H-MORPH: AN INDIRECT APPROACH TO ADVANCING FRONT HEX MESHING

STEVEN J. OWEN[1,2,*] AND SUNIL SAIGAL[1]

[1]*Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania, U.S.A.*
[2]*Sandia National Laboratories, Albuquerque, NM. U.S.A.*

ABSTRACT

H-Morph is a new automatic algorithm for the generation of a hexahedral-dominant finite element mesh for arbitrary volumes. The H-Morph method starts with an initial tetrahedral mesh and systematically transforms and combines tetrahedra into hexahedra. It uses an advancing front technique where the initial front consists of a set of prescribed quadrilateral surface facets. Fronts are individually processed by recovering each of the six quadrilateral faces of a hexahedron from the tetrahedral mesh. Recovery techniques similar to those used in boundary constrained Delaunay mesh generation are used. Tetrahedra internal to the six hexahedral faces are then removed and a hexahedron is formed. At any time during the H-Morph procedure a valid mixed hexahedral-tetrahedral mesh is in existence within the volume. The procedure continues until no tetrahedra remain within the volume, or tetrahedra remain which cannot be transformed or combined into valid hexahedral elements. Any remaining tetrahedra are typically towards the interior of the volume, generally a less critical region for analysis. Transition from tetrahedra to hexahedra in the final mesh is accomplished through pyramid shaped elements. Advantages of the proposed method include its ability to conform to an existing quadrilateral surface mesh, its ability to mesh without the need to decompose or recognize special classes of geometry, and its characteristic well-aligned layers of

* Correspondence to: Steven. J. Owen, Sandia National Laboratories, Box 5800, M.S. 0847, Albuquerque, NM, 87185-0847, U.S.A. E-mail: sjowen@sandia.gov. Phone: 505-284-6599. Fax: 505-844-9297

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

DISCLAIMER

elements parallel to the boundary. Example test cases are presented on a variety of models.

1 INTRODUCTION

A wide variety of methods have been proposed in the literature to deal with the unstructured hexahedral mesh generation problem. Amongst these methods only a limited number apply for an arbitrary shaped volume and conform to a prescribed surface mesh. Mapping[1,23], sub-mapping[4] and sweeping[5,6,7] methods are generally very fast and efficient, but can only deal with a limited class of geometry. Applicability of mapping and sweeping can be extended when the geometry is decomposed, either manually or automatically[8,9]. Other, more general algorithms, such as medial surface[10,11] and grid-based[12,13] methods, while applicable to a wider class of geometry, still have the limitation that they do not conform to a prescribed surface mesh. The ability to conform to a prescribed surface mesh is important where an incremental construction of the finite element model is employed. As each volume is meshed, interfacing elements must conform from one volume to the next. This is particularly important in the context of large assemblies, where distinct components of a final assembly may be modeled by different analysts, later to be merged into a final model.

Both the plastering[14,15] and the whisker-weaving[16,17] algorithms satisfy the criteria of arbitrary volumes and conforming to a prescribed surface mesh. Recent work[18] has shown, however, that the plastering algorithm cannot adequately resolve internal voids. Frequently, as the hexahedral elements advance to the interior of the volume, voids may be generated that are impractical to mesh with hexahedra. Mitchell[19] has shown that the minimal condition for an all-hexahedral mesh is for the surface mesh to contain an even number of quadrilaterals. Mitchell's condition, however, does not address the resulting quality of the hexahedra that might form such a mesh.

Recognizing the difficult problem of resolving an arbitrary volume into hexahedra, while maintaining a prescribed surface mesh, the whisker weaving algorithm attempts to address the problem from a global view of the topology rather than from the local view inherent in the plastering algorithm. This global view of the topology has

been invaluable in providing a better understanding of the all-hex problem. In spite of this new understanding, it remains to be seen if a whisker weaving approach can adequately decompose an arbitrary volume into hexahedra, while maintaining reasonable quality elements. Recent promising results from Mitchell[20] have shown significant advancements in the applicability and success of the whisker weaving algorithm.

The difficulties associated with the automatic generation of a boundary-constrained all-hexahedral mesh, has given rise to the concept of the hex-dominant mesh. If the majority of the mesh can be filled with hexahedra, and a relatively small volume filled with tetrahedra, it is considered adequate for many applications. The present work is directed towards such developments. A new method for generating a hex-dominant mesh for arbitrary volumes is presented. Some of the algorithms discussed are a direct extension of the Q-Morph algorithm introduced by Owen *et al.*[21]. The algorithm starts with a tetrahedral mesh and systematically transforms the tetrahedra into hexahedra. At any instant during the algorithm, a valid mixed hex-tet mesh exists. At some point, when reasonably shaped hexahedra can no longer be placed within the volume, the algorithm stops. The remaining tetrahedra in most cases are within the interior of the mesh, away from important boundary conditions, or critical features which may be important to the analysis. As an advancing front method, the H-Morph algorithm has many characteristics in common with the plastering[14,15] algorithm. Similar to plastering, H-Morph defines an initial front composed of the quadrilaterals on the surface and systematically projects new elements towards the interior of the volume in an attempt to completely fill the volume with hexahedra.

In the process of construction of new elements, a significant part of plastering involves checking for intersections and resolving closure issues. Recognizing the inherent problem with closing the void for an arbitrary volume, Meyers *et al.*[22] and Tuchinski and Clark[18] have developed a hex-dominant method (Hex-Tet) based on the plastering algorithm. Similar to H-Morph, the plastering continues until hexahedra can no longer be formed, resolving the remaining interior void with tetrahedra and pyramid elements. Although they show significant progress in developing a robust hex-dominant method, research is still on going. The significant difference between the H-Morph algorithm and Hex-Tet is the medium through which the hexahedra are generated. Hex-

Tet can be classified as a *direct* method, placing elements directly within the volume. H-Morph, on the other hand, is an *indirect* method relying on an initial set of tetrahedral elements placed in the volume, which it utilizes to guide its progress. The advantage in the latter case is the considerable benefit afforded in avoiding the significant process of intersection and closure resolution inherent in plastering, a somewhat error-prone operation. Where intersections are inadvertently missed, the plastering algorithm can have the undesired peculiarity of *overmeshing* itself, essentially filling space with hexahedra. Another advantage afforded by H-Morph is its tendency to maintain a valid finite element mesh throughout the mesh generation process. Starting with well-shaped tetrahedra, the algorithm merely transforms and removes tetrahedra as it proceeds, replacing them with hexahedral shaped elements. The algorithm could, in effect, stop at any point resulting in a valid finite element mesh. This is not the case with plastering. In addition, because the remaining voids, after plastering is complete, can be particularly complex, filling the remaining void with tetrahedra can also be a difficult task, frequently resulting in failure. Since the interior of the mesh is always filled with tetrahedra in the H-Morph algorithm, no such problem exists.

2 OVERVIEW OF H-MORPH

A brief outline of the fundamental steps involved in the H-Morph algorithm is given here. Further elaboration of these steps is given in a subsequent section.

1. **Surface Quadrilateral Mesh.** An all-quadrilateral mesh is generated on the boundary of a solid model. This can be accomplished using the Q-Morph[21] procedure. An existing set of quadrilateral faces from an adjacent volume may be alternately used.
2. **Triangles from Quads.** Each quadrilateral is divided into two triangles, keeping track of the initial quadrilaterals.
3. **Initial Tetrahedral Mesh**. The boundary triangulation is passed on to a tetrahedral mesh generator where a boundary constrained tetrahedral mesh is generated.

4. **Front Definition.** An initial set of fronts is generated. Each front is composed of a quadrilateral on the initial surface mesh. Additionally, each front is associated with two triangles, which are in turn associated with tetrahedra on the interior of the mesh.

5. **Front Face Classification.** State lists are generated and each front is classified according to its current *state*. A state reflects the current status of completing the six faces of a hexahedron with respect to any individual front. It is based on angles to adjacent fronts at its edges.

6. **Front Face Processing.** Each front is individually processed. This phase of the H-Morph process comprises the main algorithmic loop. Figure 1 illustrates the process, where vertices *ABCD* define the current front being processed. The following briefly summarizes the steps involved to process a single front.

   a) **Seams.** Where angles between adjacent fronts are less than a prescribed angle ε, a seam operation is performed.

   b) **Topology Check.** A check is made to ensure that the surrounding fronts can topologically form a hexahedron. The current front is not processed when local topological conditions exist that prohibit a hexahedron from being formed.

   c) **Cleanup.** If the current front has previously been processed and has failed for any reason, local improvement operations are performed on the tetrahedra close to the front. If this front is the start of a new level, then cleanup and smoothing operations are performed on all remaining tetrahedra.

   d) **Face Construction.** At each edge on the front that does not currently have an adjacent front where the angle, α, between fronts is less than $3\pi/4$, a quadrilateral is formed from the internal tetrahedral mesh. This step involves three main algorithmic steps.

      i. **Edge Determination.** Quadrilateral edges are retrieved or formed from the tetrahedra. Starting with front *ABCD* in Figure 1(a), a single edge *BE* may be simply selected from the adjoining tetrahedra as in Figure 1(b) or more complex local transformations may be performed.

ii. **Closure Resolution.** Where an opposing front is detected at the selected edge, closure resolution checks are used to maintain valid local topology.

iii. **Edge/Face Recovery.** Edges and faces are recovered from the tetrahedra. Two triangular facets are recovered for each quadrilateral face of the hexahedra. Figure 1(c) shows the two triangles *ABF* and *BEF* formed from the tetrahedra.

The two new triangles form the basis of a new quadrilateral where a new front structure is defined and associated with the new quadrilateral face. Step (d) is repeated for each side of the current front that requires a quadrilateral face as shown in Figure 1(d).

e) **Top face construction.** A top quadrilateral is formed to complete the hexahedral shape by recovering triangular facets from the tetrahedral mesh as in (d)iii above. This is shown in Figure 1(e)

f) **Tetrahedra Deletion and Hexahedron Formation.** All tetrahedra contained within the front and newly formed quadrilaterals are deleted and a new hexahedron formed.

g) **Local Smooth.** The node locations of the new hexahedron and its immediate neighbors are adjusted in an attempt to define a well-shaped hexahedron, as shown in Figure 1(f).

h) **Front Reclassification.** The fronts are reclassified and updated.

This process continues until all fronts have been processed, or those that remain have either failed the hexahedron formation process as their local topology is such that a hexahedron could not be formed.

7. **Smoothing.** After all fronts have been processed, a smoothing process is performed on all of the internal nodes.

8. **Pyramid Formation.** If any tetrahedra still exist in the mesh, pyramid elements are formed to interface tetrahedral elements with hexahedral elements.

3 IMPLEMENTATION

The following is a more detailed discussion of the H-Morph algorithm addressing some of the more significant implementation issues.

3.1 Quadrilateral Surface Meshing

Surfaces must first be meshed with quadrilaterals. In the current work, the Q-Morph[21] method is used, although any quadrilateral mesh generation technique may be used. To maximize the number and quality of hexahedra in the volume, it is advantageous to have regular rows of quadrilaterals, where the number of irregular nodes are minimized. The Q-Morph algorithm is able to provide these characteristics. Although the H-Morph algorithm can handle some change in element size, abrupt transitions in size in the quadrilateral mesh will generally not work well when hexahedra are placed.

3.2 Quadrilateral Subdivision

Each quadrilateral in the surface must be divided into two triangles. In doing so, it is useful to keep track of the initial quadrilaterals, as they will be defined as the initial front and provided as input to the advancing front method. It is also advantageous in splitting quadrilaterals to attempt to maximize the minimum angle of the resulting triangles. This results in a higher quality tetrahedral mesh.

3.3 Tetrahedral Mesh Generation

A boundary constrained tetrahedral mesh is generated from the triangular facets. The algorithm defined by George *et al.*[23,24] is used in the present implementation. With this algorithm the boundary facets are used as input and a Delaunay method is used to insert internal points. Internal nodal density is defined from the boundary nodal density. This method is generally very fast resulting in element quality sufficient for most applications. Any other boundary conforming method for generating a tetrahedral mesh could be used with the present developments, such as an alternate Delaunay method[25] or an advancing front scheme[26].

3.4 Generation of Initial Front

Once the interior of the volume has been meshed with tetrahedra, it is a simple matter to define a set of initial fronts. A front is a quadrilateral face composed of two triangles, which are in turn associated with two tetrahedra on the interior of the mesh. They are a convenient means of keeping track of the current progress of the advancing front algorithm, as they always define the boundary between the completed hexahedra and internal tetrahedra yet to be converted to hexahedra.

