Advancing Front Quadrilateral Meshing Using Triangle Transformations

Steven J. Owen\textsuperscript{1,2}, Matthew L. Staten\textsuperscript{2}, Scott A. Canann\textsuperscript{1,2} and Sunil Saigal\textsuperscript{1}

\textsuperscript{1}Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania, U.S.A.
\textsuperscript{2}Ansys Inc., 275 Technology Drive, Canonsburg, Pennsylvania, 15317, U.S.A.
steve.owen@ansys.com

Abstract

Quad-morphing is a new technique used for generating quadrilaterals from an existing triangle mesh. Beginning with an initial triangulation, triangles are systematically transformed and combined. An advancing front method is used to determine the order of transformations. An all-quadrilateral mesh containing elements aligned with the area boundaries with few irregular internal nodes can be generated.

KEY WORDS: mesh generation, quadrilateral, advancing front, surface meshing, Q-Morph, Paving

1. Introduction

Previous methods for unstructured quadrilateral meshing have included both direct and indirect methods. Indirect methods (Lo, 1989; Johnston, 1991; Lee, 1994; Borouchaki, 1998) include procedures that require an initial triangle mesh. Adjacent triangles are combined systematically, in most cases resulting in an all-quadrilateral mesh. While these methods can be fast, they can sometimes leave a large number of irregular nodes. An irregular node on the interior of a quadrilateral mesh is one that has more or less than four adjacent elements. Direct methods, on the other hand, do not involve an initial triangle mesh. Quadrilaterals are instead placed directly onto the surface. Quadrilaterals may be placed after first decomposing the surface into simpler regions (Baehmann, 1987; Talbert, 1991; Tam, 1991; Joe, 1995) or by using an advancing front approach (Zhu, 1991; Lo, 1985; Blacker, 1991). In most cases, direct methods provide higher quality elements with fewer irregular nodes.

Of the direct, quadrilateral methods, the paving algorithm (Blacker, 1991) provides several desirable characteristics. Blacker describes these as “(a) Boundary Sensitive. Mesh contours should closely follow the contours of the boundary. This characteristic is of particular importance since well-shaped elements are usually desirable near the boundary, (b) Orientation Insensitive. Rotating or translating a given geometry should not change the resulting mesh topology. A mesh generated in a transformed geometry should be equivalent to the original mesh transformed, and (c) Few irregular nodes. This is a critical mesh topology feature because the number of elements sharing a node controls the final shape of the elements, even after smoothing. Thus a mesh with few irregular nodes, especially near the boundary where element shape is critical, is often preferred.” The paving algorithm is currently in wide use. Since its initial development, it has been enhanced to incorporate three-dimensional surfaces (Cass, 1996), as well as other improvements (White, 1997).
In spite of the beneficial characteristics of paving, some quality and performance issues must be addressed. The advancing front method used by the paving technique requires many expensive intersection calculations as each row is placed in order to avoid overlapping elements. Figure 1(a) shows a simple case where intersection checks must be made. Figure 1(b) shows another case often encountered during paving where colliding fronts must merge. If element sizes differ greatly, poor element quality can often result.

![Figure 1. (a) First row of elements placed using paving algorithm illustrating interference of opposing elements. (b) Large element size differences between opposing fronts often encountered in paving leading to poor meshes.](image)

This paper proposes an alternative to the traditional paving algorithm. The proposed Quad-morphing (Q-Morph) algorithm maintains the desirable features of paving while addressing some of its weaknesses. Q-Morph can be categorized as an unstructured, indirect method that utilizes an advancing front algorithm to form an all-quad mesh. As an indirect method it is able to take advantage of local topology information from the initial triangulation. Unlike other indirect methods it is able to generate boundary sensitive rows of elements, with few irregular nodes.

2. Outline of Quad-Morphing Algorithm

Quad-morphing is briefly outlined in the following steps:

1. **Initial Triangle Mesh.** The surface is first triangulated. This may be done using any surface triangulation method. Any sizing (Owen, 1997) or adaptivity information should be built into the initial triangulation. The local sizing for the final quadrilateral mesh will roughly follow that of the triangle mesh.

2. **Front Definition.** The initial front is defined from the initial triangle mesh. Any edge in the triangulation that is adjacent to only one triangle becomes part of the initial front.

