AN ALGORITHM BASED ON FINITE ELEMENT WEIGHTS FOR WARPING TETRAHEDRAL MESHES\textsuperscript{∗§}

SUZANNE M. SHONTZ\textsuperscript{†} AND STEPHEN A. VAVASIS\textsuperscript{‡}

Abstract. We consider an algorithm called FEMWARP for warping tetrahedral finite element meshes that computes the warping using the finite element method itself. The algorithm takes as input a two- or three-dimensional domain defined by a boundary mesh (segments in one dimension or triangles in two dimensions) that has a volume mesh (triangles in two dimensions or tetrahedra in three dimensions) in its interior. It also takes as input a prescribed movement of the boundary mesh. It computes as output updated positions of the nodes of the volume mesh. The first step of the algorithm is to determine from the initial mesh a set of local weights for each interior node that describes each interior node in terms of the positions of its neighbors. These weights are computed using a finite element stiffness matrix. After a boundary transformation is applied, a linear system of equations based upon the weights is solved to determine the final positions of the interior nodes.

Our main concern is when this algorithm reverses elements. We analyze the causes for this undesirable behavior and propose techniques to make the method more robust against reversals. Among the methods includes combining FEMWARP with an optimization-based untangler.

Finally, we use FEMWARP to study the movement of the canine heart.

Key words. moving meshes, adaptation, finite element method, optimization-based mesh untangling, tetrahedral meshes, unstructured mesh generation, cardiology

AMS subject classifications. 65N50, 65N30, 92C10

1. Introduction. Moving meshes arise in cardiology, computer graphics, animation, and crash simulation, among other applications in science and engineering. With moving meshes, the mesh is updated at each step in time due to a moving domain boundary, thus resulting in potentially drastically varying mesh quality from step to step. One problem that can occur at each timestep is element reversal. “Reversal” means that the element changes orientation. In two dimensions, this means that the nodes of an element are clockwise when they ought to be counterclockwise, and in three dimensions it means that they violate the right-hand rule. We focus on mesh-warping procedures that avoid reversals as much as possible.

It is well-known that poor quality elements affect the stability, convergence, and accuracy of finite element and other solvers because they result in poorly-conditioned stiffness matrices and poor solution approximation \[27\]. If well-shaped elements are not the result of updating the mesh boundary, the mesh quality must be improved by topological or geometrical means after each time step.

Research has shown that mesh smoothing (or r-refinement) methods can be applied to improve the quality of a mesh. These methods adjust the positions of the vertices in the mesh while preserving its topology.

Laplacian smoothing is the most popular method for node-based mesh smoothing. In an iterative manner, it repositions the vertices of the mesh by moving each interior node to the geometric center of its neighbors. It is often used because it is computationally inexpensive and is very easy to implement. However, the method has several undesirable properties. One of them is that the method is not guaranteed to work, i.e., sometimes it reverses mesh elements. A second drawback is that if the algorithm is not run to convergence, the resulting mesh depends upon the order in which the nodes are smoothed.

\textsuperscript{∗}The majority of this work was performed while the first author was a member of the Center for Applied Mathematics at Cornell University. The work of the first author was supported by the National Physical Science Consortium, Sandia National Laboratories, and Cornell University. The work of both authors was supported in part by NSF grant ACI-0085969.

\textsuperscript{†}Department of Computer Science, University of Minnesota, Minneapolis, MN 55455, shontz@cs.umn.edu.

\textsuperscript{‡}Department of Computer Science, Cornell University, Ithaca, NY 14853, vavasis@cs.cornell.edu.

\textsuperscript{§}A preliminary version of these results appeared in shortened form in the Proceedings of the 2003 International Meshing Roundtable.
A related type of smoothing, namely Winslow smoothing, is more resistant to mesh folding due to the requirement that the logical variables be harmonic functions. This method was introduced for structured meshes by Winslow in \cite{32}. In \cite{19}, Knupp extends this idea to unstructured quadrilateral meshes; he used a finite-difference method for his derivative computations. Others have used the finite element method for the derivative computations on unstructured meshes; for example, Knupp cites a proprietary DOE paper by Tipton to which the authors do not have access. (See the references at the end of \cite{19}.)

Many others have created extensions to Winslow’s method. One type of extension has been to generate the meshes using a variational approach; for example, see Brackbill and Saltzman \cite{4}, Russell and co-workers \cite{6} and \cite{17}, and Thompson et al. \cite{29}. Dvinsky \cite{8} uses the theory of harmonic functions to generate a harmonic map between the physical domain and the logical domain in order to generate the mesh. Meshes that have been generated according to this method possess regularity and smoothness properties. Li et al. \cite{22} also introduced a moving mesh method based on harmonic maps; their method has two parts: a solution algorithm and a mesh-redistribution algorithm. In doing so, they are able to obtain the desirable properties of both the h-method and r-method for finite elements.

Other, more accurate methods for r-refinement are possible. Most of these methods are based upon optimization. Optimization-based methods are used with the goal of guaranteeing an improvement in the mesh quality by minimizing a particular mesh quality metric. Their main drawback, however, is their computational expense. Examples of optimization-based methods for r-refinement can be found in the following papers: \cite{12}, \cite{11}, \cite{10}, \cite{13}, \cite{9}, \cite{14}, \cite{11}, \cite{2}, \cite{25}, \cite{5}, \cite{31}, and \cite{33}. For a theory of algebraic mesh quality metrics see \cite{20}.

To address the above issues, Baker \cite{3} developed a three-step method for the metamorphosis of tetrahedral meshes. Each cycle of the method involves a combination of r-refinement, mesh coarsening, and mesh enrichment to adapt the mesh. The first step in the cycle is to move the interior nodes as far as possible using r-refinement while avoiding element reversal. The second step is to remove the poorly-shaped elements in the mesh using mesh coarsening. The final step is the addition of elements to improve the mesh quality by mesh refinement. This dynamic procedure was shown to be more cost-effective than just r-refinement. One disadvantage of this technique is that it comes with no theoretical guarantees as it is difficult to analyze because each cycle is a combination of three very different techniques.

We study a different mesh warping problem where the connectivity of the mesh is not allowed to change, which is important for some applications. The problem that we address is as follows: Given a three-dimensional domain, bounded by a triangulated surface mesh, and given an interior volume mesh composed of unstructured tetrahedra, suppose the triangulated surface mesh is displaced. Is there an algorithm to move the nodes of the volume mesh so that it continues to conform to the surface mesh and to be a good quality mesh? In addition, we study the analogous problem in two dimensions. Seeking simplicity, preservation of the mesh’s combinatorial structure, theoretical insight, and low computational expense, we propose a technique called FEMWARP and several variants to be described below.