3.5 State Classification

To facilitate grouping and order of the progressing hexahedral mesh, the concept of the *state* of a front is introduced. The state machine approach, first used in the context of hexahedral mesh generation by Hipp and Lober[27], define how far along each front is to creating a complete hexahedron. It is based on the number of adjacent edges where the angle, α, between fronts is less than $3\pi/4$, where α is given by

$$\alpha = \begin{cases} \cos^{-1}\left(-\mathbf{N}\cdot\mathbf{N}_A\right) & \text{for } \mathbf{E}\cdot\mathbf{N}\times\mathbf{N}_A < 0 \\ 2\pi - \cos^{-1}\left(-\mathbf{N}\cdot\mathbf{N}_A\right) & \text{otherwise} \end{cases} \quad (1)$$

where $\mathbf{E}$ is the common edge vector between the two fronts, oriented counterclockwise with respect to the current front, $\mathbf{N}$ is the inward pointing normal of the current front and $\mathbf{N}_A$ is the inward pointing normal of the adjacent front as described in Figure 2. The normal vector $\mathbf{N}$ for any quadrilateral front is defined as the average normal of its two component triangles.

Figure 3 shows an example of the five possible front states. In practice it is useful to represent the current state as a bit field, where the first four bits of an integer are used. Each bit represents whether the corresponding edge of the quadrilateral has an angle, α, less than $3\pi/4$ to its adjacent front. The state bit field can be quickly converted to one of the conditions illustrated in Figure 3. Which bits have been set also defines how the front will be processed and in which order.

3.6 Front Priority

Fronts are selected one at a time, based on their defined priority, at which a single hexahedron is built off of each. A *proto-hex* structure is defined which keeps track of the current state of the hexahedron under construction. As such it contains references to all nodes, edges and faces that will be used in the final hexahedron. As each new quadrilateral face is constructed, the proto-hex is updated accordingly.

Priority for selecting the next front is based on the following criteria:

1. Level: The front level defines the number of layers of hexahedra defined from the boundary. Fronts in lower levels are processed before fronts in higher states. This ensures that the formation of hexahedra will advance evenly from the boundary towards the interior.
2. State: Fronts in higher states are selected first, as they have fewer remaining quadrilateral side faces to generate. When any front is processed, it will generally result in the promotion of the state of it neighboring fronts. The neighboring fronts are in turn placed at the front of their new respective state list to be processed next. This ensures that new hexahedra are in general formed next to the one generated last, resulting in an orderly *marching* pattern.
3. Size: Once a front has been selected for processing based on its current state and level, it is likely that any non-zero state front will have an adjacent front that will provide a better base for the formation of the hexahedron. A check is first made on the fronts on the proto-hex for the quadrilateral with the largest area. Selection of the largest area front to be used as the base is especially advantageous where the intended hexahedron is non-isotropic or stretched. This usually minimizes the distance new edges must be projected into the space occupied by tetrahedra, resulting in a higher success rate.

3.7 Tetrahedral Transformations

Throughout the H-Morph process, various transformations must be made to the tetrahedra. These transformations include rearrangement of the local topology in order to

arrive at a condition that will better admit a topology to form hexahedra. Some of these transformations have been described in other literature[24,28]. A summary of the transformations used in the H-Morph algorithms are shown in Table 1, where the initial and final tetrahedra are described in terms of their vertices. Application of these transformations will be discussed in subsequent sections of this work.

The *swap23*, *swap22* and *face-split* operations begin with two tetrahedra sharing a single face. Table 1 illustrates how the transformed topology results in three, two and six tetrahedra respectively.

Table 1 also illustrates the *edge-split*, *edge-collapse* and *edge-suppress* operations where the initial configuration is the set of tetrahedra sharing a single internal edge in the mesh, sometimes referred to as a *shell*. For the *edge-split* operation, where the shell contains N tetrahedra, the resulting configuration will have $2N$ tetrahedra. The *edge-collapse* operation on the other hand removes exactly N tetrahedra from the mesh as it collapses the edge ab to a common node c. The most complex of the operations, *edge-suppress*, attempts to eliminate the edge ab by retriangulating the space occupied by the N tetrahedra of the shell. This can be accomplished by triangulating the non-planar polygon $P = \{n_1, n_2, \ldots n_N\}$ and forming tetrahedra $n_k n_j n_i a$ and $n_i n_j n_k b$ for each triangle in polygon P, where n_i, n_j, n_k are vertices in P. For most of the operations described in Table 1, a check must first be made to ensure that the resulting configuration will result in tetrahedra with positive volume.

An additional common operation, which does not involve a change in topology, is the *node-relocation* operation. This involves simply repositioning a single internal node in the mesh. This operation also requires local checks to ensure all neighboring tetrahedra do not become inverted. Where tetrahedra would otherwise become inverted, the node can be incrementally relocated to a position on the vector between the old and new location until all tetrahedra maintain a positive volume.

3.8 Formation of Quadrilateral Side Faces

At each edge on the base front that does not currently have an adjacent front with an angle less than $3\pi/4$, a quadrilateral is formed from the internal tetrahedral mesh. As

previously mentioned, this involves three main algorithmic steps: (1) edge determination, (2) closure resolution and (3) face recovery.

3.8.1 Edge Determination

The objective of the edge determination process is to generate a projecting edge from the base front that will form the side of a new quadrilateral face of the proto-hex. The process of edge formation involves either the selection or formation of a new edge in the internal tetrahedral mesh. To define Edge E_k to be used as the new edge of the new quadrilateral face projected from the node N_k on the base front, the ideal projection vector, $\mathbf{V}_k$ and length, h is first determined. $\mathbf{V}_k$ is defined as the average of the normal vectors to the base front and its two adjacent fronts with node N_k in common. Front normals where the angle, α between the base front and its neighbor are greater than $5\pi/4$ are not included in the average. Where the base front is in a non-zero state, length h is defined as the average length of edges on the proto-hex that project in the same topological direction as E_k. For example in Figure 4, h is defined as the average length of edges A, B and C. Where the state of the base front is zero, $h = \sqrt{A_{bf}}$, where A_{bf} is the area of the base front.

Figure 5(a) shows the simplest case for selection of an existing edge in the triangulation for use as E_k. The angle, $\theta_i = \cos^{-1}\left(\mathbf{V}_k \cdot \vec{\mathrm{E}}_i\right)$, for each edge E_i having node N_k as an end point, is determined and the edge E_i with the smallest θ_i is selected as E_k. For this first case, E_k is only selected provided $\theta < \varepsilon$, where ε is a constant small angle heuristically determined to be $\pi/6$.

Frequently, the edge E_k, although within the ε tolerance, has a length greater than $1.5h$. When this occurs, one of two options may be used. The location of node, N_m on E_k may be adjusted so that the length of E_k is close to or exactly h using the node-relocation operation described previously. Where relocation of N_m is not possible, a new node may be inserted on E_k at length h. To effect this node insertion, the *edge-split* operation is performed as described in Table 1. It is advantageous to minimize the number of new nodes introduced into the triangulation. For this reason, an attempt is first made to

shorten the length of E_k by repositioning N_m. If N_m lies on an opposite front, and the length of E_k is greater than $1.5h$, then a new node is inserted.

In cases where there is no edge E_i with $\theta_i<\varepsilon$, one of several alternative options may be selected. The tetrahedron, T_k, or face, F_k, through which vector $\mathbf{V}_k$ passes, is first determined. Examining all tetrahedron adjacent to N_k can do this. The dot product, d_i of the three inward pointing normals of triangle faces, F_i sharing node N_k with vector $\mathbf{V}_k$ are computed. If all $d_i < 0$ for any single tetrahedron, T_i, then T_i completely contains $\mathbf{V}_k$ and $T_k=T_i$. If any $d_i=0$, then $\mathbf{V}_k$ lies exactly on face, F_i, and $F_k=F_i$. For the former case where $\mathbf{V}_k$ is completely contained within T_k, Figure 5(b) and Figure 5(c) illustrate two possible resolutions. Figure 5(b) shows the case where tetrahedron defined by nodes $ABCN_k$ and its immediate neighbor $ABCN_m$, are transformed into three tetrahedra. The *swap23* operation, as described in Table 1, is a common transformation used frequently in tetrahedral cleanup operations and in boundary recovery for constrained Delaunay mesh generation. In this application, the objective of the *swap23* operation is to form an edge N_kN_m that satisfies the criterion $\theta<\varepsilon$. If before effecting the transformation, it is determined that this criterion will be met, and that the resulting tetrahedron will be non-inverted, then the transformation is performed. If on the other hand, the $\theta<\varepsilon$ criteria is not met, or the transformation would result in inverting tetrahedra, then a new node N_n is inserted on face ABC as shown in Figure 5(c), where vector $\mathbf{V}_k$ intersects ABC. Insertion of N_n is the *face-split* operation, also illustrated in Table 1, resulting in six tetrahedra being created from the original two.

For the case when $\mathbf{V}_k$ is coincident with face, F_k, it is necessary to insert an additional node. Figure 5(d) shows $\mathbf{V}_k$ in the plane of triangle $F_k=ABN_k$ and the new node N_n inserted on edge AB, using the *edge-split* operation. Each tetrahedra adjacent edge AB is split, and E_k is defined as the new edge N_kN_n

3.8.2 Closure Resolution

Also considered in the edge formation process is the resolution of closure situations. As the front advances, the selected edge E_k will be either part of an existing front or N_m, the opposite node from N_k on E_k, will be on an existing front. Since by

selecting E_k that is associated with an existing front, a closure situation will result, it is necessary to determine if the resulting topology will form a reasonable resolution of hexahedra. While a simple even/odd rule for closure is applicable and useful in the context of the Q-Morph[21] algorithm, it was determined that more heuristic means were necessary for the H-Morph algorithm in order to maintain reasonable element quality.

While examining the edges E_i at node N_k, E_i can be classified into one of three conditions: (1) E_i forms the edge of an existing front, (2) Only N_m lies on an existing front while E_i does not, or (3) Neither E_i or N_m lie on an existing front. In order to promote the closure of the local hexahedra, it is advantageous to give priority to those edges that will provide the best closure situation before those that will result in the generation of new fronts. To do this, edges satisfying the $\theta_i < \varepsilon$ condition are first classified according to the above criteria. Those in the first category are given first priority, followed by those in categories (2) and (3). Where edge E_k is in category (3), the procedures described in the previous section are employed. To further encourage selection of edges that will result in closure, when an edge is determined to be in categories (1) or (2), a modified ε value is used, based on the current level, l_c, of the base front, as

$$\varepsilon = \begin{cases} \varepsilon_b + (\varepsilon_i - \varepsilon_b)\dfrac{l_c}{4} & \text{for } l_c \le 4 \\ \varepsilon_b & \text{for } l_c > 4 \end{cases} \qquad (2)$$

where ε_b and ε_i are heuristically defined as $\pi/6$ and $\pi/3$, respectively. This relaxes the $\theta_i < \varepsilon$ condition only for edges away from the boundary, as there is generally more freedom to reposition nodes to improve element quality away from the boundary.

Edges in the first two categories are ranked according to a normalized metric, β, where β is defined as the minimum quality of any of the quadrilateral faces that would be formed as a result of selecting edge E_i. The metric is defined as

$$\beta = \min(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \cdot \min(\nu_1, \nu_2)$$
$$\alpha_i = 4\sqrt{3}\,\frac{A_i}{l_{i,1} + l_{i,2} + l_{i,3}} \qquad (3)$$

where α_i is the computed metric for each of the four permutations of triangles to be derived from the four quadrilateral vertices, A_i is the area of triangle, i and $l_{i,j}$ is the length of each of its edges. In state 0, no potential quads yet exist and β is set to 1.0, the maximum value. In order to take into account warping of the quadrilateral, the four normalized normals, $\mathbf{N}_i$, are computed for each of the triangles. In equation (3), the minimum dot product, ν_1, ν_2 of opposite triangle normals is used to reduce β, where $\nu_1 = \mathbf{N}_1 \cdot \mathbf{N}_3$ and $\nu_2 = \mathbf{N}_2 \cdot \mathbf{N}_4$. For example, for a planar quadrilateral, the value $\min(\nu_1,\nu_2)$ would be 1.0, while for triangles forming a right triangle, the multiplier would be 0.0.

Once the initial β value is determined based on the locations of the potential vertices of the new quadrilaterals to be defined, a modifier, χ, is subtracted from β based on the potential local topology that would be formed as a result of selecting E_i. For E_i in category (1) , χ=1.0 for faces where valid hexahedra cannot be formed, and χ=0.0 where no such ill effects can be determined.