3. **Front Edge Classification.** Each edge in the front is initially sorted according to its state. The state of a front edge defines how the edge will eventually be used in forming a quadrilateral. Angles between adjacent front edges determine the state of an individual front. Front edges will be updated and reshuffled as the algorithm proceeds. Figure 2 shows the four possible states of a front, where the front edge is indicated by the bold line.
4. **Front Edge Processing.** Each front edge is individually processed to create a new quadrilateral from the triangles in the initial mesh. Figure 3(a) shows front $N_A$-$N_B$ in the triangulation ready to be processed. Front edges are handled differently according to their current state classification. As quadrilaterals are formed, the front is redefined and adjacent front edge states are updated. The current front always defines the interface between quadrilateral elements in the final mesh and triangle elements in the initial triangle mesh. This process can be further subdivided into the following sub-steps:

- **Check for Special Cases.** Before proceeding to construct a quadrilateral from the current front, several special case scenarios are checked. These include situations where large transitions or small angles exist local to the front. In these cases a **seam**, or **transition seam** operation is performed.

- **Side Edge Definition.** Using the front edge as the initial base edge of the quadrilateral, side edges are defined. Side edges may be defined by using an existing edge in the initial triangle mesh, by swapping the diagonal of adjacent triangles, or by splitting triangles to create a new edge. In Figure 3(b), side edge $N_B$-$N_C$ shows the use of an existing edge, while the side edge $N_A$-$N_D$ was formed from a local swap operation.

- **Top Edge Recovery.** The final edge on the quadrilateral is created by an **edge recovery** process. During this process, the local triangulation is modified by using local edge swaps to
enforce an edge between the two nodes at the ends of the two side edges. Edge Nc-Nd in Figure 3(c) was formed from a single swap operation. Any number of swaps may be required to form the top edge.

- **Quadrilateral Formation.** Merging any triangles bounded by the front edge and the newly created side edges and top edge as shown in Figure 3(d) forms the final quadrilateral.

- **Local Smoothing.** The mesh is smoothed locally to improve both quadrilateral and triangle element quality as shown in Figure 3(e).

- **Local Front Reclassification.** The front is advanced by removing edges from the front that have two quadrilateral adjacencies and adding edges to the front that have one triangle and one quadrilateral adjacency. New front edges are classified by state. Existing fronts that may have been adjusted in the smoothing process are reclassified.

Front edge processing continues until all edges on the front have been depleted, in which case an all-quadrilateral mesh will remain, assuming an even number of initial front edges. When an odd number of boundary intervals is provided, a single triangle must be generated, usually towards the interior of the mesh.

5. **Topological Clean-up.** Element quality is improved by performing local quadrilateral transformations in an attempt to improve the individual edge valences at the nodes of the mesh.

6. **Smoothing.** A final smoothing pass is performed further improving the element qualities.

### 3. Implementation

#### 3.1 Front Definition and Classification

The initial set of front edges is defined from the initial triangulation. All edges in the triangulation adjacent to a single triangle are used as the front. The state of a front edge is determined by computing the angle at the nodes on either end of the edge with each of its adjacent front edges. Practically, the state of a front edge is defined by two bits, the first representing the state at the left node and the second, the state at the right node. If the angle at either node is less than a specified tolerance ($3\pi/4$), the node bit is set (1); otherwise it is unset (0).

Angles at the nodes on the front can be approximated by summing the angles at adjacent triangles. In direct advancing front methods (Cass, 1996), angles must be computed by first evaluating the surface normal and projecting edge vectors to a tangent plane. By approximating the angle at the front from the adjacent triangles, expensive geometric evaluations can be eliminated.

Edges are placed on one of four state lists as shown in Figure 2. Classifying front edges according to states serves two purposes. First, it defines which edges must be defined before a complete quadrilateral can be formed. Side edges must be defined only at the side of the front where the state bit has not been set. Second, it prioritizes which fronts will be processed first. Front edges in state 1-1 are given first priority followed by edges in states 0-1 and 1-0, followed by edges in state 0-0.
3.2 Front Edge Processing

Front edges are processed one at a time to form quadrilaterals from the initial triangulation. A front edge is popped from one of the four state lists, drawing from the higher states first. Priority is also given to the lowest level edge on the list. Edges in level zero are those on the initial front; level one are those on the front after the first row of quadrilaterals has been placed; level two after the second row; and so on. This ensures that an entire row of quadrilaterals will be placed before starting a new row.

Where large transitions are required, experience has shown that placing smaller quads first generally produces a better graded mesh. In order to do so, it is sometimes necessary to select short, higher level fronts before selecting longer lower level fronts. The criteria used for selecting the next front to be processed is, therefore, based not only on the current state and level of the front but also on its size.

3.2.1 Side Edge Definition

The current state of a front edge determines how the edge is processed. Front edges in states 0-0, 1-0 and 0-1 must first define either one or two side edges. A side edge may be formed in one of three ways: (1) an existing edge in the initial triangle mesh may be used, (2) the diagonal between two adjacent triangles may be swapped, or (3) an edge may be created by splitting a pair of triangles.