The FEMWARP algorithm has already been considered in the previous literature, e.g., by Baker \cite{3}, although he rejected it in favor of a method based on linear elasticity. It is shown, however, in \cite{28} that there appear to be few advantages of linear elasticity over FEMWARP; whereas, a significant disadvantage is that the linear elasticity matrix problem is three times larger (in 3D) than FEMWARP’s (although FEMWARP must solve the smaller linear system three times).

FEMWARP is described in Section \ref{sec:FEMWARP} and a generalization of it called the “linear weighted Laplacian smoothing” (LWLS) framework is considered in Section \ref{sec:LWLS}. There are two main advantages to methods within the LWLS framework. First, if a continuous deformation of the boundary is given, then methods within the framework are valid for computing the resulting trajectory that specifies the movement of the interior nodes. In addition, these trajectories will be continuous. This is vital for some applications where continuity of motion is required. A second big advantage is that sparse
matrix algorithms may be used to solve (2.2). The sparsity structure is apparent, since, on average, an interior node has six neighbors in 2D, whereas a typical 2D mesh may have thousands of nodes. In Section 4, we explain the relationship between traditional Laplacian smoothing and LWLS methods.

The principal failure mode for FEMWARP is that it can reverse elements. The causes of element reversal are covered in Section 5. Techniques to prevent some reversals, including small-step FEMWARP and mesh refinement, are also covered in that section. Another technique to avoid reversals based on the Opt-MS mesh untangler is covered in Section 6. In Section 7, we test our algorithms on several types of deformations of three-dimensional meshes. In Section 8, we apply FEMWARP to study the motion of the beating canine heart under normal conditions.

2. The FEMWARP algorithm. The first step of the FEMWARP algorithm is to express the coordinates of each interior node of the initial mesh as a linear combination of its neighbors. Let a triangular or tetrahedral mesh be given for the domain Ω in two or three dimensions. Let \( b \) and \( m \) represent the numbers of boundary and interior nodes, respectively. Form the \((m + b) \times (m + b)\) stiffness matrix \( A \) based on piecewise linear finite elements defined on the initial mesh for the boundary value problem

\[
\Delta u = 0 \quad \text{on } \Omega.
\]

It is well-known [18] that this matrix is determined as follows. Let \( \phi_i \) be the continuous piecewise linear function (where the pieces of linearity are given by the triangulation) such that \( \phi_i(x_i) = 1 \), where \( x_i \) is the \( i^{th} \) node of the mesh, and \( \phi_i(x_j) = 0 \), where \( x_j \) is any other node in the mesh \( (j \neq i) \).

Define for each \( i, j = 1, \ldots, m + b \)

\[
A(i, j) = \int_{\Omega} \nabla \phi_i \cdot \nabla \phi_j.
\]

This matrix will be sparse and symmetric positive semidefinite. Its nonzero entries correspond to pairs of neighboring nodes in the mesh.

Next, let \( A_I \) denote the \( m \times m \) submatrix of \( A \) whose rows and columns are indexed by interior nodes, and let \( A_B \) denote the \( m \times b \) submatrix of \( A \) whose rows are indexed by interior nodes and whose columns are indexed by boundary nodes. Let \( x \) be the \((m + b)\) vector consisting of \( x \)-coordinates of the nodes of the initial mesh, where we assume that the interior nodes are numbered first. Then it follows from well-known theory that \( [A_I, A_B]x = 0 \) because any linear function of the coordinates is in the null-space of the discretized Laplacian operator. For the same reason, a similar identity holds for the \( y \)- and \( z \)-coordinates. An equivalent way to write this equation is

\[
A_I x_I = -A_B x_B. \quad (2.1)
\]

If we divide each row of \([A_I, A_B]\) by the diagonal element in that row, we obtain a linear system whose diagonal entries are 1’s and whose row sums are 0’s. This means that the \([A_I, A_B]\), thus scaled, expresses each interior nodal coordinate as an affine combination of the neighboring nodal coordinates.

The formation of \( A_I \) and \( A_B \) is the first step of the FEMWARP method. Consider now the application of a user-supplied transformation to the boundary of the mesh. We denote the new positions of the boundary nodes by \([\hat{x}_B, \hat{y}_B]\) in two dimensions or \([\hat{x}_B, \hat{y}_B, \hat{z}_B]\) in three.

The final step is to solve a linear system of equations similar to (2.1) for the new coordinates of the interior nodes. We solve (2.1) for \([\hat{x}_I, \hat{y}_I]\) :

\[
A_I [\hat{x}_I, \hat{y}_I] = -A_B [\hat{x}_B, \hat{y}_B]. \quad (2.2)
\]

or the analog in three dimensions. This concludes the description of FEMWARP.
3. The LWLS framework. The FEMWARP method can be generalized as follows. One produces, using some method, a family of weights \( w_{ij} \) for each ordered pair of nodes \((i, j)\) such that \( j \) is a neighbor of \( i \) and such that \( i \) is interior. Denote by \( N(i) \) the set of neighbors of \( i \), and let the coordinates of node \( i \) be \((x_i, y_i)\) in 2D or \((x_i, y_i, z_i)\) in 3D. We want these weights to express interior nodes as affine combinations of their neighbors, which imposes the following constraints on the \( w_{ij} \)'s: For each \( i \) indexing an interior node,

\[
\sum_{j \in N(i)} w_{ij} = 1, \\
\sum_{j \in N(i)} w_{ij} x_j = x_i, \\
\sum_{j \in N(i)} w_{ij} y_j = y_i,
\]

and in three dimensions, there is a fourth equation for \( z \)-coordinates. Let \([A_I, A_B]\) be \( m \times (m + b)\) matrix with 1’s on the diagonal and \(-w_{ij}\) in position \((i, j)\) whenever \( j \in N(i) \). The above three equations may be rewritten in terms of \( A_I \) and \( A_B \) as follows:

\[
A_I e_I + A_B e_B = 0, \\
A_I x_I + A_B x_B = 0, \\
A_I y_I + A_B y_B = 0,
\]

where \( e_I \) and \( e_B \) are vectors of all 1’s whose length is equal to the number of interior and boundary nodes respectively, \( x_I \) and \( x_B \) are the vectors of \( x \)-coordinates of mesh nodes in the interior and boundary respectively, and similar for \( y_B \) and \( y_I \).