When edge E_i is in category (2), a local traversal of the front can be effected to determine if E_i will form a valid closure by assigning each local front an integer, λ. Beginning from node N_k, the front faces in the same topological plane are assigned λ=0. As the traversal continues, those fronts adjacent to the fronts where λ=0 and that form an angle at the common edge smaller than $3\pi/4$ are assigned λ=1. This process continues for the fronts assigned λ=1, where unassigned adjacent fronts forming an angle smaller than $3\pi/4$ are assigned λ=2. The criterion for determination of a valid closure now becomes that of interrogating the front faces neighboring node N_m. For any face adjacent to N_m where $\lambda<2$ then χ=1.0. For all faces where $\lambda \geq 2$, χ=0.0.

For closure edges in categories (1) and (2), where more than one edge E_i is available for selection, E_i with the maximum β is selected, where category (1) edges always have precedence over category (2). Additionally, edge E_i is rejected outright if β falls below a minimum threshold, β_t. Similar to the value for ε, because nodes have more latitude to move without seriously affecting element quality towards the interior of the mesh, the value β_t is a function of the level, l_c, of the current base front as

$$\beta_i = \begin{cases} \beta_b + (\beta_i - \beta_b)\frac{l_c}{4} & \text{for } l_c \le 4 \\ \beta_i & \text{for } l_c > 4 \end{cases} \quad (4)$$

where β_b is the minimum allowable β at the boundary and β_i is the minimum allowable on the interior of the mesh. In the current work, β_b=0.25 and β_i=-0.5. For complete details of the above discussion, the reader is referred to Reference [30].

3.8.3 Face/Edge Recovery

One of the keys to the success of the H-Morph algorithm is the edge and face recovery process. In the literature[23,24,29], edge and face recovery has typically been presented in the context of boundary recovery for tetrahedral mesh generation. Within the context of the H-Morph algorithm, the edge and face recovery process is used for delimiting each individual edge and face of the hexahedra. Since the recovery process is applicable only for triangular facets, two facets per hexahedral face must be recovered.

The two-dimensional edge recovery process, described within the context of the Q-Morph[21] algorithm, can be effectively modified for the three-dimensional case. Where the two-dimensional edge recovery process involved the use of diagonal edge swaps, its three-dimensional counterpart involves the use of the *swap23* and *swap22* operations described in Table 1. Unlike the two-dimensional case, the additional process of recovering a face is required. This is by virtue of the fact that even after enforcing the edges of a triangle facet in the tetrahedral mesh, there is no guarantee that a face will occupy the plane inside the edges. It is conceivable that any number of edges and faces may penetrate the plane of the face under consideration. The face recovery process is a set of procedures for eliminating these penetrating faces and edges to recover the triangle facet between three existing edges. The *edge-suppress* operation of Table 1 is most frequently used to effect the recovery of a face.

In the current implementation, many of the same methods utilized by George and Bourachaki[23,24] are utilized for both edge and face recovery. The significant exception is where an internal vertex or *Steiner* point is needed to resolve a difficult topological situation. In the H-Morph algorithm, by perturbing the location of an internal node, an

otherwise unresolvable situation can be remedied without resorting to the insertion of a Steiner point. As a result, fewer internal nodes must be inserted. A detailed description of the edge and boundary recovery for use in the H-Morph algorithm is outlined in References [30] and [31].

3.9 Quadrilateral Top Face

With all four sides of the hexahedron formed, the H-Morph process now comes to finishing off the hex. In some cases, it is possible that the top has already been formed from another front advancing from the opposite direction. When this occurs, the algorithm simply drops to the hexahedron formation phase and continues. For the more common case where a top face is not yet apparent, the same boundary recovery techniques are used as described in the previous section. After successful recovery, the top front is formed and proto-hex updated with the final face.

3.10 Tetrahedra Deletion and Hexahedron Formation

In this phase of the algorithm, the tetrahedra contained within the completed proto-hex must first be eliminated. In order to effect this operation, a local search is made starting with one of the tetrahedra immediately adjacent the base front. The search continues recursively advancing from one adjacent tetrahedra to the next, placing references to the tetrahedra on a list as it proceeds. The process continues until the search runs into the quadrilateral faces of the proto-hex. With a complete list of tetrahedra contained within the proto-hex, it is now a simple task to delete each tetrahedron in the list, making sure to delete any unused faces, edges and nodes as it proceeds.

Using the information provided by the proto-hex, the new hexahedron element can be formed. It should be noted that at least two tetrahedra will be immediately adjacent any new front that was formed as a result of the current hex-formation process.

3.11 Local Smoothing

After the hexahedron has been formed, because subsequent hexahedra will use the current hexahedron as a base from which to build, it is important that the best possible element shape be provided. This may involve adjustments to the nodes on the new

hexahedron, as well as those on tetrahedra and hexahedra in the near vicinity. For the H-Morph implementation, a combination of isoparametric smoothing[32] combined with a three-dimensional extension of the smoothing technique used in the Q-Morph[21] and Paving[33] algorithms, proved to be most effective.

For any single node surrounded completely by tetrahedra, or by hexahedra, a simple Laplacian smooth seemed to be the most efficient and effective way to adjust nodes. In cases where the node is on the front, where there are a combination of both tetrahedra and hexahedra at the node, the Laplacian smoothing proved ineffective. At this point of the H-Morph algorithm, where the quality of the hexahedra at the boundary is critical, nodes are smoothed in order to improve hexahedral element quality, frequently at the expense of the quality of the nearby tetrahedra. While the objective for hexahedral elements was to define the best possible shape, the objective for tetrahedral elements at this stage, is simply to prevent them from becoming inverted. Therefore, after computing a new node location based on the hexahedral element quality, the *node-relocation* operation is used to maintain positive volumes for any neighboring tetrahedra.

Two cases for smoothing nodes at the front are addressed: (1) *corner* nodes and (2) *row* nodes. Interrogating each of the edges attached to the node, N_s to be smoothed, can identify the difference between a corner node and row node. If one or more edges at N_s is completely surrounded by hexahedral elements, then it is defined as a *row* node. Otherwise, it is considered a *corner* node.

3.11.1 Corner Nodes

The smoothed location, $\mathbf{P}_s$ of corner node, N_s is defined as

$$\mathbf{P}_s = \frac{\sum_{i=1}^{n} \mathbf{P}_i}{n} \qquad (5)$$

where n is the number of adjacent hexahedra to N_s and $\mathbf{P}_i$ is the contribution of adjacent hexahedra, i, to N_s to the location, $\mathbf{P}_s$. If the nodes and edges of a hexahedron are indexed in the manner shown in Figure 6 with respect to the node, N_s, then the contribution, $\mathbf{P}_i$ can be defined as

$$\mathbf{P}_i = \frac{\mathbf{V}_I + \mathbf{V}_J + \mathbf{V}_K}{3} \quad (6)$$

$$\mathbf{V}_I = \mathbf{P}_1 + \frac{c_2(\mathbf{P}_3 - \mathbf{P}_2) + c_4(\mathbf{P}_4 - \mathbf{P}_5) + c_6(\mathbf{P}_7 - \mathbf{P}_6)}{c_2 + c_4 + c_6} \quad (7)$$

where $\mathbf{P}_j$ {j=1,2,...7} is node *j*, on the hexahedron and $\mathbf{V}_I$ is the contribution to location $\mathbf{P}_i$ taken as the average of all edge vectors of the hexahedron in direction *I*. $\mathbf{V}_J$ and $\mathbf{V}_K$ are defined similarly.

In Equation (7), c_k=0 for any edge *k* that was defined as part of the current proto-hex and c_k=1, otherwise. In the case where the proto-hex was defined from a state 1 base front, the denominator in Equation (7) will be zero. When this occurs, the appropriate term is factored out of the final contribution to $\mathbf{P}_i$. For the rare case when the proto-hex was formed from a state 0 base front, c_k=1 $\forall$ *k*.

3.11.2 Row Nodes

For *row* nodes, the location $\mathbf{P}_s$, is based upon the cumulative contribution from all hexahedra that surround any of the edges attached to N_s. Let E_s be an edge attached to N_s where all adjacent elements are hexahedra. Each set of hexahedra sharing a common edge, E_s is treated simultaneously, and the contribution from each edge E_s at N_s is averaged to define the location $\mathbf{P}_s$. Equation (5) adequately describes the definition of $\mathbf{P}_s$ for row nodes, except that *n*, is now the total number of edges E_s adjacent N_s. $\mathbf{P}_i$ is now computed for each edge, E_s, as a perturbation Δ_i from its original location as

$$\mathbf{P}_i = \mathbf{P}_i + \Delta_i \quad (8)$$

Blacker and Stephenson[33] develop the vector Δ_i for *row* nodes for the two-dimensional *paving* algorithm. The following is a similar development, except within the context of three-dimensional hexahedral elements. The perturbation Δ_i has a contribution from three main components: (1) isoparametric smoothing, (2) adjustment for squarness or perpendicularity of the surrounding elements, and (3) adjustment for angular smoothness, each of which are represented by Δ_a, Δ_b, and Δ_c, respectively.

While the Laplacian smoothing technique averages the locations of surrounding nodes to N_s, the isoparametric smooth computes the smoothed location by summing those nodes which are directly attached by an edge, but subtracting those which are diagonal at a face. When defined in terms of the hexahedra at edge E_s, the location $(P_s)_{iso}$ can be defined as

$$(\mathbf{P}_s)_{iso} = \frac{\sum_{j=1}^{n}\left[\sum_{k=1}^{7} c_k \mathbf{P}_k\right]}{3n} \tag{9}$$

$$c_k = \begin{cases} 2, & k = 1,3,4 \\ -1, & k = 2,5,7 \\ 0, & k = 6 \end{cases} \tag{10}$$

where n is the number of hexahedra at edge E_s and $\mathbf{P}_k$ are the nodal locations on element, j, adjacent to E_s where the nodes are oriented as in Figure 7(a) with respect to node N_s. The constant c_k is a convenient method for adding or subtracting the required nodal location based on its orientation on the hex with respect to N_s. The isoparametric contribution, Δ_a, shown in Figure 7(a), can now be defined as

$$\Delta_a = (\mathbf{P}_s)_{iso} - \mathbf{P}_s \tag{11}$$

Blacker and Stephenson[33] next propose the adjustment for squareness or perpendicularity, Δ_b to be a modification of the isoparametric contribution as

$$\Delta_b = (\mathbf{P}_j - \mathbf{P}_s) + (\Delta_a + \mathbf{P}_s - \mathbf{P}_j)\frac{l_d}{l_a} \tag{12}$$

$$l_d = \left\|(\mathbf{P}_s)_{iso} - \mathbf{P}_j\right\|, \qquad l_a = \sum_{k=1}^{n_f} \left\|\mathrm{E}_k\right\| \Big/ n_f \tag{13}$$

where $\mathbf{P}_j$ is the location of the opposite node on E_s from N_s, E_k are the opposite edges on the faces adjacent to E_s and n_f is the number of faces adjacent to E_s. Figure 7(a) shows E_k, $\{k = 1,2,\ldots n_f\}$ where $n_f = 5$.

The final contribution, Δ_c, to the smoothed location of N_s provides a component intended to smooth the angles between neighboring edges. Let $\mathbf{P}_k$ $\{k = 1,2,\ldots n_f\}$, be node

locations on the faces adjacent to E_s where the node at $\mathbf{P}_k$ is opposite node to $\mathbf{P}_j$ as shown in Figure 7(b). The vectors $\mathbf{V}_{b1}$ and $\mathbf{V}_{b2}$ can then be defined as

$$\mathbf{V}_{b1} = \frac{\sum_{k=1}^{n_f} \mathbf{P}_k - \mathbf{P}_j}{\left\| \sum_{k=1}^{n_f} \mathbf{P}_k - \mathbf{P}_j \right\|}, \qquad \mathbf{V}_{b2} = \frac{\dfrac{\mathbf{P}_s - \mathbf{P}_j}{\left\| \mathbf{P}_s - \mathbf{P}_j \right\|} + \mathbf{V}_{b1}}{2} \tag{14}$$

Also, define Φ_k $\{k = 1,2,\ldots n_f\}$ as the oriented plane defined by the points $\mathbf{P}_k$, $\mathbf{P}_{fk}$, $\mathbf{P}_{k+1}$ as illustrated in Figure 7(b), then the point $\mathbf{P}_{Qk}$ can be defined as the intersection of edge E_s with Φ_k, and points $\mathbf{P}_Q$, and $\mathbf{P}_{b2}$ are defined as

$$\mathbf{P}_Q = \frac{\sum_{k=1}^{n_f} \Phi_k \cap \overline{E_s}}{n_f}, \qquad \mathbf{P}_{b2} = \mathbf{V}_{b2} l_c \tag{15}$$

$$l_c = \begin{cases} \dfrac{l_Q + l_d}{2} & \text{for } l_d > l_Q \\ l_d & \text{otherwise} \end{cases} \tag{16}$$

$$l_Q = \left\| \mathbf{P}_Q - \mathbf{P}_j \right\| \tag{17}$$

where l_d is described in Equation (13). Finally the contribution Δ_c can be defined as

$$\Delta_c = \mathbf{P}_{b2} - \mathbf{P}_s - \mathbf{P}_j \tag{18}$$

and the perturbation, Δ_i, applied in Equation (8) is obtained as the average of Δ_b and Δ_c

$$\Delta_i = \frac{\Delta_b + \Delta_c}{2} \tag{19}$$

3.12 Seams and Wedges

During the H-Morph process, when the angles between adjacent internal fronts fall below a threshold angle, ε, they are placed on a separate list to be *seamed*. Similar to its two-dimensional counterpart described in Owen et al.[21], the seam operation merges

two fronts where an otherwise poor quality hexahedral element would result. Blacker and Meyers[15] describe the use of seams in the context of the plastering algorithm, also elaborating on *wedges*. In this work, the *driving* of wedges is a mechanism for resolving troublesome topology, by moving the difficulty towards a frontal boundary, where it can be more easily explicated. The direction to drive a wedge can be somewhat ambiguous without a global understanding of the topology. Frequently the resolution of a wedge can only be accomplished by driving it to the surface mesh, resulting in the modification of the boundary. Where this is not an acceptable option, it can be left in the interior of the mesh, unresolved, eventually defining what has become known as a *knife* shaped element[34]. Although conditions for the introduction of wedges are certainly possible during the H-Morph algorithm, it was deemed imprudent to address the implications of driving wedges through the mixed hexahedra/tetrahedra mesh. These topological situations are instead left unresolved, leaving the local region with tetrahedra, later to be transformed into pyramids.