![Figure 4. Side edge selection](image)

Figure 4 shows a situation in which an existing edge is used. A new side edge is to be defined at node Nk, which is a node on the front between edges EF1 and EF2. The ideal vector $V_k$ for the new side edge is defined by bisecting the vectors formed by EF1 and EF2. Angles $\theta_i$ are computed between $V_k$ and all edges, $E_i$, of triangles sharing node Nk. The edge with the smallest angle $\theta_i$ is selected as the candidate side edge. The edge is selected, provided $\theta_i$ is less than a constant $\epsilon$ ($\pi/6$). Edge E2 in Figure 4 is selected as the side edge in this situation.

When there is no angle $\theta_i$ less than $\epsilon$, one of two options may be used. The opposite edge, (Eo in Figure 5) may either be swapped or split. The swap option is used if the angle $\beta$ between $V_k$ and $V_m$ is less than $\epsilon$. The split option is performed if $\beta > \epsilon$ or the resulting length of $E_k$ from a swap is excessively long compared to $E_{f1}$ and $E_{f2}$. In this latter case, a new node Nn is defined, splitting edge Eo at the intersection of vector $V_k$ and edge Eo. Edges Eo and Em are also added to the triangle mesh, splitting the two triangles adjacent to edge Eo. Edge Eo is then used as the side edge of the prototype quad. The following shows a summary of the criteria for selection or creation of edge $E_k$, to be used as a side edge in the new quadrilateral:
EI for swap N N for < and N N

2 split N N otherwise

\[ E_k = \begin{cases} E_i & \text{for } \theta < \varepsilon \\ \text{swap } \Rightarrow \overline{N_k N_m} & \| N_k N_m \| < \sqrt{3} \left( \| E_{P1} \| + \| E_{P2} \| \right) \\ \text{split } \Rightarrow \overline{N_k N_n} & \text{otherwise} \end{cases} \] [1]

Figure 5. Side edge creation

3.2.2 Top Edge Recovery

Once the base and the two sides of the quadrilateral have been formed, the next step is to define the top edge. This is done by recovering the edge between the end nodes of the two sides. The edge recovery technique, which was presented independently in the literature by Jones (1990), Sloan (1993), and George (1991), is used commonly in boundary constrained Delaunay triangle meshing. Edge recovery involves systematically swapping edges between adjacent triangles until an edge is achieved between the desired nodes. An example of an edge recovery process is shown in Figure 6. The triangulation before recovery is shown at the top left with the successive swaps numbered. In this example a total of four local swaps was required to recover the edge Nc-Nd from the triangulation. Algorithm 1 details the transformations necessary to accomplish the recovery.
1. LET $S$ be the line segment from $N_C$ to $N_D$
2. LET $\mathcal{A}(S)$ be a list of edges $E_i$ that are intersected by $S$ (see algorithm 2)
3. FOR EACH $E_i \in \mathcal{A}(S)$
   4. LET $T(E_i)$ be the set of 2 triangles adjacent $E_i$
   5. LET $T^{-1}(E_i)$ be the set of 2 triangles where the diagonal edge $E_i$ has been swapped.
   6. IF area of both triangles in $T^{-1}(E_i) > 0$ THEN
      7. Form $T^{-1}(E_i)$
      8. Delete $E_i$ from $\mathcal{A}(S)$
      9. IF $E_j$ be the edge common to both triangles in $T^{-1}(E_i)$
         10. IF $E_j$ intersects $S$ add $E_j$ last on $\mathcal{A}(S)$,
         11. ELSE,
             Place $E_i$ last on $\mathcal{A}(S)$
      12. NEXT $E_i$ on $\mathcal{A}(S)$

Algorithm 1. Edge recovery process

The edge recovery process requires that an initial set of edges, $\mathcal{A}(S)$, through which the recovered edge will pass, be first compiled. Algorithm 2 and Figure 7 detail how this may be accomplished.

3.2.3 3D Edge Recovery

Current literature assumes that the edge recovery process will be performed on a planar domain. Since this condition cannot be guaranteed in this application, an extension of the edge recovery process to include three-dimensional surfaces was necessary. Specifically, the dot product calculations of steps 5 and 16 in Algorithm 2 must be performed on vectors in a plane that is tangent to the surface. The tangent plane can be approximated from the neighboring triangles. For example, the tangent plane normal $P_i$ at edge $E_i$ can be estimated from the average normal vector of triangles $T_i(E_i)$ and $T_{i+1}(E_i)$. The dot product calculation in step 16 can then be replaced as:
Step 5 can be modified in a similar manner. An approximated tangent plane normal, $\mathbf{p}_c$, at $N_C$, can be defined as the average normal vector of the triangles in $T(N_C)$. The dot product calculation can then be replaced as:

$$\left( (\mathbf{p}_c \times \mathbf{v}_s) \cdot \mathbf{p}_c \right) \cdot \left( (\mathbf{p}_c \times \mathbf{v}_k) \times \mathbf{p}_c \right) > 0 \quad \text{and} \quad \left( (\mathbf{p}_c \times \mathbf{v}_s) \times \mathbf{p}_c \right) \cdot \left( (\mathbf{p}_c \times \mathbf{v}_{k+1}) \times \mathbf{p}_c \right) < 0$$

