a candidate for the next advancement. Likewise should $M^3_{i,2}$ and $M^3_{i,1}$ be in use, the only candidate for advancement would be $M^2_{i\rightarrow up}$. To ensure non-manifold connections are not created, we would ensure that cases where only $M^3_{i,0}$ and $M^3_{i,1}$ are selected at any grid edge are not permitted. Similarly, we ensure that the case where cells $M^3_{i,0}$ and $M^4_{i,1}$ are both used by

the volume would not be permitted, as this would create a condition where a hanging or dangling face would exist in the volume.

Also part of this procedure is the selection of the cells adjacent new faces as the algorithm progresses. Cells can be selected by choosing from the two adjacent cells at the new face. Depending on which of the three candidate faces is selected, zero, one or two new cells may be added to the volume with each new face added to a surface.

4.4 Embedding Volumes

The final step of the algorithm is to ensure that all remaining cells M_i^3 interior to the volume G_i^3 have been captured and appropriately assigned. Similar to other procedures this can be represented by the mapping $M_i^3 \mapsto G_i^3$. At this stage, all surfaces will have been captured and have exactly one adjacent cell. The remaining cells can now be captured by selecting one known interior cell and recursively traversing to collect all cells that are bounded by the grid faces that are associated with surfaces. All cells inthe Cartesian grid not selected as part of this procedure are discarded. An example of a base grid with embedding procedures completed is shown in figure 12 where the different colors/shades represent the separate captured surfaces in the Cartesian grid.

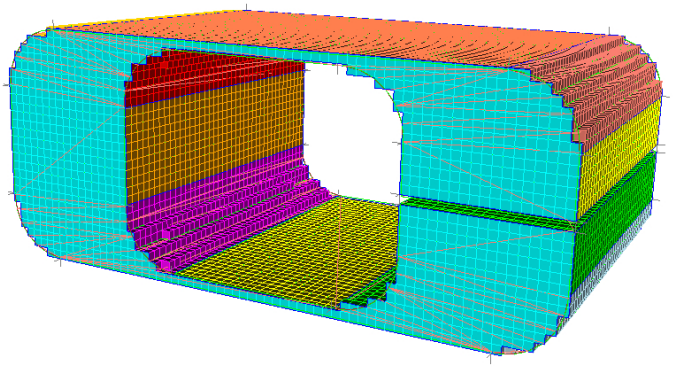

Fig. 12. Left: A completed embedding procedure is illustrated using the C-geometry model shown in figure 2. Note that topological equivalence is maintained through the thin sections and narrow gap.

5 Completing the Mesh

The embedding procedures discussed in this work focus specifically on providing a base mesh that is topologically equivalent to a given boundary represen-

tation. Once this is achieved, subsequent sheet insertion and mesh optimization steps are employed to build a final mesh as described by Shepherd [10]. Up to this point in the procedure, a Cartesian grid has been used because of its efficiency and low memory requirements. Sheet insertion procedures, however, require a full unstructured mesh data representation to insert appropriate layers of hexes. As a result, the base mesh is transferred to an unstructured data representation before sheet insertion and mesh optimization algorithms are executed. A simple implementation of a completed mesh on the example C-Geometry model illustrated in figure 3 is shown in figure 13.

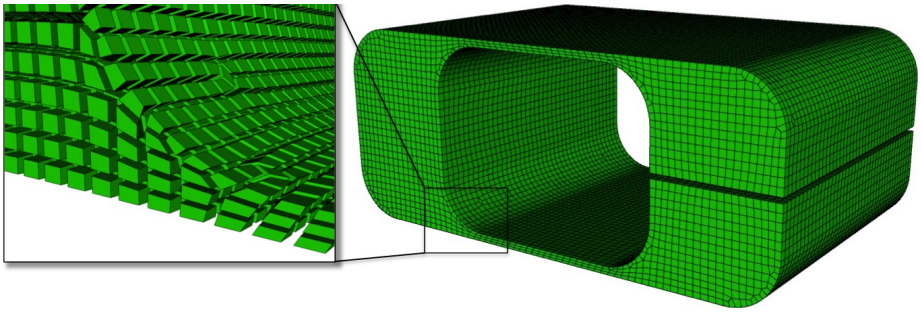

Fig. 13. The mesh is completed by inserting sheets or layers at the boundaries and smoothing. A closeup of the exploded mesh is shown to illustrate the inserted layers.