Once these weights are derived, then, as in FEMWARP, the user-supplied deformation may be applied to the boundary nodes yielding deformed boundary coordinates \((\hat{x}_B, \hat{y}_B)\) in two dimensions. Finally, the new position of the interior nodes is computed via

\[
A_I [\hat{x}_I, \hat{y}_I] = -A_B [\hat{x}_B, \hat{y}_B].
\]

We will say that any method following this framework is a linearly weighted Laplacian smoothing (LWLS) method. The precise relationship between these methods and traditional Laplacian smoothing is the subject of Section 4. The FEMWARP method was not written exactly this way since the diagonal elements in \([A_I, A_B]\) for FEMWARP are not rescaled to be equal to 1. Usually for FEMWARP it is preferable to preserve the symmetry of \( A_I \) and therefore omit the scaling step. This omission has no effect on the final answer.

One concern with the weights in FEMWARP is that, in general, they are not always nonnegative. Sometimes it is desirable to have all nonnegative weights. Geometrically, this means that the interior nodes are expressed as convex (rather than merely affine) combinations of neighbors. Nonnegative weights are useful if the weight matrix is used to interpolate properties besides nodal coordinates, and it is mandatory that no extrapolation occur (e.g., because a nodal value that strays outside the range of boundary data would violate some kind of physical constraint).

In [28], the first author experimented with other weighting schemes including a method called LBWARP in which the weights \( w_{ij} \) are obtained by solving a log barrier optimization problem for each \( i \):

\[
\text{maximize} \quad \sum_{j \in N(i)} \log w_{ij} \\
\text{subject to} \quad \sum_{j \in N(i)} w_{ij} = 1 \\
\sum_{j \in N(i)} w_{ij} x_j = x_i \\
\sum_{j \in N(i)} w_{ij} y_j = y_i.
\]
The rationale for this optimization formulation is to ensure that each \( w_{ij} \) is positive, and, more strongly, that it is bounded well away from 0 (since the logarithmic term tends to \(-\infty\) if any of the \( w_{ij} \)'s approaches zero). This problem is convex, and there is an efficient algorithm to find the unique global minimum. It can be proved \(28\) that the \( w_{ij} \)'s that solve this optimization problem have a positive lower bound that depends only on the worst-case aspect ratio of triangles or tetrahedra in the mesh.

A further important property that a method in the LWLS family ought to satisfy is that \( A_I \) should be nonsingular to ensure a unique solution for the new coordinates. For FEMWARP, the nonsingularity of \( A_I \) follows from well-known theory of finite element analysis. For LBWARP, the nonsingularity follows because \( A_I \) is an irreducible and diagonally dominant matrix with strict inequality for at least one row.

Because the mesh is connected and a positive weight is associated with each edge, we see that \( A_I \) is irreducible. Also, \( A_I \) is diagonally dominant, as its diagonal entries are 1, and its off-diagonal entries are nonpositive and sum to a number in \([-1, 0]\) in each row. Since there is at least one interior node adjacent to a boundary node, \( A_I e_I \neq 0 \), where \( e_I \) is a vector of all 1's of length \( m \). Therefore, \( A_I \) satisfies the definition of an irreducibly diagonally dominant matrix by the above argument. Thus, \( A_I \) is invertible \(30, \) Theorem 1.8).

Experiments in \(28\) indicate that weights obtained from LBWARP generally do not perform as well as FEMWARP for the mesh warping application in terms of resisting element reversal.

One useful property that holds for any method in the LWLS framework including FEMWARP and LBWARP is that the method is exact for affine transformations. Let us state this as a theorem. The theorem is stated for the two-dimensional case, and it extends in the obvious way to three dimensions.

**Theorem 3.1.** Let \( A_B \) and \( A_I \) be generated using a method from the LWLS framework, and assume \( A_I \) is nonsingular. Let \( [\hat{x}_B, \hat{y}_B] \) be the user-specified deformed coordinates of the boundary. Suppose there exists a \( 2 \times 2 \) nonsingular matrix \( L \) and \( 2 \)-vector \( v \) such that for each \( j \in B \),

\[
\begin{pmatrix}
\hat{x}_j \\
\hat{y}_j
\end{pmatrix} = L \begin{pmatrix}
x_j \\
y_j
\end{pmatrix} + v.
\]

Let \( [\hat{x}_I, \hat{y}_I] \) be the deformed interior coordinates computed by the method. Then for each \( i \in I \),

\[
\begin{pmatrix}
\hat{x}_i \\
\hat{y}_i
\end{pmatrix} = L \begin{pmatrix}
x_i \\
y_i
\end{pmatrix} + v.
\]

**Proof.** The positions of the interior nodes in the deformed mesh are given by

\[
[\hat{x}_I, \hat{y}_I] = -A_I^{-1}A_B([x_B, y_B]LT + e_Bv^T) \tag{3.1}
\]

where, as above, \( \hat{x}_I, \hat{y}_I \) are column vectors composed of the \( x \)- and \( y \)-coordinates of the interior nodes respectively and \( x_B, y_B \) are the corresponding vectors for boundary nodes, and finally \( e_B \) is vector of all 1’s of length \( b \).

In order to show that affine mappings yield exact results with any algorithm within the framework, we want to show that (3.1) is the same as:

\[
[\hat{x}_I, \hat{y}_I] = [x_I, y_I]LT + e_Iv^T. \tag{3.2}
\]

Observe that the equivalence of (3.1) and (3.2) would follow immediately from:

\[
A_I([x_I, y_I]LT + e_Iv^T) = -A_B([x_B, y_B]LT + e_Bv^T). \tag{3.3}
\]

Thus, it remains to check that (3.3) holds.
Because the weights for each interior node sum to 1, \( A_I e_I + A_B e_B = 0 \) as noted above. Hence \((A_I e_I + A_B e_B)v^T = 0\). Also, because \([x_I, y_I]\) and \([x_B, y_B]\) denote the original positions of the nodes, we know that \( A_I [x_I, y_I] + A_B [x_B, y_B] = 0\). So \((A_I [x_I, y_I] + A_B [x_B, y_B])L^T = 0\).

Putting these together, we see that
\[
(A_I [x_I, y_I] + A_B [x_B, y_B]) v_T = 0
\]

Therefore, (3.3) holds, and the lemma is proven.