**Algorithm 2. Formation of $\Lambda(S)$**

1. LET $T(N_C)$ be the ordered set of ccw triangles and quads adjacent $N_C$, $T_k(N_C) \in T(N_C)$
2. LET $E(N_C)$ be the ordered set of ccw edges adjacent $N_C$; $E_k(N_C) \in E(N_C)$, where $E_k(N_C)$ and $E_{k+1}(N_C)$ are on $T_k(N_C)$
3. LET $\mathbf{v}_k$ be the vector from $N_C$ in direction of $E_k(N_C)$
4. LET $\mathbf{v}_s$ be the vector from $N_C$ to $N_D$
5. FOR EACH $T_k(N_C) \in T(N_C)$
   a. IF $\mathbf{v}_s \cdot \mathbf{v}_k > 0$ and $\mathbf{v}_s \cdot \mathbf{v}_{k+1} < 0$, LET $T_i(E_i) = T_k(N_C)$
6. LET $E_i$ be the edge opposite $N_C$ on $T_i(N_C)$
7. IF $E_i$ is not on front THEN, Add $E_i$ to $\Lambda(S)$, ELSE fail
8. WHILE not done
   a. LET $T_{i+1}(E_i)$ be the triangle adjacent $E_i$ where $T_{i+1}(E_i) \neq T_i(E_i)$
   b. IF $N_D$ is on $T_{i+1}(E_i)$, THEN done
   c. LET $T_i(E_i) = T_{i+1}(E_i)$
   d. LET $N_i$ be the node opposite $E_i$ on $T_i(E_i)$
   e. LET $\mathbf{v}_i$ be the vector from $N_C$ to $N_i$
   f. LET $E_n$ be the next ccw edge on $T_i(E_i)$ from $E_i$
   g. LET $E_{n+1}$ be the next cw edge on $T_i(E_i)$ from $E_i$
   h. IF $\mathbf{v}_s \cdot \mathbf{v}_i < 0$, THEN $E_i = E_n$, ELSE $E_i = E_{n+1}$
   i. IF $E_i$ is not on front THEN, Add $E_i$ to $\Lambda(S)$, ELSE fail
18. CONTINUE

**Figure 7. Formation of $\Lambda(S)$**
3.3 Quadrilateral Formation

The quadrilateral is formed from the edge on the front, two side edges, and recovered top edge. Before forming the quadrilateral, the triangles contained within the four edges must first be deleted. This can be done with a procedure that starts with the triangle adjacent to the front edge and recursively advancing to adjacent triangles, deleting them as it proceeds. Unused nodes and edges are also removed. The recursion continues until the top or side edges of the prototype quadrilateral are encountered.

3.4 Local Smoothing

Smoothing is an important part of the Q-Morph algorithm. Node locations local to the new quadrilateral are readjusted to improve element shape. This includes nodes both behind and in front of the current front. Local smoothing accomplished before processing the next front, as smoothing angles between adjacent fronts will affect the front states and hence the final topology of the quadrilateral mesh. In practice, any node on the new quadrilateral and any node connected by an edge are smoothed. Nodes on the front must be handled differently than those behind or ahead of the front.

For nodes not located on the current front, a simple Laplacian smooth (Field, 1988) is adequate, or alternatively, a modified length weighted Laplacian smooth as suggested by Blacker (1991). Since it is at the front where the new quadrilateral elements are formed, it is more critical at these nodes that the smoothing produce well proportioned quadrilaterals. A modified form of the smoothing process suggested by Blacker is used for the row nodes defined in that reference. These are nodes connected to exactly two adjacent quadrilaterals at the front. The smoothing process presented by Blacker involves an isoparametric smooth (Hermann, 1976) followed by corrections for squareness and angle smoothness. For cases where large transitions may be involved, it is useful to take advantage of sizing information provided by the triangles ahead of the front. As a result, an improved transition can be achieved.