While not addressing wedges, the seam operation *can* be defined conveniently within the framework of the H-morph algorithm. Figure 8(a) and Figure 8(b) show the two cases that the seam operation is intended to resolve. Edge AB in these figures have adjacent fronts that form an angle less than ε. While in the plastering algorithm, it is a simple matter to merge the two adjacent fronts, however, within the context of H-Morph there will be any number of tetrahedra immediately adjacent the fronts that must be dealt with.

3.12.1 Case 1 Seam

To resolve the basic case illustrated in Figure 8(a), the edge *CD* and triangle *BCD* must exist in the tetrahedral mesh. To enforce this condition, the boundary recovery functions described in section 3.8.3 are once again utilized to recover the facet. With edge *CD* now in place, the *shell*(CD) can be defined and the *edge-collapse* operation described in Table 1, utilized to merge the two nodes C and D. The result is the configuration shown in Figure 8(b). The configuration is then treated as a *case 2* seam and resolved appropriately. The edge collapse operation locates node C such that all of the attached tetrahedra are non-inverted. The optimization-based smoother can also be

invoked in the event a valid location cannot be found. Since there is no guarantee that the edge-collapse operation will be successful, in the present work, the seaming operation has the flexibility to back up and fail gracefully, leaving the topology to be either resolved later with a hexahedra or with a combination of tetrahedra and pyramids.

3.12.2 Case 2 Seam

A Case 2 seam operation is performed when two edges of an adjacent pair of fronts are in common, as shown in Figure 8(b). In this case, two facets must be recovered from the tetrahedra. These are illustrated in Figure 8(b) as faces *EFC* and *EFA*. Once recovered, it is necessary to delete all tetrahedra within the region to be collapsed, described by the nodes *E-F-A-B-C*. An algorithm similar to that illustrated in section 3.10, where all tetrahedra within the proto-hex are deleted before formation of the hexahedron, can be utilized for this purpose. Finally, the *edge-collapse* operation of Table 1 can be utilized on edge *EF* to merge the two fronts together resulting in the configuration of Figure 8(c). Local smoothing and state reclassification of the neighboring fronts is also done to complete the process.

3.13 Front Reclassification

When a new hexahedron has been formed, any new fronts that were created in the process of forming it are classified and placed on their appropriate front list. It is important to note that these new fronts are placed at the top of their respective state lists so that they will be the next in line to be processed. This enables a more systematic order to the processing of the fronts.

During the local smoothing process, done after formation of hexahedra and seaming operations, it is likely that the angle, α, between any number of the nearby fronts will have changed. When this results in α becoming greater than $3\pi/4$ or dropping below $3\pi/4$, it is necessary to move the front to a new, more appropriate state list. To effect this process in an efficient manner, a list is made of all fronts adjacent to any node moved during the local smoothing process. Angle, α, for each front on this list to its neighboring fronts are recomputed and their state adjusted appropriately.

3.14 Global Smoothing

Before completing the mesh, all nodes on the interior of the mesh must be smoothed. This involves smoothing nodes, which are simultaneously adjacent to both tetrahedra and hexahedra. This can be done using a constrained Laplacian smoother, coupled with an optimization based smoothing technique, similar to that described by Freitag and Ollivier-Gooch[35] or Canann *et al.*[36]. In either the constrained Laplacian smoothing or the optimization based smoothing, a consistent metric is required to maintain the quality of the elements. While there are currently several methods for determining metrics for tetrahedral elements in the context of an all-tetrahedral mesh, a consistent metric has yet to be defined for a mixed element mesh that will be applicable for any element type.

Although a simple metric was implemented as part of this research for use with a constrained Laplacian/optimization-based smoother, in practice, the metric failed to adequately and consistently represent the element quality between tetrahedra and hexahedra. As a result, in most cases, while the quality of the tetrahedra increased, the overall quality of hexahedra immediately adjacent tetrahedra remaining on the interior of the mesh actually deteriorated. Therefore, the task of defining a consistent element metric applicable for a mixed element mesh has been left as an open problem.

3.15 Pyramid Formation

At this point in the H-Morph algorithm, the model will have either successfully processed all of the fronts, or those that prevail will have failed to produce a valid hexahedra resulting in tetrahedra remaining immediately adjacent the front. Each remaining front is now processed in order to form pyramids interfacing the hexahedra with the tetrahedra. The pyramid formation process consists of transforming and combining any number adjacent tetrahedra at the quadrilateral face of a hexahedron. Owen *et al.*[37] provide a detailed description of the process.

4 EXAMPLES

A number of examples were solved using the H-Morph procedure to demonstrate its validity in generating hexahedral-dominant meshes. A simple blocky-type configuration as shown in Figure 9 is first considered. This type of model is most often handled using a mapped meshing technique[1] after manually decomposing the geometry into *mappable* regions. An automatic geometry decomposition technique[8,9] or the sub-mapping[4] method can also be used with this class of geometry. The H-Morph algorithm is able to mesh this geometry without the need for decomposition. Figure 9(a) shows the initial boundary constrained tetrahedral mesh where a total of 4197 tetrahedra have been generated. Figure 9(b) and Figure 9(c) show intermediate stages of the H-Morph algorithm, where the tetrahedra are systematically converted into hexahedra. Finally, Figure 9(d) shows the completed mesh with a total of 756 hexahedral elements. The H-Morph algorithm is able to resolve this model completely into regular hexahedral elements.

The next example chosen to demonstrate the capability of the H-Morph algorithm to generate an all-hexahedral mesh is shown in Figure 10. This class of problems is generally handled using a sweeping method[5,6,7], where the surface meshes on *source* and *target* areas on opposite sides of the volume are topologically identical. The H-Morph algorithm is able to mesh this model without the need for recognizing this geometry as a special case, *sweepable* model. To test its validity relative to sweeping algorithms, the same mesh was provided on opposite sides of the model in this example. Figure 10(a) shows the initial tetrahedral mesh of 430 elements and the final regular hexahedral mesh with 80 elements is shown in Figure 10(b). The algorithm clearly is able to successively resolve this class of sweepable geometries.

The *cutblock* model, shown in Figure 11 is an example of a simple geometry that has an imposed surface mesh, which does not match from one side of the volume to the other. As a result, mapping or sweeping methods cannot be applied. Figure 11(a) shows the completed finite element model with a total of 998 elements including hexahedra, tetrahedra and pyramids. H-Morph is able to convert over 97 percent of the volume to hexahedra. Figure 11(b) shows the tetrahedra and pyramids on the interior of the model

after removing all hexahedral elements. Table 2 provides more detailed information about the number and percent hexahedra by volume in the model.

The remaining examples in Figure 12 and Figure 13 also represent complex surface meshes which cannot be resolved by mapping or sweeping methods. The geometry used in Figure 13 is a standard model previously used by Meyers *et al.*[22] to illustrate the Hex-Tet algorithm. The statistics for the present algorithm are presented in Table 2. The data indicates that H-Morph is able to fill these volumes with approximately 92 percent hexahedral elements.

5 PERFORMANCE

All elements solved in the previous examples have positive Jacobian Ratios. This indicates that H-Morph is, at a minimum, able to generate elements that are non-inverted. No mathematical proof, however, exists that guarantees positive Jacobian Ratios. The current research has focussed primarily on obtaining good quality hexahedral elements within the volume. In most cases, the hexahedral elements are of sufficient quality for finite element analysis as defined by standard shape checking procedures[38]. Because of the open problem of global smoothing of a mixed element mesh, the quality of the tetrahedra and pyramids is currently less than desirable.

Although Table 2 indicates a relatively small *volume* of tetrahedra remaining in the example problems, in some cases an unusually high *number* of tetrahedra may remain. The current implementation has shown that as the hexahedra are formed and smoothed, the interior nodes attached to tetrahedra, tend to *bunch-up* near the advancing front. This is caused by the fact that, given a constant edge length, it will take fewer layers of regular hexahedra to fill space than layers of regular tetrahedra. Hence more nodes are required for the tetrahedral mesh than for the transformed hexahedral mesh. While the H-Morph process tends to eliminate nodes as needed, there are cases where the nodes are pushed in front of the advancing front during the local smoothing process, resulting in higher numbers of tetrahedra than one would expect. The insertion of additional nodes as a result of the *face-split* and *edge-split* operations (described in Table 1) may also contribute to this phenomenon. Another important open issue to be

addressed in the context of the H-Morph algorithm is the reduction of the number of remaining tetrahedra in the model after completion of the hexahedral layers.

In its present implementation, H-Morph generates approximately 20 hexahedral elements per second on a standard NT workstation. This is slow by most standards, but tends to be about par when compared to the published performance of the Hex-Tet[22] algorithm. Optimization of the H-Morph algorithms must be addressed as they gain maturity.

6 CONCLUSION

A new method for generating a hexahedral-dominant mesh that will conform to a prescribed quadrilateral surface mesh is introduced in this work. The proposed method is general purpose, and is able to mesh without the need to decompose or recognize special classes of geometry. The indirect approach proposed for the generation of hexahedral elements begins with an initial tetrahedral mesh and systematically transforms the tetrahedra into a topology appropriate for the formation of well-shaped hexahedra. The process, built on the ideas initially developed in the Q-Morph algorithm[21] for two-dimensional quad meshes, utilizes an advancing front approach. Beginning at the boundary surface mesh, quadrilateral fronts are classified and processed, replacing tetrahedra with hexahedra as the algorithm proceeds. New procedures have been developed for tetrahedral transformations, nodal smoothing, edge determination, closure resolution and edge and face recovery. As an *indirect* method, the proposed H-Morph algorithm avoids the problematic intersection calculations inherent in other similar *direct*[14,15,18,22] methods. The proposed method is also able to maintain a valid mixed hexahedral-tetrahedral mesh throughout the entire procedure. This eliminates the need for meshing complex internal voids that would otherwise remain after meshing using a direct method. The H-Morph algorithm has currently been implemented and tested on a limited number of models sufficient to demonstrate its validity in generating hex-dominant meshes. Further development of the proposed method is on-going and performance and robustness will inevitably improve as the algorithms gain maturity.

7 ACKNOWLEDGEMENTS

We wish to acknowledge Ansys Inc. for providing the funding and resources for the development of this work.