4. Relationship to Laplacian smoothing. In this section we explain the fairly direct connection between Laplacian smoothing and algorithms from the LWLS framework. We start with the following theorem, which is a consequence of well-known results in the previous literature.

**Theorem 4.1.** Gauss-Seidel iteration will converge for solving the linear system
\[
A_I [\hat{x}_I, \hat{y}_I] = -A_B [\hat{x}_B, \hat{y}_B]
\]
that is produced by either the LBWARP or FEMWARP algorithm.

**Proof.** In the case of FEMWARP, \( A_I \) is symmetric and positive definite hence Gauss-Seidel is convergent [16, Theorem 10.1.2]. In the case of LBWARP, \( A_I \) is irreducibly diagonally dominant hence Gauss-Seidel is convergent [30, Theorem 3.4].

The Gauss-Seidel algorithm applied to these matrices corresponds to iteratively replacing the coordinates of each interior vertex with a weighted average of the coordinates of its neighbors. The weights come from the entries of \( A_I \) and \( A_B \). In the case of FEMWARP, some of the weights may be negative.

Ordinary laplacian smoothing, as mentioned in Section 1, corresponds to the iterative process of replacing each mesh node with the centroid of its neighbors. Thus, it can be regarded as a member of the LWLS framework in which the weight of each neighbor of node \( i \) is \( 1/|N(i)| \), where \( N(i) \) is the set of neighbors of node \( i \).

5. Element reversal and small-step FEMWARP. Sometimes FEMWARP fails to yield a valid triangulation because it reverses elements. The purpose of this section is to explore the causes for reversal and propose some workarounds. The discussion of workarounds continues into the next section.

Let \( \Omega \) be the original polygonal or polyhedral domain, and let \( \hat{\Omega} \) be the domain whose boundary is given by the user-specified deformation of the boundary nodes of \( \Omega \), i.e., by the nodes at coordinates \((\hat{x}_B, \hat{y}_B)\). Assume that this user-specified deformation is not self-intersecting and preserves orientation.

Let \( \phi \) be the mapping from \( \Omega \) to \( \hat{\Omega} \) computed by FEMWARP. In other words, interpolate the interior nodal deformations linearly over the elements to arrive at a continuous function on the whole domain. (In the case that FEMWARP fails, parts of \( \phi(\Omega) \) may protrude outside of \( \hat{\Omega} \).)

Let \( \phi^* \) be the mapping that is obtained from the exact (continuum) Laplacian. In other words, solve the boundary value problems
\[
\Delta \hat{x} = 0 \quad \text{on} \ \Omega, \\
\Delta \hat{y} = 0 \quad \text{on} \ \Omega, \\
\hat{x} = \hat{x}_B \quad \text{on} \ \partial \Omega, \\
\hat{y} = \hat{y}_B \quad \text{on} \ \partial \Omega
\]
and define \( \phi^*(x, y) = (\hat{x}(x, y), \hat{y}(x, y)) \). Let us call this warping algorithm “continuum FEMWARP”.

Finally, let \( \phi^+ \) be the mapping that is obtained by linear interpolation over the elements of \( \phi^* \) evaluated at nodes. Thus, \( \phi^+ \) is intermediate between \( \phi \) and \( \phi^* \) in the sense that for \( \phi^+ \) we use the exact solution to the continuum problem only at nodal points and use interpolation elsewhere.

There are three possible reasons that FEMWARP could fail:
An Algorithm for Warping Tetrahedral Meshes

Fig. 5.1. All meshes in this section were generated by the Triangle mesh generator; this is an example of a mesh used herein.

1. Mapping $\phi^*$ might have reversals.
2. Mapping $\phi^+$ might have reversals even though $\phi^*$ has none.
3. Mapping $\phi$ might have reversals even though $\phi^+$ has none.

Let us call these Type 1, Type 2, and Type 3 failures. For Type 1 failure, “reversal” means the existence of a point $x \in \Omega$ such that $\nabla \phi^*(x)$ has a nonpositive determinant. For the second and third types, “reversal” means that a triangle is reversed. Type 1 reversals are caused by the boundary deformation alone and are not related to the mesh. Type 2 reversals are due to continuous versus discrete representation of $\phi^*$, and Type 3 reversals can be analyzed using traditional error estimates for the finite element method.

Let us start with Type 1 reversals. It is difficult to characterize inputs for which $\phi^*$ will have reversals. In [28], a sufficient condition is given for two-dimensional domains to ensure that $\phi^*$ will be an invertible function, but the condition is unrealistically stringent and is nontrivial to check in practice.

Rather than presenting the theorem from [28], we choose to present a series of examples and discussion on how to avoid reversals. The geometry for most of the examples in this section is a two-dimensional annulus with outer radius 1 and inner radius $r < 1$. Meshe

where $A = a + b/(x^2 + y^2)$ and $B = c + d/(x^2 + y^2)$, and $a, b, c, d$ are constants determined by the boundary conditions. In particular, to match the boundary conditions just described, one must satisfy the equations $a + b = \cos \theta$, $c + d = \sin \theta$, $a + b/r^2 = s/r$, $c + d/r^2 = 0$. These equations are uniquely solved by choosing

$$\{a, b, c, d\} = \{\cos(\theta) - rs, rs - r^2 \cos(\theta), \sin(\theta), -r^2 \sin(\theta)\} / (1 - r^2).$$

(5.1)

This function $\phi^*$ is invertible, i.e., avoid reversals, provided the determinant of its Jacobian is always positive. This determinant may be computed in closed form: $\det(\nabla \phi^*) = a^2 + c^2 - (b^2 + d^2)/(x^2 + y^2)^2$. This quantity is minimized when $x^2 + y^2 = r^2$. Therefore, reversals of Type 1 occur if and only if $r^2(a^2 + c^2) \leq b^2 + d^2$. Substituting the above formulas for $a, b, c, d$ yields the result
S. Shontz and S. Vavasis

that reversals occur if and only if

\[ 2r \cos(\theta) - r^2 s - s < 0. \]

For example, if \( r = s = 0.5 \) (no compression), then reversals occur when \( \cos(\theta) < 0.625 \), i.e., \( \theta > 51.4\ldots^\circ \). If \( r = 0.5 \) while \( s = 0.75 \), then reversals occur when \( \theta > 20.4\ldots^\circ \).