![Figure 8. Definition of edge length $l_D$ at node on front $N_k$.](image)

Let $l_D$ be the length of the edge $N_k-N_j$ where $N_k$ is the node on the front to be smoothed as shown in Figure 8. Where a very large transition in element size is required, $l_D$ can be defined as the average length of all edges connected to $N_k$. For smaller transitions, fewer irregular nodes will be created if equation 4 is used. Let $n$ be the number of nodes ahead of the front connected to $N_k$, then $l_D$ can be defined as an average of edge lengths on adjacent quadrilaterals and edges ahead of the front as follows:
The method used for computing $l_D$ is decided purely on heuristics. For the current implementation, if the ratio of largest to smallest edge length, $t_r$, on the boundary is less than 2.5, then the smoothing method proposed by Blacker (1991) is used unmodified. This method is preferred since it tends to produce the fewest number of irregular nodes. As $t_r$ increases, it is necessary to introduce irregular nodes so that a smooth transition may be afforded. Equation 4 is used for $l_D$ when $t_r$ is greater than 2.5, and the average length of adjacent edges to $N_k$ is used when $t_r$ is greater than 20.

When smoothing nodes at the front, as a result of improving the quadrilaterals, it is possible that the triangles immediately ahead of the front become inverted. While the Q-Morph algorithm does not require triangles to be near equilateral, it does rely on the fact that all triangles are uninverted throughout the meshing process. To ensure that this does not occur, triangles and quadrilaterals neighboring the smoothed node must be checked for consistent normals. In the case of an inverted element, an adjustment must be made to the new node location. The node location can be adjusted incrementally on a vector from the old location to the new location until all neighboring elements are no longer inverted.

### 3.5 Local Update and Reclassification of Fronts

Once a new quadrilateral has been formed, it becomes necessary to update the current list of fronts. Fronts are redefined so that edges now adjacent to both a triangle and a quadrilateral are placed on one of the four state lists and edges no longer adjacent to a triangle are removed from the state lists. In addition, other nearby edges on the front may need to be updated. By smoothing nodes on the front, angles between adjacent fronts may have been changed; thus necessitating moving fronts to an alternate state list.

### 3.6 Closing the Front

When defining a new side edge, the opposite node, $N_m$, as shown in Figure 9, may lie on an opposing front. Edge $E_k$ may have been selected from the existing triangles as in Figure 4, or from a swap operation as in...
Figure 5. In either case, $E_k$ can only be used if the number of edges on each resulting front loop is even. A front loop may be defined as all edges on a front comprising a continuous unbroken ring. Any number of loops may be active at a given time in the process of meshing. In order to ensure an all-quadrilateral mesh, it is required that any individual loop be comprised of an even number of edges. Selecting an edge to be used as a new side edge may result in the formation of a loop with an odd number of front edges. To avoid this occurrence, the potential number of front edges on the new loop about to be formed is first determined. If the number of edges is even, then the selection is made and a new loop is defined. If the number of edges on the new loop is odd, no connection is made. Instead the edge, $E_k$, is split creating a new node, $N_n$, as shown in Figure 10. This permits a subsequent side selection operation to define an even number of front edges on adjacent loops.

If the side edge is to be created from a swap or split operation, as in Figure 5, the edge $E_o$ should first be checked to see if it is part of the opposing front. Since swapping or splitting $E_o$ would destroy the continuity of the front, the operation should not be performed. For this reason, it is advantageous, when $N_m$ is on an opposing front, to allow for a larger value of $\varepsilon$. This increases the chances of selecting an existing edge and closing the front.

3.7 Seams

When the angle, $\alpha$, between two adjacent edges on the front is small, then a seaming operation is performed. Although paving (Blacker, 1991) incorporates a seaming operation, it must also be defined within the context of the Q-Morph algorithm, in order to account for triangles ahead of the front. In addition to angle $\alpha$, the criteria for seaming is also based on the number of quadrilateral elements, $n_Q$, adjacent to the node to be seamed. Blacker proposes the following criteria for seaming:

$$\begin{cases} 
\alpha < \varepsilon_1 & \text{for } n_Q \geq 5 \\
\alpha < \varepsilon_2 & \text{otherwise}
\end{cases}$$

where $\varepsilon_1 < \varepsilon_2$ \hspace{1cm} [5]

![Seaming operation](image)

**Figure 11. Seaming operation**

To accomplish the seam, nodes $N_{k-1}$ and $N_{k+1}$, shown in Figure 11, must be merged. Let $N_k$ be the node on the front whose angle, $\alpha$, satisfies equation 5. The temporary edge, $E_o$, connecting $N_{k-1}$ and $N_{k+1}$, if not already part of the initial triangle mesh, is first recovered using Algorithm 1 above. Knowing $E_o$, its
adjacent triangle comprising nodes $N_{k-1}, N_k, N_t$ can be determined. With this information, all triangles and
nodes bounded by the quadrilateral composed of nodes $N_{k-1}, N_k, N_{k+1}, N_t$ can be deleted. Finally nodes $N_{k-1}$
and $N_{k+1}$ can be merged at a location midway between their initial positions. Local smoothing is then
performed, followed by an update of the states of any adjacent fronts that may have changed. In rare cases,
the new location of node $N_{k+1}$ may result in one or more inverted elements. In this case, an optimization
based smoothing algorithm (Canann, 1998) is employed which adjusts the node location with the objective
of improving a local shape metric for neighboring elements.