List of Figures

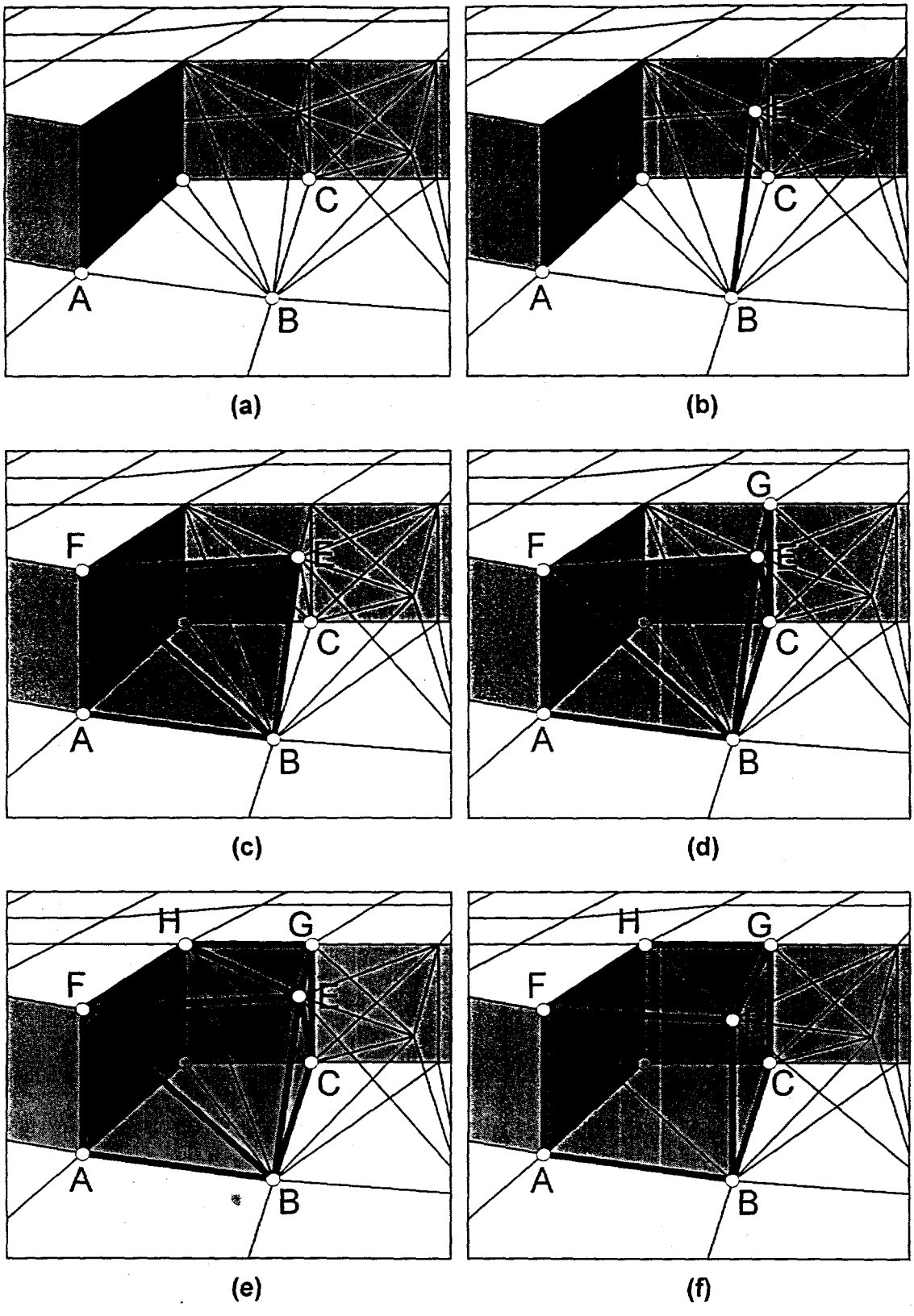


Figure 1. Construction of a hexahedron with the H-Morph algorithm.

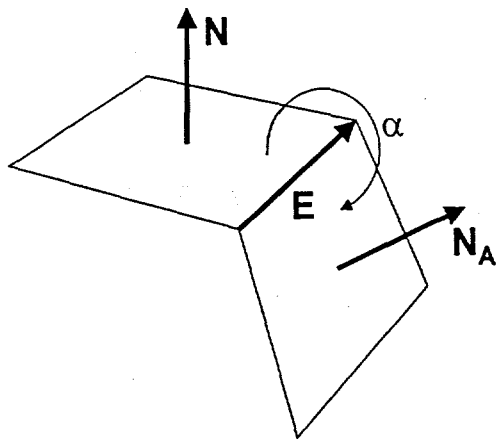


Figure 2. Definition of angle, α between fronts

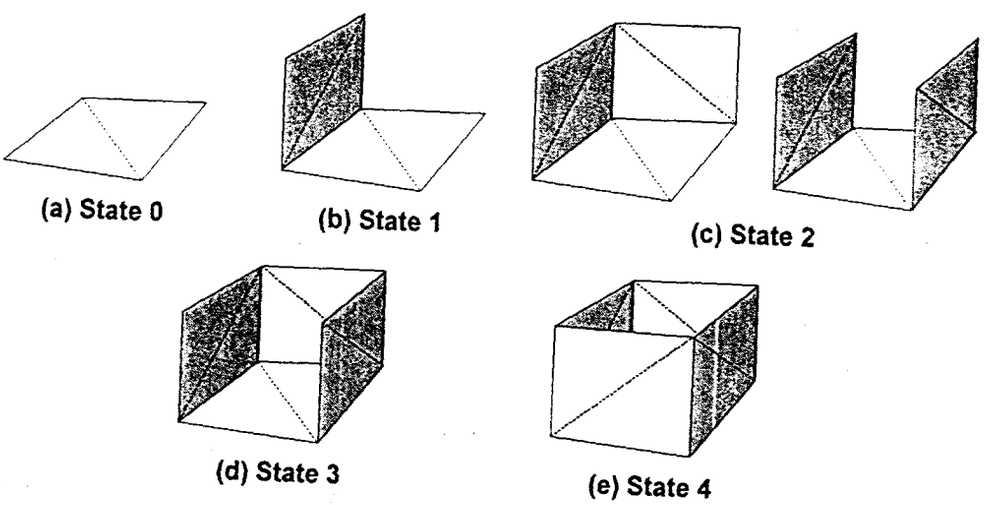


Figure 3. States of a front

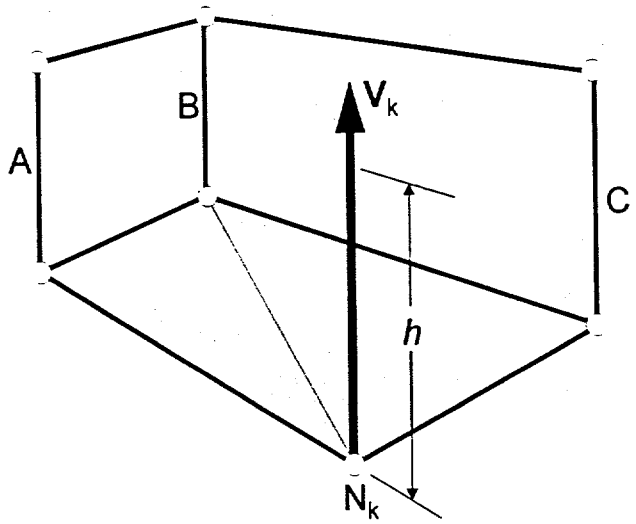


Figure 4. Definition of *h* and V_k during edge formation

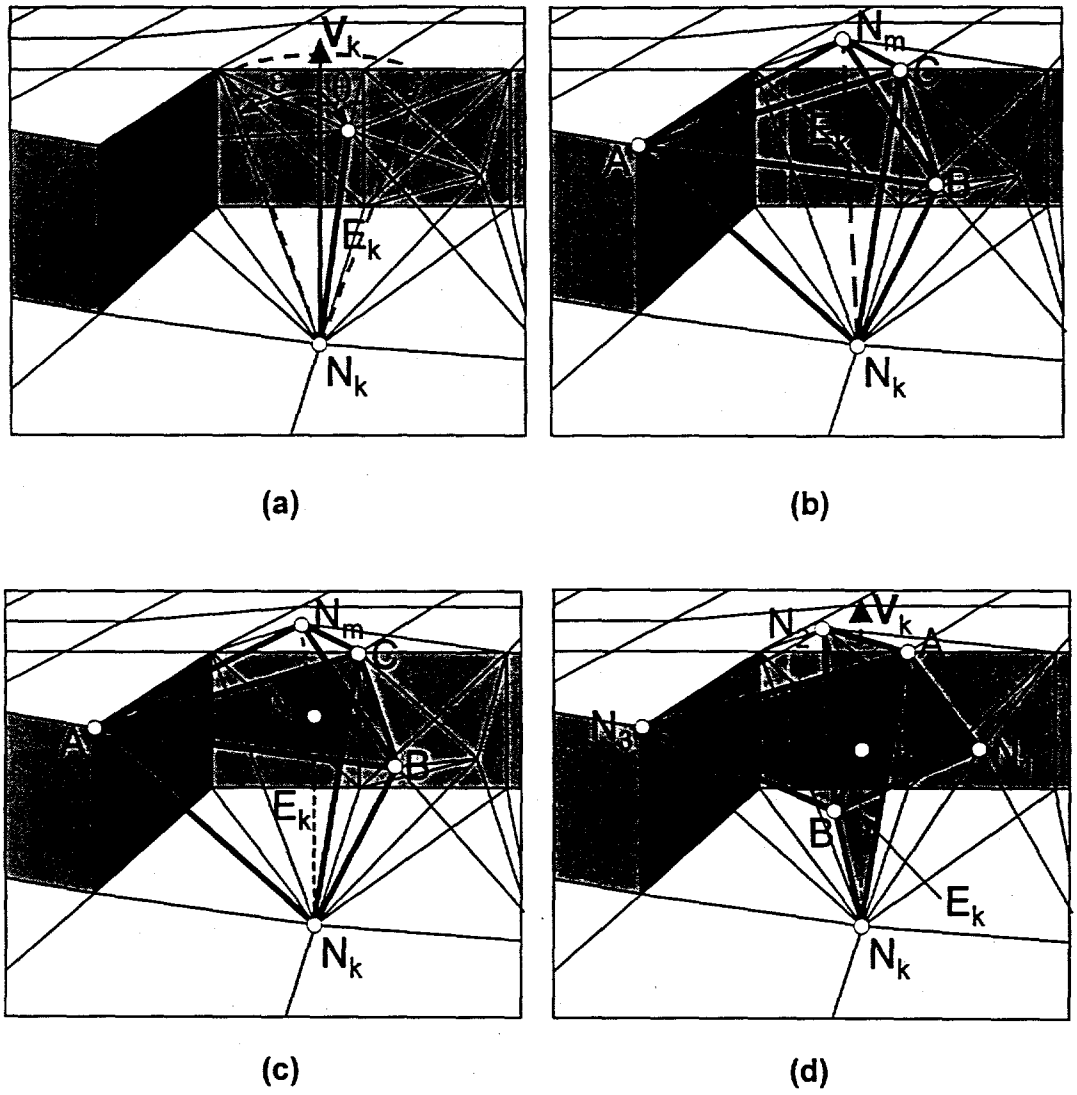


Figure 5. Definition of Edge E_k. (a) Edge defined from existing edge when $\theta < \varepsilon$ for any edge at N_k. (b) Edge defined by swapping (*Swap23*). (c) Edge defined by splitting a face (*face-split*). (d) Edge defined by splitting an edge (*edge-split*).

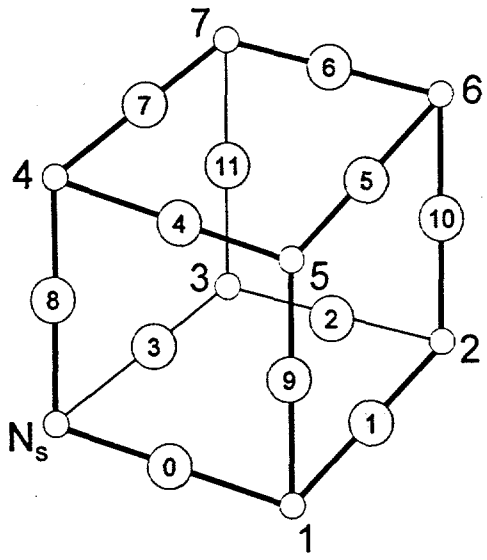
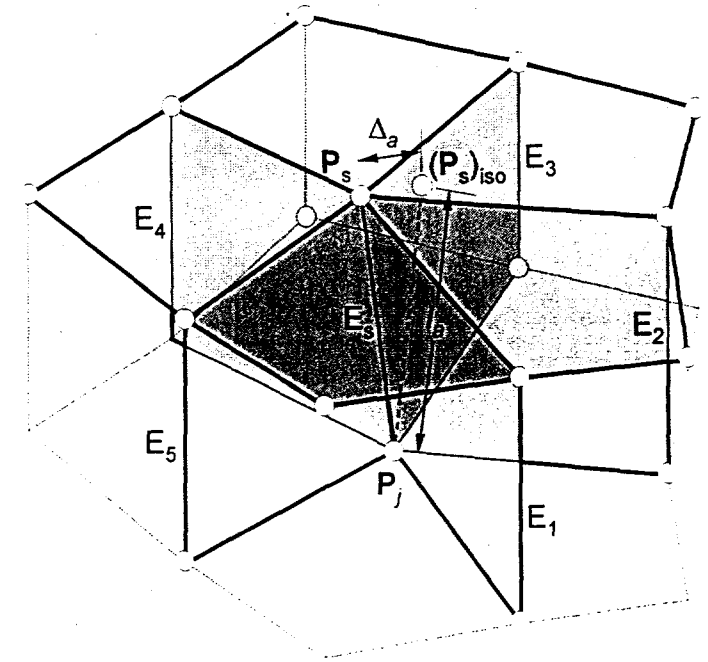
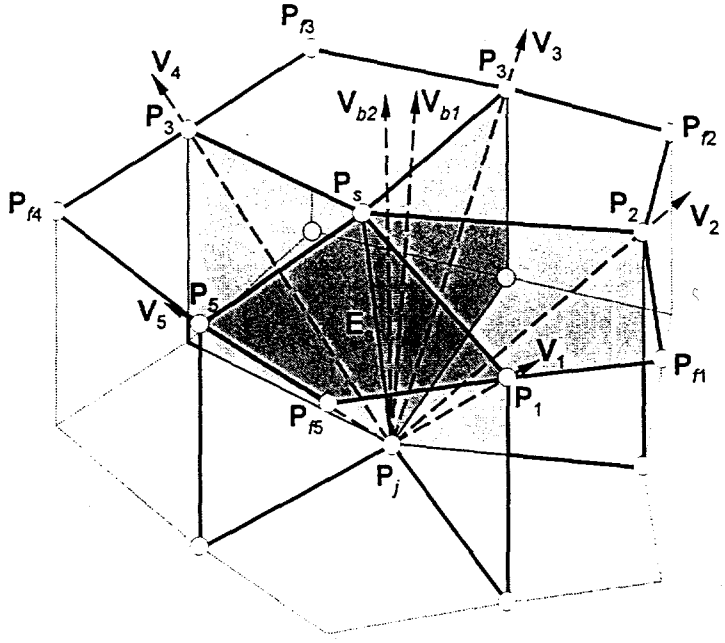