We tested the FEMWARP algorithm on the cases described above with a mesh for the annular region as discussed earlier. We used a mesh with inner radius \( r = 0.5 \). This particular mesh contained 10,950 triangles with maximum side length of 0.039. For \( s = 0.5 \), when we selected \( \theta = 51^\circ \), FEMWARP ran on this mesh without reversals, whereas \( \theta = 52^\circ \) caused reversals. As mentioned in the previous paragraph, \( \theta \approx 51.4^\circ \) is the cutoff for Type 1 reversals. When \( r = 0.5 \), \( s = 0.75 \), FEMWARP succeeded for \( \theta = 22^\circ \) but failed for \( \theta = 23^\circ \), again, very close to the cutoff for Type 1 reversals.

The point of the experiments in the last paragraph is that for a reasonably refined and reasonably high-quality mesh, most FEMWARP reversals seem to be Type 1 reversals. In other words, FEMWARP fails when continuum FEMWARP fails or is close to failure. Other experiments not reported here seem to confirm this point. Therefore, in order to extend the range of deformations that can be handled by FEMWARP, the best strategy is to come up with a way to avoid Type 1 reversals.

One simple method to avoid Type 1 reversals is to take several smaller steps instead of one big step. For example, suppose \((\hat{x}_B', \hat{y}_B')\) are positions for the boundary nodes intermediate between their initial positions and their final positions \((\hat{x}_B, \hat{y}_B)\). Then one could define a two-step continuum FEMWARP as follows. Solve

\[
\Delta \hat{x}' = 0 \quad \text{on } \Omega,
\]

\[
\Delta \hat{y}' = 0 \quad \text{on } \Omega,
\]

\[
\hat{x}' = \hat{x}'_B \quad \text{on } \partial \Omega,
\]

\[
\hat{y}' = \hat{y}'_B \quad \text{on } \partial \Omega
\]

for \( \hat{x}' \) and \( \hat{y}' \) to determine a mapping \( \phi_1 : \Omega \to \Omega' \) given by \((x, y) \mapsto (\hat{x}', \hat{y}') \) (where \( \Omega' \) is the domain bounded by \((\hat{x}_B', \hat{y}_B')\)) followed by

\[
\Delta \hat{x} = 0 \quad \text{on } \Omega',
\]

\[
\Delta \hat{y} = 0 \quad \text{on } \Omega',
\]

\[
\hat{x} = \hat{x}_B \quad \text{on } \partial \Omega',
\]

\[
\hat{y} = \hat{y}_B \quad \text{on } \partial \Omega'
\]

for \( \hat{x}, \hat{y} \) to obtain a map \( \phi_2 \). Finally, \( \phi^* = \phi_2 \circ \phi_1 \).

The idea in the previous paragraph can be extended to more steps with smaller increments. The limiting case of an infinite number of infinitesimal steps yields an algorithm that we call “infinitesimal-step continuum FEMWARP”. This algorithm corresponds to solving a time-dependent system of partial differential equations, but it is difficult even to write down the system that describes this limit because it requires notation for inverting systems of Laplacians. We suspect that infinitesimal-step continuum FEMWARP will never suffer from reversals as long as the boundary does not get tangled.

In the case of the annulus, it is possible again to write down infinitesimal-step continuum FEMWARP in closed form. For simplicity, let us assume \( r = s \) so that the only deformation is the rotation of the outer boundary. Assume this rotation is broken up into infinitesimally small rotations. (Another choice would be to connect the initial positions to the final positions with line segments, and break up the boundary motion as infinitesimal increments along the line segments. This way to obtain a continuous boundary motion is undesirable, however, because for a sufficiently
large rotation, the line segments would cut through the inner boundary of the annulus and hence cause tangling of the boundaries.)

With the setup described in the last paragraph, the deformation for an outer rotation of \( \theta \) computed by infinitesimal-step continuum FEMWARP maps a point at initial position \( \rho(\cos \phi, \sin \phi) \) \((r \leq \rho \leq 1)\) to \( \rho(\cos(\phi+\alpha), \sin(\phi+\alpha)) \), where \( \alpha = (1-r^2/\rho^2)\theta/(1-r^2) \). This map is clearly bijective for any value of \( \theta \); it corresponds to rotating each concentric circle of the annulus by an amount that interpolates between 0 (when \( \rho = r \)) and \( \theta \) (when \( \rho = 1 \)).

Thus, by using small-step FEMWARP with sufficiently small steps, we can essentially eliminate Type 1 failures. Small-step FEMWARP preserves the attractive property of FEMWARP that it is exact for affine maps, as long as all the intermediate steps are also affine. Unfortunately, it loses the attractive property that only one coefficient matrix for solving the linear system needs to be factored. Small-step FEMWARP requires the solution of a different coefficient matrix for each step. This drawback is partly ameliorated by the fact that even though the matrices are different, they are factored. Small-step FEMWARP preserves the attractive property of FEMWARP that it is Type 1 failures. Small-step FEMWARP requires the solution of a different coefficient matrix for each step. This drawback is partly ameliorated by the fact that even though the matrices are different, they have the same nonzero pattern, and hence the symbolic phase of sparse direct solution may be reused. If instead an iterative method is being used to solve the mesh warping equations, then the sparsity pattern may be reused in the preconditioner. In addition, the factored coefficient matrix at step \( t_k \) can be used as a preconditioner for an iterative method at step \( t_{k+1} \).

Elimination of Type 1 failures means that the mapping function \( \phi^+ \) has no reversals in the sense that the determinant of its Jacobian is positive everywhere; equivalently, it does not reverse any infinitesimally small triangles. A Type 2 failure occurs because the triangles in the mesh have finite \((\text{non-infinitesimal})\) size and hence can still be reversed by \( \phi^+ \). The following theorem characterizes when this can happen.

**Theorem 5.1.** Suppose that \( f : \Omega \to \mathbb{R}^2 \) is bijective, orientation-preserving and \( C^2 \) on \( \Omega \) with \( \nabla f \) nonsingular. Let \( T \) be a triangle in the mesh with vertices \( \{v_1, v_2, v_3\} \), and let \( T' \) be the triangle whose vertices are \( \{f(v_1), f(v_2), f(v_3)\} \). If \( [5.3] \) below holds, then \( T' \) is not reversed.