Another operation described by Blacker (1991) is the transition seam. This is required when there is a large
difference in size between adjacent fronts. In Figure 12(a), $E_F1$ and $E_F2$ are the edges on the front adjacent
to $N_k$. If the ratio of lengths between $E_F1$ and $E_F2$ is greater than 2.5, then a transition seam operation is
performed. The longer of the two edges, $E_F1$ and $E_F2$, is first split at its midpoint adding node $N_{k-1/2}$ or
$N_{k+1/2}$. In Figure 12(b), $E_F2$ is split, dividing its adjacent triangle and quadrilateral as shown. Edges $E_F$, $E_{FL}$,
and $E_{FR}$ can then be defined as front edges. With this new configuration, edge $E_F$ can be processed as a
front in state 1-1, requiring only the recovery of the top edge between $N_{k-1/2}$ and $N_{k+1}$ as in Figure 12(c).
Finally, the transition seam is completed after local smoothing and updating of the front as shown in Figure
12(d).

![Figure 12. Transition seam operation](image-url)
3.8 Transition Split

An operation useful for improving transitions is shown in Figure 13. Although similar to the transition seam operation shown in Figure 12, it is applicable when $\alpha > \varepsilon_1$ or $\alpha > \varepsilon_2$ (see equation 5). The transition split operation is performed when the ratio of lengths between $E_{F1}$ and $E_{F2}$ is greater than 2.5. Let $Q_1$ be the quadrilateral adjacent to the longer of $E_{F1}$ or $E_{F2}$. $Q_1$ is split into two quadrilaterals and a single triangle, as shown in Figure 13(a). Front $E_{F1}$ in Figure 13(a) is split at its midpoint, which causes its adjacent triangle to be split. An additional node is then inserted at the centroid of $Q_1$. As a result, new front edges, $E_F$, $E_{FL}$, and $E_{FR}$, can be defined as shown in Figure 13(b). Similar to the transition seam, $E_F$ can now be defined as a front in state 1-1 and processed to create a new quadrilateral. Figure 13(c) shows the configuration after smoothing and reclassification of fronts.

![Figure 13. Transition split operation](image)

3.9 Topological Cleanup and Smoothing

Once all of the front edges have been processed and an all-quadrilateral mesh is generated, it is often beneficial to perform local topological cleanup operations to decrease the number of irregular nodes. Although Q-Morph attempts to minimize the number of irregular nodes, they may, as a necessity, be introduced as a result of non-orthogonal boundaries or from element size transitions. Irregular nodes may also be introduced when the local nodal density and connectivity provided by the initial triangle mesh is insufficient to generate equilateral quadrilaterals. Many of these irregular nodes can be eliminated through local topological cleanup. Topological cleanup modifies the connectivity of the quadrilaterals through a series of single-step operations including edge swaps, face opens, face closes, and two-edge node removals/insertions (Staten,1997; Canann,1994; Kinney,1997). In addition, these single operation modifications can be combined into multi-step modifications to further decrease the number of irregular nodes. By reducing irregular nodes through topological cleanup, the mesh contours can more closely follow the contours of the boundary.

The final smoothing step involves a limited number of iterations of a constrained Laplacian smoothing algorithm. Each node is moved to the centroid of its neighbors only if an improvement in element shape metric (Lee,1994) would result. In situations where Laplacian smoothing produces poor results, an optimization based smoothing (Canann,1998) operation may be performed.
4. Example Problems

Four example problems shown in Figure 14 to Figure 18 demonstrate various features of the Q-Morph algorithm. The first example, shown in Figure 14, demonstrates the progression of the Q-Morph algorithm on a simple planar domain with two holes. Figure 14(a) shows the initial triangle mesh before Q-Morph begins. In this case an advancing front triangle mesher (Canann, 1997) was used to create the triangles. The method used for triangulation is unimportant, inasmuch as the appropriate nodal density is provided. Figure 14(b)-(g) show the progression of the algorithm as each successive layer of elements is completed. Figure 14(c) shows an additional layer of small elements meshed on the internal circle loop before meshing the larger elements of the outer loop. To improve element transitions, provision is made in Q-Morph to mesh loops with smaller elements before those with larger elements. The mesh is completed in Figure 14(h) after a final pass of cleanup and smoothing.