Figure 6. Hexahedron node and edge numbering



(a)



(b)

Figure 7 (a) Configuration of hex elements around edge E_s. (b) Definition of vectors for angular smoothness criteria

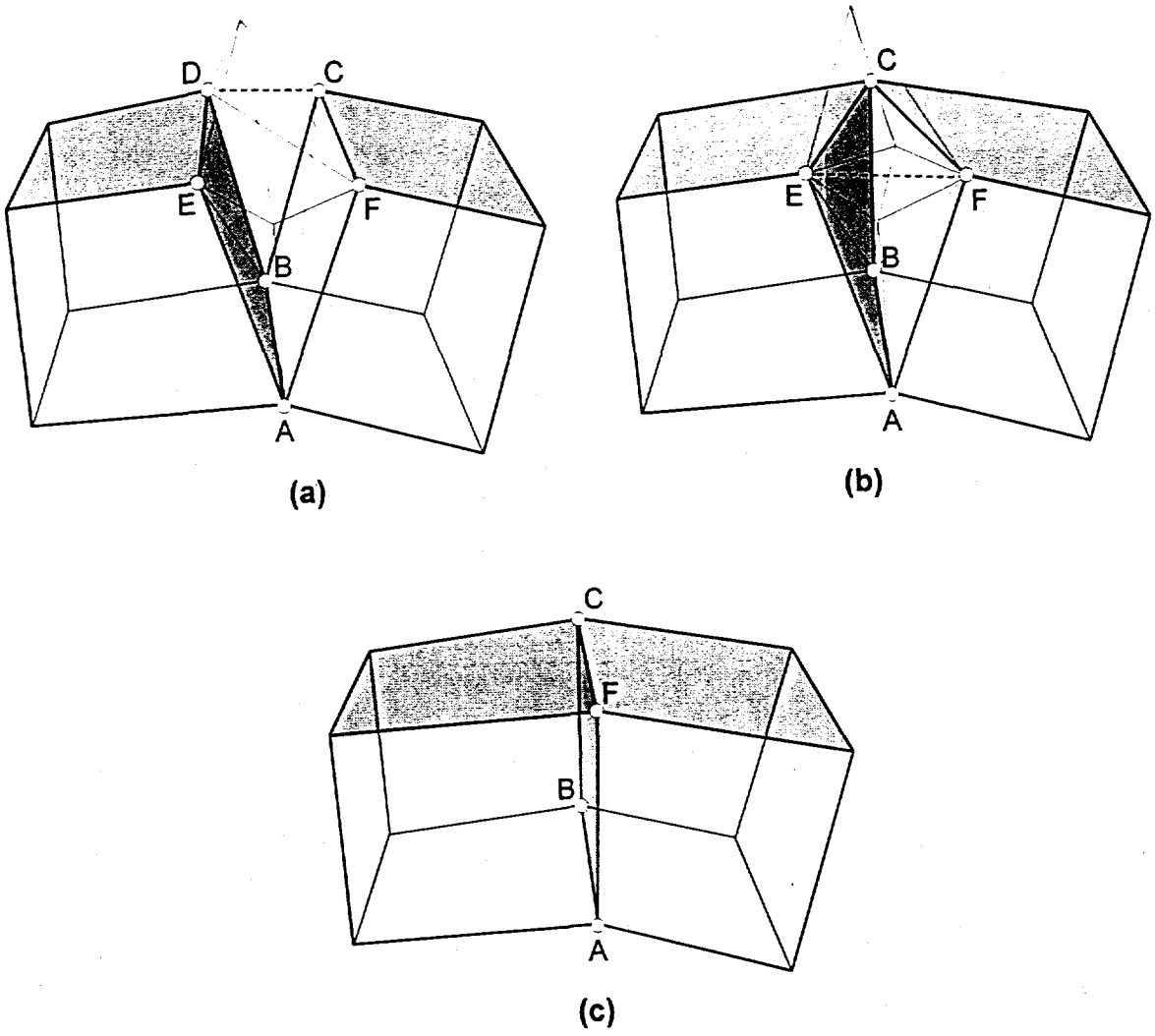


Figure 8. Seam Operation (a) case 1, (b) case 2, (c) resolved seam

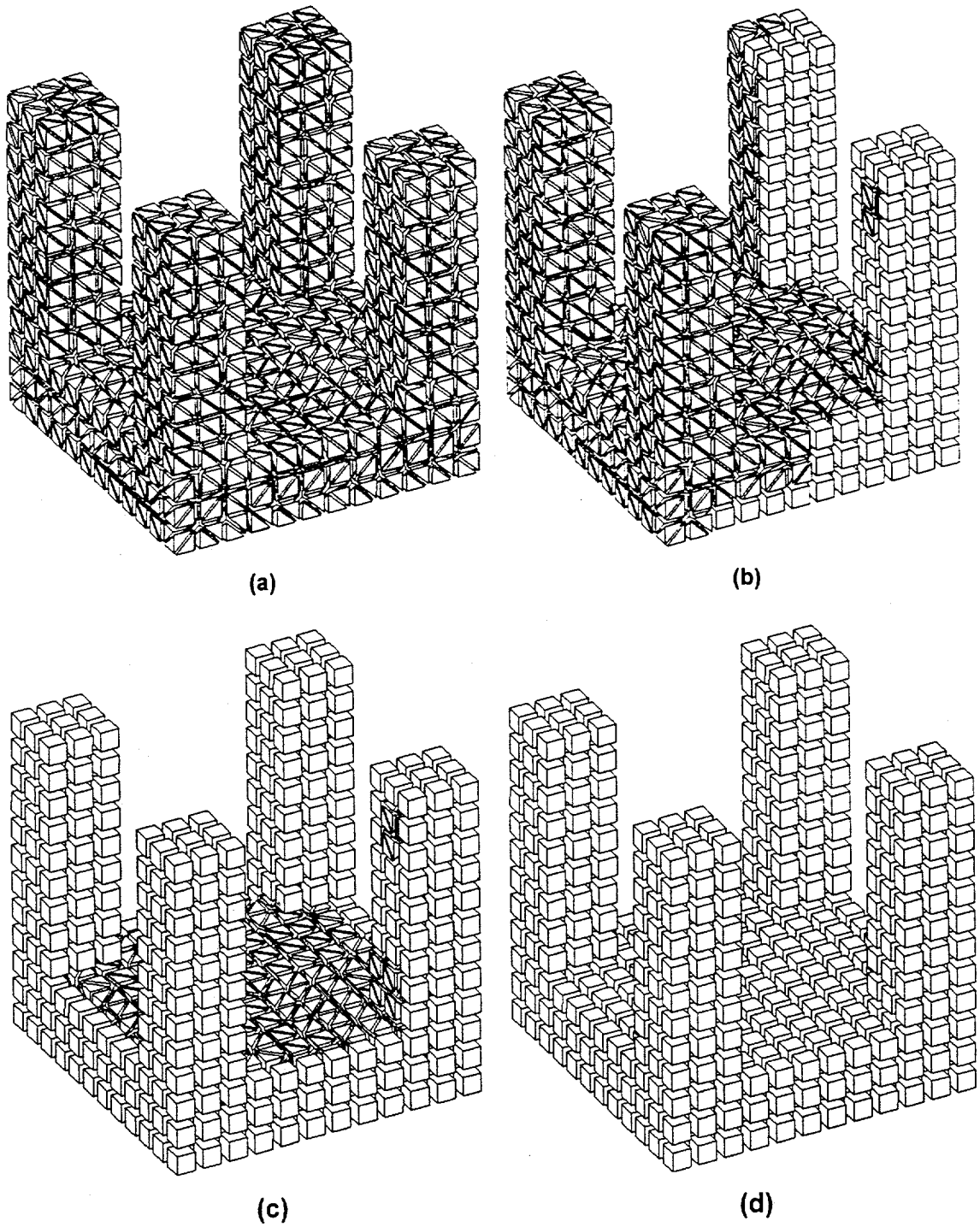


Figure 9. Progression of H-Morph algorithm on simple blocky model

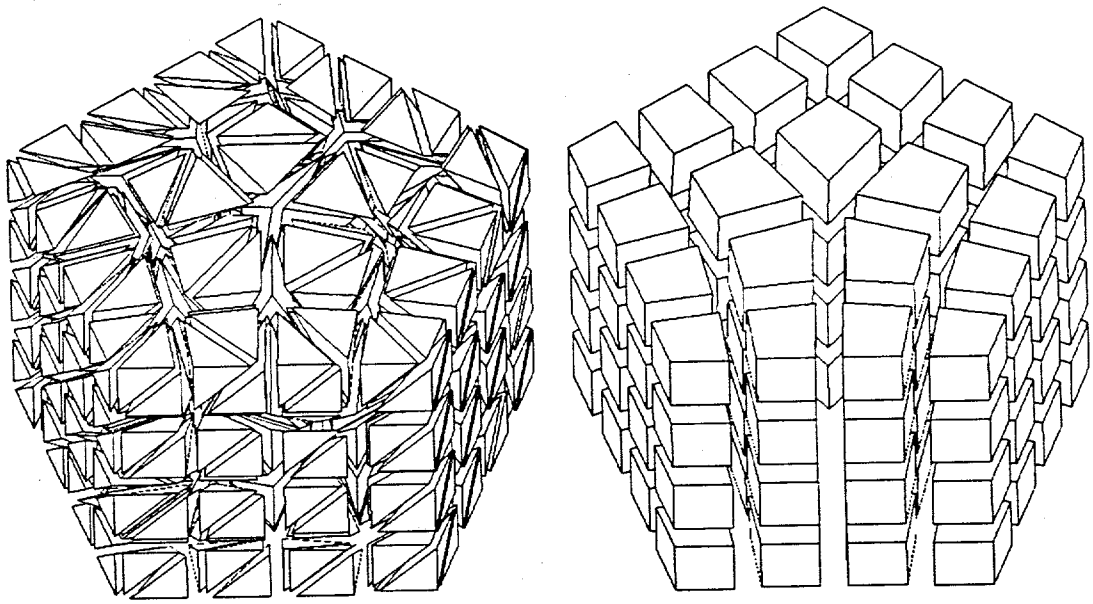
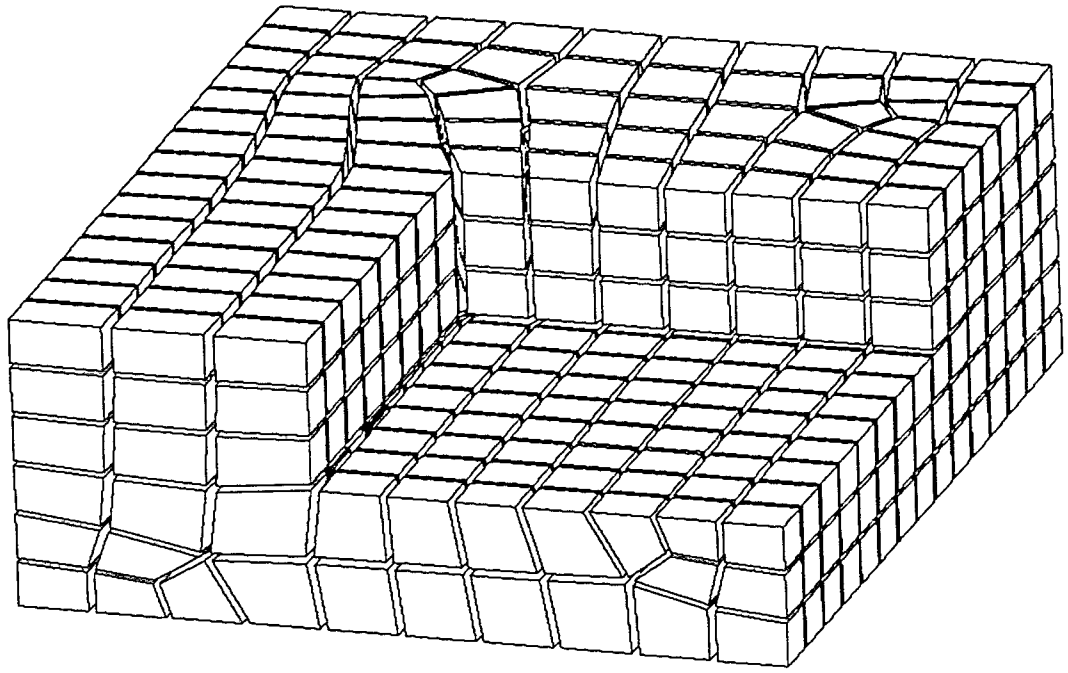
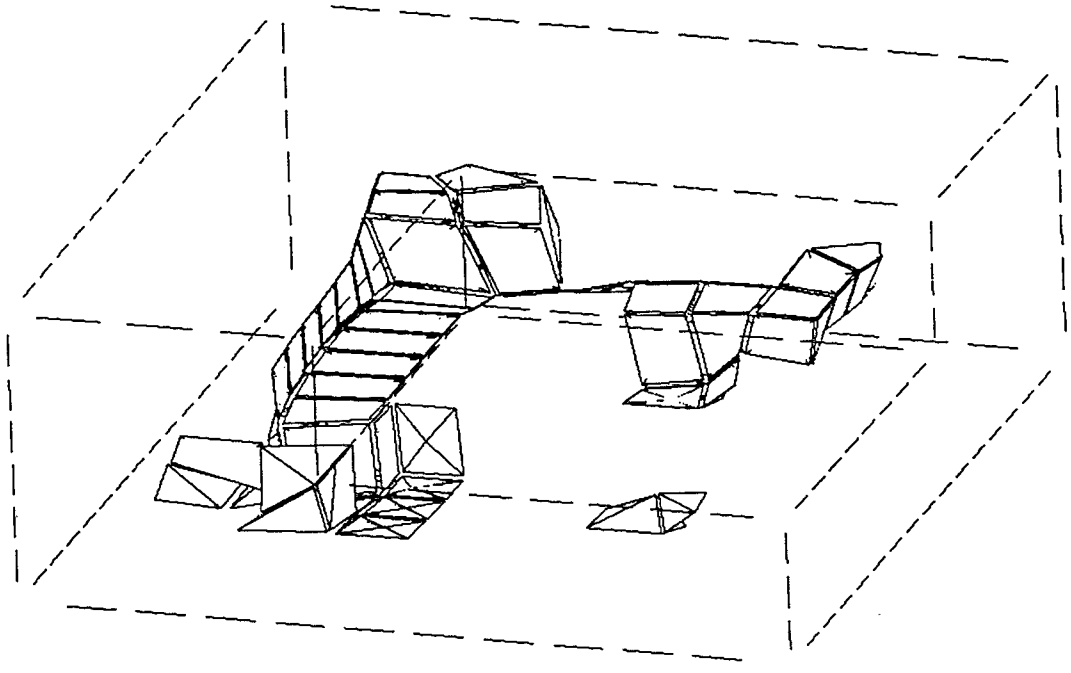