**Proof.** Recall that triangle \( T \) with vertices \( \{v_1, v_2, v_3\} \) is positively oriented if and only if \( \det(A) > 0 \), where

\[
A = (v_2 - v_1, v_3 - v_1).
\]

In order to analyze the analogous quantity for \( \{f(v_1), f(v_2), f(v_3)\} \), we start with the following algebra, which invokes the fundamental theorem of calculus twice:

\[
f(v_2) - f(v_1) = \int_0^1 \nabla f((1-t)v_1 + tv_2)(v_2 - v_1) \, dt
\]

\[
= \left( \int_0^1 \nabla f((1-t)v_1 + tv_2) \, dt \right) (v_2 - v_1)
\]

\[
= \left( \nabla f(v_1) + \int_0^1 [\nabla f((1-t)v_1 + tv_2) - \nabla f(v_1)] \, dt \right) (v_2 - v_1)
\]

\[
= \left( \nabla f(v_1) + \int_0^1 \left[ \int_0^1 \nabla^2 f((1-s)v_1 + s((1-t)v_1 + tv_2))(v_2 - v_1) \, ds \right] \, dt \right) (v_2 - v_1)
\]

\[
= \nabla f(v_1)(v_2 - v_1) + e_1
\]

where \( \|e_1\| \leq M h^2 \), where \( h \) is the maximum side length of \( T \) (an upper bound on \( \|v_2 - v_1\| \)) and \( M \) is an upper bound on \( \|\nabla^2 f\| \) in the triangle. Similarly,

\[
f(v_3) - f(v_1) = \nabla f(v_1)(v_3 - v_1) + e_2,
\]

where again \( \|e_2\| \leq M h^2 \). Therefore,

\[
(f(v_2) - f(v_1), f(v_3) - f(v_1)) = \nabla f(v_1)A + E \quad (5.2)
\]
where \( A \) is as above and \( \| E \|_2 \leq \sqrt{2} M h^2 \). Observe that \( \nabla f(v) A \) has positive determinant by assumption. Therefore, the left-hand side can have negative determinant only if \( E \) is a sufficiently large perturbation to change the determinant sign. If \( E \) is such a large perturbation, then by the continuity of the determinant, there is a perturbation \( E' \) no larger than \( E \) such that the det(\( \nabla f(v) A + E' \)) = 0, i.e., \( \nabla f(v) A + E' \) is singular. Furthermore, \( \| E' \| \leq \sqrt{2} M h^2 \). By Theorem 2.5.3 of [16], this means that

\[
\sqrt{2} M h^2 \geq \sigma_{\min}(\nabla f(v_1) A) \geq \sigma_{\min}(\nabla f(v_1)) \sigma_{\min}(A),
\]

where \( \sigma_{\min}(A) \) and \( \sigma_{\max}(A) \) denote the smallest and largest singular values of \( A \), respectively. It follows from the equation \( AA^{-1} = I \) that the columns of \( A^{-1} \) are parallel to the altitude segments of triangle \( T \) perpendicular to \( v_1 v_3 \) and \( v_1 v_2 \) respectively, but scaled so that their lengths are the reciprocals of those altitude lengths. Therefore, \( \sigma_{\max}(A^{-1}) \leq \sqrt{2}/\text{minalt}(T) \), where \( \text{minalt}(T) \) means the minimum altitude. Thus, \( \sigma_{\min}(A) \geq \text{minalt}(T)/\sqrt{2} \). Substituting this inequality into (5.3) and rearranging yields

\[
\frac{\sigma_{\min}(\nabla f(v_1))}{M} \leq 2h \text{ asp}(T)
\]

where \( \text{asp}(T) \), the aspect ratio of \( T \), equals \( h/\text{minalt}(T) \). The aspect ratio is often used as a shape-quality metric; lower values mean a better shaped triangle. Thus, reversal cannot happen if the opposite inequality holds:

\[
\frac{\sigma_{\min}(\nabla f(v_1))}{M} > 2h \text{ asp}(T).
\]

The point of this theorem is that Type 2 reversals cannot occur for a sufficiently refined mesh (i.e., \( h \) sufficiently small in (5.4)), assuming that the mesh quality does not decay, and assuming that \( \phi^* \) is a nonsingular function. (Assuming Type 1 reversals are excluded, function \( \phi^* \) is never singular on the interior because Laplace solutions are analytic. It could be singular at the boundary if, for example, \( \Omega' \) has a corner where \( \Omega \) had none.)

We tested this theorem for two examples, each of which diverges a bit from the theoretical prediction. For the first example, we generated a uniform mesh for the rectangle \([0, 2] \times [0, 1]\) using Triangle and mapped all the nodes using the function \( f(x, y) = (x, y + \alpha x(2 - x)) \). For each mesh, \( \alpha \) was incremented by 1 until reversal occurred. (No Laplace solution was involved in this test case.) We tabulated the values of \( h \) versus \( \alpha \) in Table [5]. As predicted by the theorem, the tables shows that as \( h \) decreases, a larger value of \( \alpha \) is tolerated. Contrary to the theorem, however, the table shows that \( \sigma_{\min}(\nabla f)/\| \nabla^2 f \| \) is decreasing faster than \( h \). In other words, reversals are avoided to a greater extent than predicted by the theorem. The reason for this discrepancy is that the perturbation term \( E \) in (5.3) is not well aligned with the direction that drives \( (\nabla f) A \) toward singularity in this example. In particular, \( E \) affects only the \( y \)-components (since the transformation is linear for \( x \)-coordinates). On the other hand, transformation \( \phi \) stretches the triangles substantially in the \( y \)-direction, so that the most effective way to perturb \( (\nabla f) A \) toward singularity is a small change to the \( x \)-components.

As a second test case, consider the transformation of the annulus with radii \((0.5, 1)\) that results from continuous-warping continuum FEMWARP, that is, the transformation that rotates a point at radius \( r \) by angle \( \alpha(1 - r^2/\rho^2)/(1 - r^2) \), where \( r \) is the inner radius \( (r = 0.5 \text{ for this test}) \). For each mesh, the parameter \( \alpha \) was stepped in increments of \( \pi/16 \) until reversals were encountered. We tabulated values of \( h \) versus the first of \( \alpha \) causing failure in Table [6]. As predicted by the theorem, decreasing \( h \) corresponded to increasing values of \( \alpha \), i.e., greater distortion of the domain. Again, these results do not initially correspond to the preceding theorem quantitively: in this case, \( h \) is decreasing faster than \( \sigma_{\min}(\nabla f)/\| \nabla^2 f \| \). Only the last three rows of the table show that \( h \) and \( \sigma_{\min}(\nabla f)/\| \nabla^2 f \| \) are decreasing proportionally. The reason for this discrepancy is that \( \nabla^2 f \) is much
over the space of piecewise linear choices for $u$ whereas the difference between the two mappings is $O(\sigma)$ causes reversals in the mesh. The first column shows the mesh cell size (maximum edge length). The third column is $\sigma_{\min}(\nabla f)/\|\nabla^2 f\|$ evaluated at a vertex of a triangle that reversed.