![Figure 14. Progression of Q-Morph](image)

Figure 15 and Figure 16 compares Q-Morph against Lee’s (1994) quad meshing algorithm, which uses an indirect method, coupled with an advancing front scheme to combine triangles into quadrilaterals. The toroidal surface of Figure 15 is composed of four surface patches represented as rational B-Splines. Q-Morph utilizes projection and geometric evaluation routines as part of the local and final smoothing procedures to maintain nodal locations on the three-dimensional surface. Both Figure 15(a) and (b) were generated using the same initial triangle mesh as well as the same cleanup and smoothing procedures. Despite using an advancing front scheme, Lee’s algorithm shown in Figure 15(b), has difficulty maintaining well-aligned rows of elements introducing many irregular internal nodes. Figure 16 further illustrates the
ability of the Q-Morph algorithm to generate well-aligned rows of elements parallel to a complex domain boundary, while still maintaining the required element size transitions.

Figure 15. Results of Q-Morph compared with Lee’s (1994) advancing front indirect method on toroidal surface

Figure 16. Comparison of Q-Morph with Lee’s Algorithm illustrating element boundary alignment
Figure 17 demonstrates the use of Q-Morph with a planar surface requiring a high degree of transition. Figure 17(a) shows the partially completed quad mesh with two layers of quads placed. Figure 17(b) shows the same area after final cleanup and smoothing. In order to maintain a specified nodal density near the top of the area, a sizing function (Owen, 1997) was used during the triangle meshing process. The algorithm’s ability to maintain the desired mesh density while still enforcing well-aligned rows of elements transitioning quickly to larger size elements is demonstrated in this example.

(a) Partially completed quad mesh

(b) Mesh after cleanup and smoothing

Figure 17. Large transition mesh for CFD application

The final example in Figure 18 is an industrial application of the Q-Morph algorithm. For this example, the model consisting of 104 separate areas was first constructed using a commercial CAD software application. Surfaces are once again represented by rational B-splines. In practice, the Q-Morph algorithm is used as part of a set of meshing tools which also include mapping methods. In this example, the narrow fillet regions are better represented with a mapped meshing technique, which can more appropriately create elements of high aspect ratio. Q-Morph is better suited to generating near-equilateral, isotropic quadrilaterals. Selection of the appropriate quad meshing method can be done automatically based on the number of lines comprising the area and its aspect ratio. After assigning line divisions, each area is first meshed with triangles and then transformed into quadrilaterals. Note that where an odd number of divisions is assigned to an area, Q-Morph forms a single triangle in the mesh, generally towards the interior of the area.
5. Performance

Both speed and element quality of the resulting elements from Q-Morph was evaluated as part of this study. Table 1 shows performance results from two of the example problems above. For the models in Figure 15 and Figure 17, various element densities were specified and their results noted.

5.1 Speed

Table 1 shows CPU times for both the quad-conversion and the clean-up and smoothing portions of the Q-Morph algorithm. Tests were performed on a 195 MHz SGI UNIX workstation. For the toroidal surface in Figure 15, times are necessarily affected by the number of geometric evaluations required. Times range from 141 to 242 quads converted per CPU second. This is in contrast to the flat surface of Figure 17, where times ranged from 313 to 369 quads converted per CPU second. Clean-up and smoothing times were however slower for Figure 17 than for Figure 15 as the transition in element size defined by the quad conversion required additional iterations to converge. A wide variety of factors can affect the overall speed of the algorithm. Table 1 illustrates two cases where geometry and element transition is critical.
5.2 Element Quality

Element quality was measured by shape metric, $\beta$ similar to that described by Lo (1989) and Canann (1998). For this implementation, $\beta$ is defined as the minimum triangle shape metric, $\alpha$, defined by any of the four possible triangles formed by the vertices of the quadrilateral. A $\beta$ value of 1.0 represents a perfect square, while a value of 0.0 represents a quadrilateral with a single corner angle of $\pi$. Concave or inverted quadrilaterals may be represented by negative values of $\beta$.

Both minimum and average metrics immediately following quad conversion and after clean-up and smoothing are shown in Table 1. In some cases, inverted or poorly shaped quadrilaterals can be created during the quad conversion as indicated by the negative or zero metrics. Average metrics are however very high. In all cases tested, clean-up and smoothing improved the poorly shaped quads to well within usable limits. Table 1 also shows cases where a single triangle is created in the mesh. This occurs automatically in order to resolve situations where an odd number of boundary intervals are specified.