Figure 10. Sweepable model (Penta) resolved with H-Morph

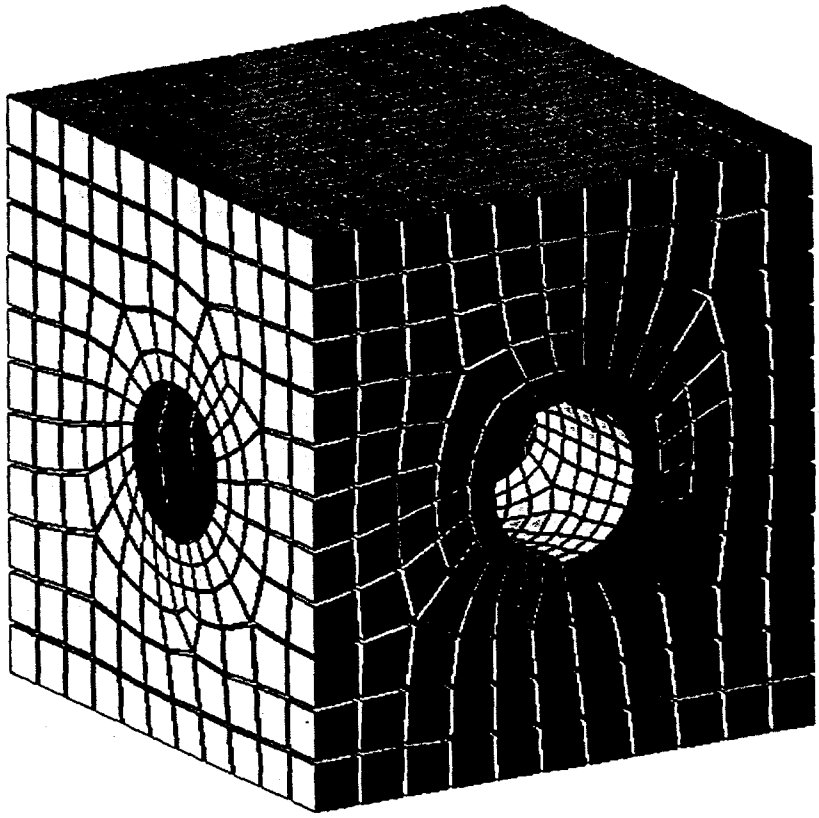


(a)

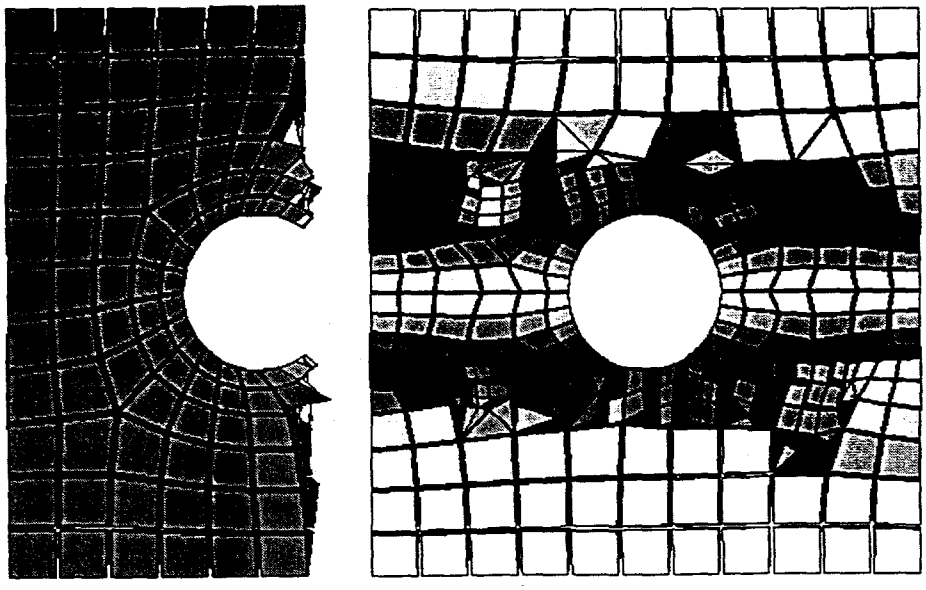


(b)

Figure 11. Cutblock model. (a) Hex elements formed from H-Morph, (b) Pyramids and Tetrahedra Remaining after H-Morph



(a)



(b)

Figure 12 Cube with intersecting cylinders. (a) Finite Element Mesh, (b) Cut-away views of Finite Element Mesh

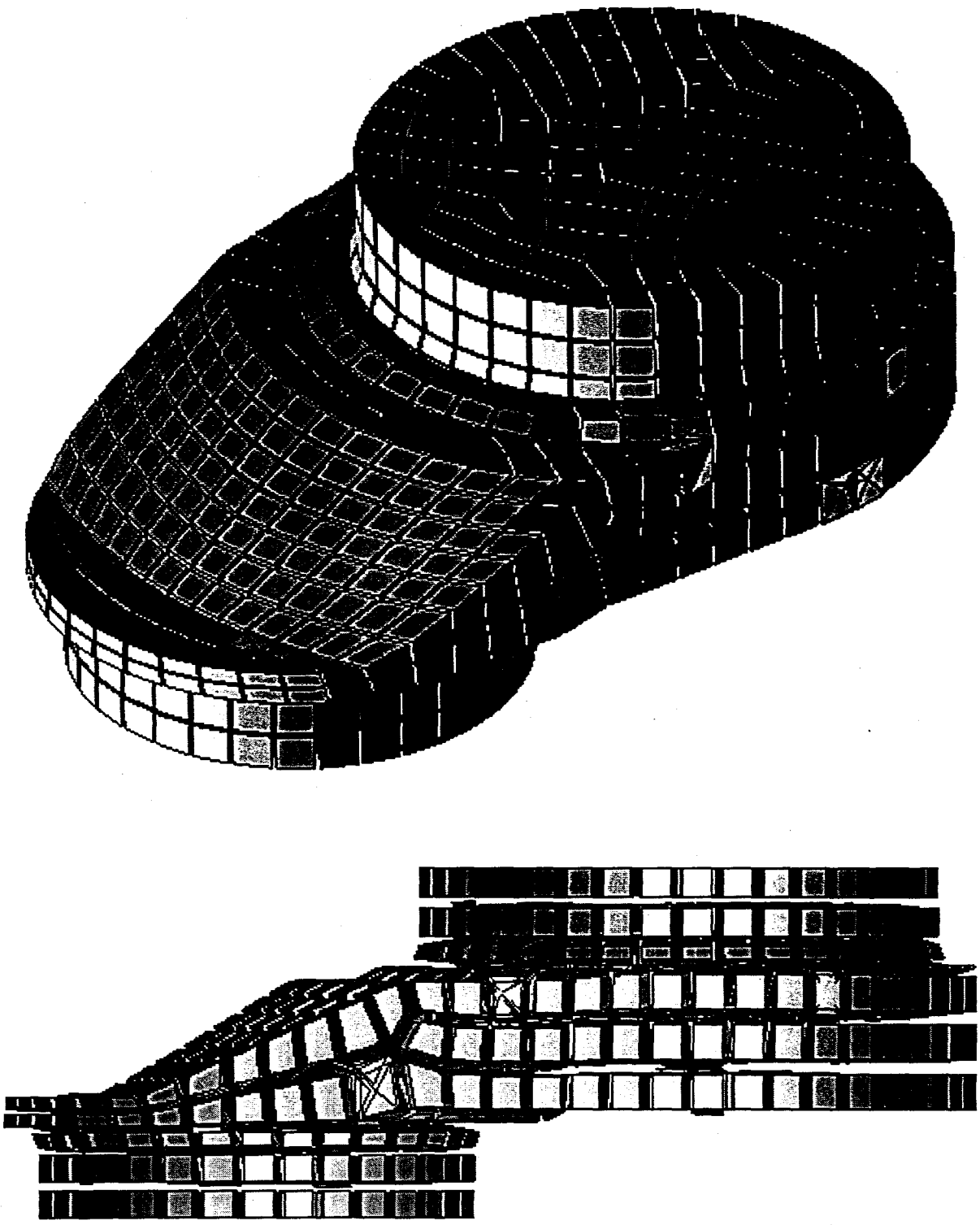


Figure 13 Throw model (Courtesy of Sandia National Labs)