| $h$   | $\alpha_{\text{fail}}$ | $\sigma_{\min}(\nabla f)/\|\nabla^2 f\|$ |
|-------|------------------------|----------------------------------|
| 0.205 | 10                     | $4.6 \cdot 10^{-4}$              |
| 0.108 | 14                     | $2.5 \cdot 10^{-3}$              |
| 0.057 | 30                     | $5.7 \cdot 10^{-4}$              |
| 0.030 | 51                     | $1.9 \cdot 10^{-4}$              |
| 0.015 | 95                     | $5.6 \cdot 10^{-5}$              |

Consider the following test. We generated a sequence of small-step rotations for the annulus in two ways. In the first case, we took steps that rotate the outer boundary by $\pi/16$ and leave the inner boundary invariant, each time computing the Laplace solution exactly analytically. This corresponds to iteratively applying the transformation $\phi^+(x, y) = (Ax + By, -Bx + Ay)$ to the mesh, where $A, B, C, D$ are functions of $x^2 + y^2$ as above, and $a, b, c, d$ are constants determined by $\|\nabla \phi\|$. With $r = s = 0.5, \theta = \pi/16$. In the second case, we solved Laplace’s equation for the above boundary condition using the finite element mesh that results from small-step FEMWARP. In both cases we tried meshes with several different values of $h$. The results are tabulated in Table 5.2. As can be seen, discretized small-step FEMWARP outperformed continuum small-step FEMWARP.

In other words, not only did Type 3 reversals not occur, but in fact it seems to be preferable to use the discretized solution for mesh warping rather than the continuum solution. The difference between $\phi^+$ and $\phi$ is the usual discretization error in finite element methods. A possible explanation for the improved resistance to reversals of the finite element solution is as follows. After several steps of small-step FEMWARP, Laplace’s equation is solved on a mesh with mostly poorly-shaped elements, some extremely poorly-shaped. A Laplace solution minimizes the functional $F(u) = \int_{\Omega} \nabla u \cdot \nabla u$ over $H^1$ functions $u$ on the domain, and the finite element solution minimizes the same functional $F(u)$ over the space of piecewise linear choices for $u$. A very poorly-shaped element is “stiffer” than
Continuum versus discretized small-step FEMWARP applied to a mesh of an annulus in order to test for Type 3 reversals. The table shows that discretized small-step FEMWARP seems less prone to reversals than continuum small-step FEMWARP.

| h    | \( \alpha_{\text{fail}}^{\text{continuum}} \) | \( \alpha_{\text{fail}}^{\text{discrete}} \) |
|------|---------------------------------|-----------------|
| 0.202 | 7\( \pi/16 \)                     | \( \pi/2 \) |
| 0.114 | \( \pi/2 \)                      | 9\( \pi/16 \) |
| 0.058 | 9\( \pi/16 \)                     | 5\( \pi/8 \) |
| 0.031 | 5\( \pi/8 \)                      | 7\( \pi/8 \) |
| 0.015 | 11\( \pi/16 \)                    | \( \pi \) |

Variable stepsize versus constant stepsize for small-step FEMWARP applied to a mesh of an annulus. Second and third columns are the maximum amount of rotation prior to reversals for the two methods. Fourth and fifth columns are the number of Cholesky factorizations required by the two methods.

| h    | \( \alpha_{\text{VS}}^{\text{max}} \) | \( \alpha_{\text{CS}}^{\text{max}} \) | NCHOL\text{VS} | NCHOL\text{CS} |
|------|---------------------------------|-----------------|-----------------|-----------------|
| 0.202 | 1.7426                          | 1.7671          | 13              | 72              |
| 0.114 | 2.2089                          | 2.0862          | 24              | 85              |
| 0.058 | 2.6998                          | 2.4789          | 29              | 101             |
| 0.031 | 3.4852                          | 2.7980          | 34              | 114             |

others in the following sense. An affine linear function \( u \) defined over a triangle that has an angle close to 180° will have a quite large gradient value (compared to a well-shaped triangle with the equal area and equal nodal values) unless the nodal values lie in a certain restricted range. Therefore, the extra stiffness of these elements will cause them to be deformed less than the better-shaped elements in the optimal solution that minimizes \( F \). Since the poorly-shaped elements are those most in danger of being reversed, this is a desirable effect.

The last topic to consider in this section is how to select a stepsize for small-step FEMWARP. The theory developed above indicates that as long as the step size is well below a step large enough to cause reversals of \( \phi^* \), the step size should not matter so much. In fact, we propose the following simple strategy, which seems to be effective. Attempt to take a very large step (e.g., a rotation of size \( \pi \) in the case of the annulus). If this fails (causes reversals), then halve the stepsize and try again until success. Update the mesh and try another such step. Note that in the process of searching for a correct stepsize, the coefficient matrix in FEMWARP is the same for each trial. Therefore, the Cholesky factors do not need to be recomputed until the mesh is updated.

Another way to carry out small-step FEMWARP would be to take constant (small) steps on each iteration. We compared these two methods and found that the first was much more efficient, and furthermore, reversals are resisted better by the first strategy. Therefore, the repeated halving strategy is recommended. Table 5 summarizes the result for the annulus again. For the halving strategy, updates were pursued until the stepsize dropped below \( \pi/128 \). For the constant-step strategy, the stepsize was taken to be \( \pi/128 \).

6. Mesh warping and mesh untangling. In the previous section we considered reasons why FEMWARP can fail and also some possible workarounds. The workarounds in the previous section, however, are not available in all circumstances. For example, the workaround for Type 1 reversals, namely, small-step FEMWARP, requires a homotopy from the old to new boundary conditions and also requires solution of many linear systems with distinct coefficient matrices. The workaround for Type 2 reversals requires refined meshes, which may not be available.

Another workaround is to switch to a different algorithm, for example, Opt-MS, a mesh untangling method due to Freitag and Plassmann [14]. Opt-MS takes as input an arbitrary tangled
mesh and a specification of which nodes are fixed (i.e., boundary nodes) and which are movable. It then attempts to untangle the mesh with a sequence of individual nodal moves based on linear programming. More details are provided below.

In this section, we consider the use of Opt-MS for mesh warping. We find that the best method is a hybrid of FEMWARP and Opt-MS.