6 Assembly Models

Not yet addressed in this work is the extension of the proposed embedding procedures to multi-volume or assembly models. Although not yet implemented at this writing, extensions to multi-volume problems would fit well within the context of the procedures already proposed. The assumption here is that the B-Rep graph is represented as a cellular topology structure. In other words, adjacent volumes would share common curves and surfaces and prohibit any overlapping geometric entities. Extension to multi-volume models would primarily involve generalizing the cell selection procedures at vertices, curves and surfaces. The following changes and additions to the procedures would need to be accomplished:

1. Vertex embedding: The current method incorporates a graph search to determine the optimal path between edges at a node so that at least one face is paired with each surface at the vertex. With multiple volumes sharing a single vertex, a complete set of paths would need to be established for each volume sharing the vertex. Each set of surfaces common to a single volume at the vertex will define a compete loop of faces at the node.

Cell selection at a vertex would also need to be enhanced so that at least one cell is paired with each volume sharing the vertex. Using a Cartesian grid, and without using mesh enrichment, the maximum number of volumes sharing an interior vertex would be eight. Similarly, a vertex at a boundary would be limited to seven adjacent volumes.

2. Curve embedding: Extension to multiple volumes would include selection of at least one grid cell at each grid edge assigned to a curve for each volume sharing the curve. Orientation of cells for each volume would also be defined for each volume as represented in figure 9. Limits on numbers of volumes sharing a common curve for a Cartesian grid would be four for interior curves and three for curves at a free boundary.
3. Surface embedding: The advancing front procedure for selecting faces to associate with a specific surface would be unchanged for the multi-volume problem. Cell selection at a face would now incorporate selecting and assigning both adjacent cells for the adjacent volumes rather than a single cell.
4. Volume embedding: With a complete set of faces, edges, and nodes associated with the B-Rep topology successfully assigned, the interior cells associated with each volume can be recursively enumerated by starting with a known interior cell and traversing to neighboring cells in the same manner as the single volume problem.

7 Conclusion

Hexahedral methods that begin with a Cartesian grid as a base mesh are attractive because of their potential for automation. Their primary use, however has been for bio-medical applications that do not include significant topology. These methods have not been as effective for topology rich models such as those common to three-dimensional CAD solid models described by a boundary representation. While some work has been done to ensure curves and surfaces are captured in a Cartesian-based mesh, this work proposes a new systematic approach to ensure that features of a B-Rep model are adequately represented in a final hexahedral mesh.

This work introduces an approach to embedding features in a Cartesian grid. Without such procedures there is no guarantee that important features defined in a CAD model will be captured in the simulation model. Using traditional in-out procedures can often neglect topological equivalence, resulting in non-manifold or disjoint cells, as well as insufficient local node valence for a given B-Rep topology. Current methods generally resolve these issues by using mesh enrichment strategies that can introduce highly refined and poorly shaped elements. The embedding procedures described here build a base mesh from a Cartesian grid that is intended to meet topological equivalence requirements of a B-Rep while reducing the need for mesh enrichment.

While a Cartesian grid provides efficiency and ease-of-use, for complete generality, it is clear that mesh enrichment strategies will need to be applied. However it is expected that the proposed embedding procedures will limit the use of such strategies. Future work will need to extend these procedures to combine mesh enrichment with topology embedding to ensure that topological equivalence is maintained for an arbitrary B-Rep configuration.