Table 1. Tetrahedral transformations used in H-Morph

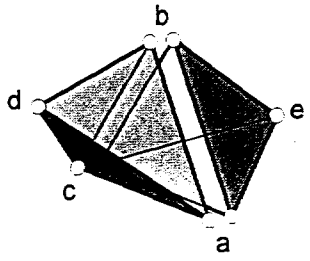
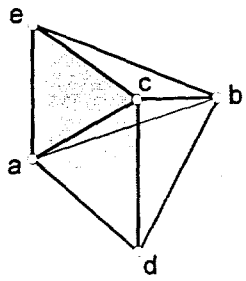
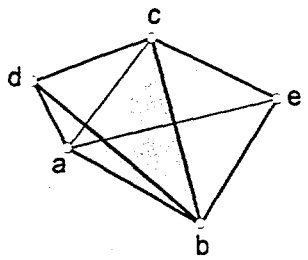
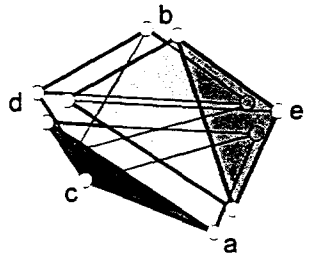
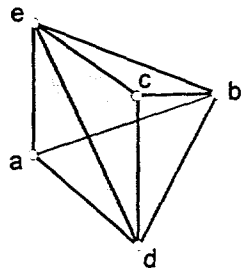
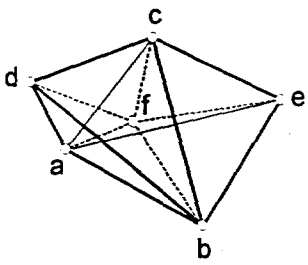
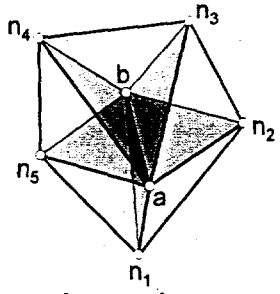
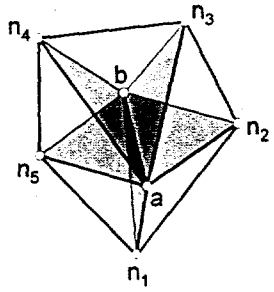
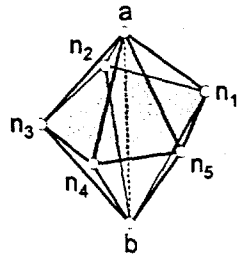
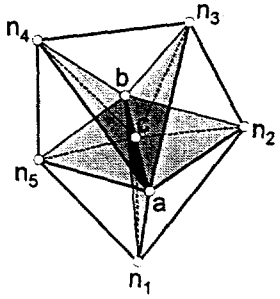
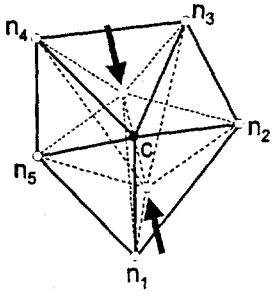
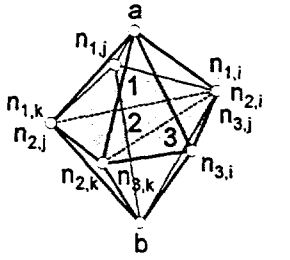
swap23	swap22	face-split
 abce, acbd	 aceb, adcb	 abce, acbd
 abde, bcde, cade	 adeb, edcb	 abfe, bcfe, cafe, bafd, cbfd, acfd
edge-split	**edge-collapse**	**edge-suppress**
 abn_1n_2, abn_2n_3, abn_in_{i+1},...abn_Nn_1 N = no. adjacent tets at edge ab	 abn_1n_2, abn_2n_3, abn_in_{i+1},...abn_Nn_1 N = no. adjacent tets at edge ab	 abn_1n_2, abn_2n_3, abn_in_{i+1},...abn_Nn_1 N = no. adjacent tets at edge ab
 acn_1n_2, acn_2n_3, acn_in_{i+1},...acn_Nn_1, cbn_1n_2, cbn_2n_3, cbn_in_{i+1},...cbn_Nn_1		 $n_{1,k}n_{1,j}n_{1,i}a$, $n_{1,i}n_{1,j}n_{1,k}b$, $n_{2,k}n_{2,j}n_{2,i}a$, $n_{2,i}n_{2,j}n_{2,k}b$,... $n_{M,k}n_{M,j}n_{M,i}a$, $n_{M,i}n_{M,j}n_{M,k}b$ M = no. unique trias in polygon $P=\{n_1,n_2,...n_N\}$

Table 2. Summary of Examples

Model		Blocky Model Figure 9	Penta Figure 10	Cutblock Figure 11	Cube w/cyls Figure 12	Throw Figure 13
Tets	Number/%	0	0	243/24.4	5888/66.82	2638/48.89
	% Volume	0	0	1.37	7.58	3.67
Pyrs	Number/%	0	0	114/11.45	0	934/17.31
	% Volume	0	0	1.45	0	3.39
Hexes	Number/%	758	80	639/64.16	2924/33.18	1824/33.8
	% Volume	**100**	**100**	**97.18**	**92.42**	**92.94**
Total	Elements	758	80	996	8812	5396
	Nodes	1252	155	1047	4922	3220

REFERENCES

1 W.A. Cook, and W.R. Oakes, 'Mapping Methods for Generating Three-Dimensional Meshes', *Comp. Mech. Engrg*, Aug., 67-72 (1982).

2 J. F. Thompson, *Numerical Grid Generation, Foundation and Applications*, Elsevier, 1985.

3 J. F. Thompson, B. K. Soni and N. P. Weatherill ed., *Handbook of Grid Generation*, CRC Press, London, 1999.

4 D. R. White, 'Automated Hexahedral Mesh Generation by Virtual Decomposition', *Proc. 4th Int. Meshing Roundtable*, 1995, pp. 165-176.

5 M. L. Staten, S. A. Canann, and S. J. Owen, 'BMSWEEP: Locating Interior Nodes During Sweeping', *Proc. 7th Int. Meshing Roundtable*, 1998, pp. 7-18.

6 L. Mingwu, S. E. Benzley, G. Sjaardema and Tim Tautges, 'A Multiple Source and Target Sweeping Method for Generating All-Hexahedral Finite Element Meshes', *Proc. 5th Int. Meshing Roundtable*, 1998, pp. 217-228.

7 T. D. Blacker, 'The Cooper Tool', *Proc. 5th Int. Meshing Roundtable*, 1996, pp.13-29.

8 T. J. Tautges, S. Liu, Y. Lu, J. Kraftcheck and R. Gadh, 'Feature Recognition Applications in Mesh Generation', *Trends in Unstructured Mesh Generation, ASME*, AMD **220**, 117-121 (1997).

9 S. Liu and R. Gadh, 'Basic LOgical Bulk Shapes (BLOBS) for Finite Element Hexahedral Mesh Generation', *5th Int. Meshing Roundtable*, 1996, pp. 291-306.

10 M. A. Price and C. G. Armstrong, 'Hexahedral Mesh Generation by Medial Surface Subdivision: Part I', *Int. J. Numer. Meth. Engng.*, **38**, 3335-3359 (1995).

11 M. A. Price and C. G. Armstrong, 'Hexahedral Mesh Generation by Medial Surface Subdivision: Part II', *Int. J. .Numer. Meth. .Engng.*, **40**, 111-136 (1997).

12 R. Schneiders, 'A Grid-Based Algorithm for the Generation of Hexahedral Element Meshes', *Engng. Comp.*, **12**, 168-177 (1996).

13 M. Wierse, J. Cabello and Y. Mochizuki, (1998) 'Automatic Grid Generation with HEXAR', *Proc. 6th International Conference on Numerical Grid Generation in Computational Field Simulations, ed. M. Cross et al.*, University of Greenwich, UK., 1998, pp. 843-852.

14 S. A. Canann, 'Plastering and Optismoothing: New Approaches to Automated, 3D Hexahedral Mesh Generation and Mesh Smoothing', *Ph.D. Dissertation*, Brigham Young University, Provo, UT, 1991.

15 T. D. Blacker and R. J. Myers, 'Seams and Wedges in Plastering: A 3D Hexahedral Mesh Generation Algorithm', *Engng Comp.*, **2**, 83-93 (1993).

16 T. J. Tautges, T. Blacker and S. Mitchell, 'The Whisker-Weaving Algorithm: A Connectivity Based Method for Constructing All-Hexahedral Finite Element Meshes', *Int. J. Numer. Meth. Engng*, **39**, 3327-3349 (1996).

17 P. Murdoch and S. E. Benzley, 'The Spatial Twist Continuum', *Proc. 4th Int. Meshing Roundtable*, 1995, pp. 243-251.

18 P. Tuchinsky and B. W. Clark, 'The Hex-Tet, Hex-Dominant Automesher: An Interim Progress Report', *Proc. 6th Int. Meshing Roundtable*, 1997, pp.183-193.

19 S. Mitchell, 'A Characterization of the Quadrilateral Meshes of a Surface Which Admit A Compatible Hexahedral Mesh of the Enclosed Volume', *5th MSI Workshop in Comp. Geom.*, 1995.

20 N. T. Folwell and S. A. Mitchell, 'Reliable Whisker Weaving via Curve Contraction', *Proc. 7th Int. Meshing Roundtable*, 1998, pp. 365-378.

21 S. J. Owen, M. L. Staten, S. A. Canann and S. Saigal, 'Q-Morph, An Indirect Approach to Advancing Front Quad Meshing', *Int. J. Numer. Meth. Engng.*, **44**, 1317-1340 (1999)

22 R. J. Meyers, T. J. Tautges and P. M. Tuchinsky, 'The Hex-Tet Hex-Dominant Meshing Algorithm as Implemented in Cubit', *Proc. 7th Int. Meshing Roundtable*, 1998, pp. 151-158.

23 P.L. George, F. Hecht and E. Saltel, 'Automatic Mesh Generator with Specified Boundary', *Comp. Meth. Appl. Mech. Engng.*, **92**, 269-288 (1991)

24 P. L. George and H. Borouchaki, *Delaunay Triangulation and Meshing*, Hermes, Paris, 1998.

25 D. L. Marcum and N. P. Weatherill, 'Unstructured Grid Generation Using Iterative Point Insertion and Local Reconnection', *AIAA Journal*, **33**(9),1619-1625 (1995).

26 S. H. Lo, 'Volume Discretization into Tetrahedra - II. 3D Triangulation by Advancing Front Approach', *Comp. Struct.*, **39**(5), 501-511 (1991).

27 J. Hipp and R. Lober, 'Plastering: Automated All-Hexahedral Mesh Generation Through Connectivity-Resolution', *Proc. 3rd Int. Meshing Roundtable*, 1994.

28 B. Joe, 'Construction of Three-Dimensional Improved-Quality Triangulations Using Local Transformations', *Siam J. Sci. Comput.*, **16**, 1292-1307 (1995).

29 N.P. Weatherill and O. Hassan, 'Efficient Three-dimensional Delaunay Triangulation with Automatic Point Creation and Imposed Boundary Constraints', *Int. J. Numer. Meth. Engrg.*, **37**, 2005-2039 (1994)

30 S. J. Owen, 'Non-Simplical Unstructured Mesh Generation', *Ph.D. Dissertation*, Carnegie Mellon University, 1999

31 S. J. Owen, 'Constrained Triangulation: Application to Hex-Dominant Mesh Generation', *Proc. 8th International Meshing Roundtable*, 1999

32 S. A. Canann, 'Optismoothing: An Optimization-Driven Approach to Mesh Smoothing', *Finite Elements in Analysis and Design*, **13**, 185-190 (1993).

33 T. D. Blacker, and M. B. Stephenson, 'Paving: A New Approach to Automated Quadrilateral Mesh Generation', *Int J. Numer. Meth. Engng*, **32**, 811-847 (1991).

34 B. W. Clark, and S. E. Benzley 'Development and evaluation of a degenerate seven-node hexahedron finite element', *Proc. 5th International Meshing Roundtable*, 1996, pp. 321-322.

35 L. A. Freitag, 'On Combining Laplacian and Optimization-Based Mesh Smoothing Techniques', *Trends in Unstructured Mesh Generation, ASME*, AMD **220**, 37-43 (1997).

36 S. A. Canann, J. R. Tristano and M. L. Staten, 'An Approach to Combined Laplacian and Optimization-Based Smoothing for Triangular, Quadrilateral, and Quad-Dominant Meshes', *Proc, 7th Int. Meshing Roundtable* (1998).

37 S. J. Owen, S. A. Canann and S. Saigal, 'Pyramid Elements for Maintaining Tetrahedra to Hexahedra Conformability', *Trends In Unstructured Mesh Generation, ASME*, AMD **220**, 123-130 (1997).

38 S. Kelley, 'Element Shape Checking', *Ansys Theory Manual*, chapter **13**, (1998).