5.3 Robustness

A diversity of surfaces has been meshed using the Q-Morph algorithm and is currently part of a commercial FEA software release (Ansys, 1998). As such, it has been successfully ported to a wide variety of platforms, including Windows, NT and UNIX environments. In general, the Q-Morph algorithm is most beneficial on surfaces where the geometric feature sizes are larger than the specified element size. In addition, high quality quadrilaterals can be expected, provided the background triangle mesh captures the details of the surface and the background triangles are of reasonable quality (i.e. $\alpha > 0.1$). In most cases where these conditions are not met, Q-Morph will be successful, however element quality may suffer.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 15</td>
<td>351</td>
<td>0</td>
<td>0.371</td>
<td>0.893</td>
<td>1.45</td>
<td>350</td>
<td>0</td>
<td>0.515</td>
<td>0.905</td>
<td>0.44</td>
</tr>
<tr>
<td></td>
<td>1208</td>
<td>0</td>
<td>0.391</td>
<td>0.905</td>
<td>5.31</td>
<td>1206</td>
<td>0</td>
<td>0.529</td>
<td>0.925</td>
<td>1.89</td>
</tr>
<tr>
<td></td>
<td>4870</td>
<td>0</td>
<td>0.170</td>
<td>0.936</td>
<td>25.4</td>
<td>4845</td>
<td>0</td>
<td>0.376</td>
<td>0.948</td>
<td>10.9</td>
</tr>
<tr>
<td></td>
<td>19209</td>
<td>0</td>
<td>-0.155</td>
<td>0.940</td>
<td>136</td>
<td>19070</td>
<td>0</td>
<td>0.359</td>
<td>0.949</td>
<td>34.2</td>
</tr>
<tr>
<td>Figure 17</td>
<td>727</td>
<td>1</td>
<td>0.00</td>
<td>0.740</td>
<td>2.32</td>
<td>696</td>
<td>1</td>
<td>0.255</td>
<td>0.802</td>
<td>2.23</td>
</tr>
<tr>
<td></td>
<td>1892</td>
<td>0</td>
<td>0.00</td>
<td>0.790</td>
<td>5.19</td>
<td>1785</td>
<td>0</td>
<td>0.344</td>
<td>0.859</td>
<td>4.36</td>
</tr>
<tr>
<td></td>
<td>4472</td>
<td>1</td>
<td>0.00</td>
<td>0.811</td>
<td>12.1</td>
<td>4288</td>
<td>1</td>
<td>0.370</td>
<td>0.889</td>
<td>15.7</td>
</tr>
<tr>
<td></td>
<td>10581</td>
<td>0</td>
<td>-0.155</td>
<td>0.817</td>
<td>33.4</td>
<td>10231</td>
<td>0</td>
<td>0.382</td>
<td>0.889</td>
<td>31.4</td>
</tr>
</tbody>
</table>

Table 1. Performance results from Q-Morph
6. Future Work

The Q-morph algorithm has been implemented and is currently part of a recent commercial release of ANSYS (1998). Although significant improvements have been made to the quad meshing capabilities of the ANSYS meshing tools through the addition of Q-morph, the main research objective was to define a prototype for the more difficult problem of hexahedral meshing. Work is currently under way to extend the principles introduced by Q-morph, to a general-purpose hex-dominant (H-morph) mesher.

7. Conclusion

The Q-Morph algorithm is an indirect quadrilateral meshing algorithm that utilizes an advancing front approach to transform triangles into quadrilaterals. It generates an all-quadrilateral mesh, provided the number of intervals on the boundary is even. The resulting mesh has few irregular internal nodes and produces elements whose contours, in general, follow the boundary of the domain. Overall element quality is excellent. The Q-Morph algorithm borrows many of its techniques from the paving method (Blacker, 1991; Cass, 1996) but adapts them for use as an indirect method, operating on an existing set of triangles. In so doing, it is able to improve upon the paving technique by resolving some of its inherent difficulties. The intersection problem, common to most direct methods of advancing front meshing, is eliminated by relying on the topology of the initial triangle mesh to close opposing fronts. Improvements also include facility for handling individual element placement through the use of states for classifying front edges. Facility for handling transition in element sizes has also been addressed through the use of sizing information provided by the initial triangle mesh and the definition of specific transformations that enable improved mesh transitions. Additionally, the initial triangle mesh provides information that reduces the cost of direct evaluations on three dimensional surface geometry.

References

ANSYS 5.5 (1998) Software program, ©Ansys, Inc., Canonsburg, PA USA


Jones, N. L. (1990), “Solid Modelling of Earth Masses for Applications in Geotechnical Engineering”, *Doctoral Dissertation*, University of Texas at Austin.