In contrast to traditional in-out procedures for generating a base mesh, using the proposed embedding algorithms also provides more control over mesh topology, such that numbers of hexes placed at curves and vertices can better account for local geometry. Future work will study how sheet insertion procedures can work together with embedding algorithms to better control placement of sheets to maximize element quality.

The topology embedding problem, while a key component of automatic hexahedral meshing, is clearly only one part of a fully automatic procedure. Careful insertion of boundary sheets as well as projection and mesh optimization methods are necessary to provide a robust, quality hexahedral mesh. Ongoing work to couple the proposed procedures into a fully automatic hexahedral meshing algorithm for topology rich models is still necessary and under development.

References

1. Blacker TD, Meyers RJ (1993) Seams and Wedges in Plastering:A 3D Hexahedral Mesh Generation Algorithm, Engineering with Computers, 2(9):83–93
2. CD-Adapco (2009) http://www.cd-adapco.com/
3. Ito Y, Shih AM, Soni BK, (2009) Octree-based reasonable-quality hexahedral mesh generation using a new set of renement templates, International Journal for Numerical Methods in Engineering, 77(13):1809–1833
4. Kwak DY, Im YT (2002) Remeshing for metal forming simulationsPart II: Three-dimensional hexahedral mesh generation, International Journal for Numerical Methods in Engineering, 53:2501–2528
5. Ledoux F, Shepherd JF (2009) Topological and geometrical properties of hexahedral meshes. Submitted to Engineering with Computers
6. Ledoux F, Shepherd JF (2009) Topological modications of hexahedral meshes via sheet operations: A theoretical study. Submitted to Engineering with Computers
7. Quadros WR, Owen SJ (2009) Geometry Tolerant Mesh Generation, Submitted to 18th International Meshing Roundtable
8. Schneiders R, Schindler F, Weiler F (1996) Octree-based Generation of Hexahedral Element Meshes, In: Proceedings of the 5th International Meshing Roundtable, 205–216
9. Shepherd JF (2007) Topologic and Geometric Constraint-based Hexahedral Mesh Generation, PhD Thesis, University of Utah, Utah
10. Shepherd JF (2009) Conforming Multi-Volume Hexahedral Mesh Generation via Geometric Capture Methods, Sandia National Laboratory Report, SAND2009-3382

11. Staten ML, Kerr RA, Kerr, Owen SJ, Blacker TD (2006) Unconstrained Paving and Plastering: Progress Update, In: Proceedings, 15th International Meshing Roundtable 469–486
12. Tautges TJ, Blacker TD, Mitchell SA (1996) The Whisker Weaving Algorithm: A Connectivity-Based Method for Constructing All-Hexahedral Finite Element Meshes, International Journal for Numerical Methods in Engineering, 39:3327–3349
13. Tchon KF, Hirsch C, Schneiders R (1997) Octree-based Hexahedral Mesh Generation for Viscous Flow Simulations, American Institute of Aeronautics and Astronautics A97-32470 781–789
14. Thakur A, Banerjee AG, Gupta SK (2009) A survey of CAD model simplification techniques for physics-based simulation applications, Computer Aided Design, 41(2) 65–80
15. Yin J, Teodosiu (2008) Constrained mesh optimization on boundary, Engineering with Computers 24:231-240
16. Zhang H, Zhao G (2007) Adaptive hexahedral mesh generation based on local domain curvature and thickness using a modied grid-based method, Finite Elements in Analysis and Design, 43:691–704
17. Zhang Y, Bajaj CL (2006) Adaptive and Quality Quadrilateral/Hexahedral Meshing from Volumetric Data. Computer Methods in Applied Mechanics and Engineering 195: 942–960
18. Zhang Y, Hughes TJR, Bajaj CL (2007) Automatic 3D Mesh Generation for a Domain with Multiple Materials. Proceedings of the 16th International Meshing Roundtable 367–386