To untangle the mesh, Opt-MS performs repeated sweeps over the interior nodes. For each interior node, it repositions the node at the coordinates that maximize the minimum signed area (volume) of the elements adjacent to that node (called the “local submesh”). The signed area is negative for a reversed element, so maximizing its minimum value is an attempt to fix all reversed elements.

Let \( x \) be the location of the free vertex, that is, the current interior vertex being processed in a sweep. Let \( x_1, \ldots, x_n \) be the positions of its adjacent vertices, and \( t_1, \ldots, t_n \) be the incident triangles (tetrahedra) that compose the local submesh. Then the function that prescribes the minimum area (volume) of an element in the local submesh is given by

\[
q(x) = \min_{1 \leq i \leq n} A_i(x),
\]

where \( A_i \) is the area (volume) of simplex \( t_i \). In 2D, the area of triangle \( t_i \) can be stated as a function of the Jacobian of the element as follows: \( A_i = \frac{1}{2} \det(x_i - x, x_j - x) \) which is a linear function of the free vertex position; the same is true in 3D. Freitag and Plassmann use this fact to formulate the solution to

\[
\max q(x) = \max \min_{1 \leq i \leq n} A_i(x)
\]

as a linear programming problem which they solve via the simplex method. On each sweep, \( m \) linear programs are solved which sequentially reposition each interior node in the mesh. Sweeps are performed until the mesh is untangled or a maximum number of sweeps has occurred.

A shortcoming of Opt-MS, in comparison to FEMWARP, is that it is not intended to handle a very large boundary motion even if that motion is affine. For example, starting from a 2D mesh, if the boundary vertices are all mapped according to the function \((x, y) \mapsto (-x, -y)\) while the interior nodes are left unmoved, in many cases Opt-MS is unable to converge. FEMWARP, on the other hand, will clearly succeed according to Theorem 3.1.

Therefore, we propose the following algorithm: one applies first the FEMWARP mesh warping algorithm, and if the mesh is tangled, one uses Opt-MS to untangle the mesh output by FEMWARP (as opposed to the original mesh). The rationale for this algorithm is that FEMWARP is better able to handle the gross motions while Opt-MS handles the detailed motion better.

To test whether this algorithm works, we applied it again to the mesh depicted in Fig. 5.1. The boundary motion is as follows: we rotate the outer circular boundary by \( \theta_1 \) degrees and the inner boundary by \( \theta_2 \) degrees and then test three algorithms: FEMWARP alone, Opt-MS alone, and hybrid. (The hybrid was tested only in the case that the two algorithms individually both failed.) The results are tabulated in Table 6.1. The table makes it clear that the hybrid often works when Opt-MS and FEMWARP both fail. Thus, the hybrid method is another technique for situations when FEMWARP alone fails and small-step FEMWARP combined with mesh refinement may be unavailable. The hybrid method has the disadvantage, when compared to FEMWARP, that it does not produce a continuous motion of interior nodes but rather only a final configuration.

7. Three-dimensional tests. In this section we compare the robustness against reversals of FEMWARP, small-step FEMWARP, and the hybrid FEMWARP/Opt-MS method on examples of 3D meshes [21] and [15]. We choose specific nonlinear boundary deformations parameterized by a scalar \( \alpha \) in order to determine how much deformation each test mesh could withstand when warped according to each method:

\[
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix} \mapsto \begin{pmatrix}
  2 & -1 & 0 \\
  -2 & 5 & 0 \\
  0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix} + \alpha \begin{pmatrix}
  0.1xy \\
  0.5yz \\
  0.1x^2
\end{pmatrix}.
\]

Table 7 gives the results obtained from warping various three-dimensional meshes according to this boundary deformation. The data in this table is as follows. The first three columns give the
The table indicates that small-step FEMWARP was about equal in robustness against reversals compared with FEMWARP except for the last mesh. In the case of “tire”, small-step FEMWARP was much more robust. This is probably because the reversals in the first three rows are primarily Type 2 reversals since the meshes are very coarse, whereas the “tire” mesh is finer and is therefore more likely to see Type 1 reversals according to the arguments given in the previous section. Small-step FEMWARP is intended to fix Type 1 reversals but is not effective against Type 2 reversals.

We also compared Opt-MS and the hybrid FEMWARP/Opt-MS method described in the last section. Opt-MS performed poorly because, as mentioned earlier, it is not designed to handle large boundary motions. The performance of the hybrid is comparable, and in some cases superior, to that of small-step FEMWARP.

8. Application to Cardiology. We now use FEMWARP in order to study the movement of the beating canine heart under normal conditions. To do this, we obtained data from the Laboratory of Cardiac Energetics at the National Institutes of Health (NIH) [24]. We were given \((x, y, z, t)\) data for 192 points on the inner surface of the left and right ventricles of the beating canine heart from

---

### Table 6.1

| \(\theta_1\) | \(\theta_2\) | 0 | 15 | 30 | 45 | 60 | 75 | 90 | 105 | 120 | 135 | 150 | 165 | 180 |
|------------|------------|---|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|
| 0          | B          | B | B  | B  | O  | O  | O  | -  | -   | -   | -   | -   | -   | -   |
| 15         | B          | B | B  | B  | B  | O  | O  | H  | -   | -   | -   | -   | -   | -   |
| 30         | B          | B | B  | B  | B  | B  | H  | O  | H   | -   | -   | -   | -   | -   |
| 45         | B          | B | B  | B  | B  | B  | B  | O  | O   | H   | -   | -   | -   | -   |
| 60         | O          | B | B  | B  | B  | B  | B  | B  | O   | O   | H   | -   | -   | -   |
| 75         | O          | O | B  | B  | B  | B  | B  | B  | B   | B   | H   | H   | -   | -   |
| 90         | O          | O | O  | B  | B  | B  | B  | B  | B   | O   | H   | H   | -   | -   |
| 105        | O          | O | O  | O  | B  | B  | B  | B  | B   | F   | B   | O   | H   | -   |
| 120        | -          | - | H  | H  | H  | B  | B  | F   | F   | B   | F   | H   | -   | -   |
| 135        | -          | - | -  | H  | H  | H  | F   | B   | B   | B   | B   | -   | -   | -   |
| 150        | -          | - | -  | -  | H  | H  | F   | B   | B   | B   | B   | -   | -   | -   |
| 165        | -          | - | -  | -  | -  | H  | H  | F   | F   | B   | B   | -   | -   | -   |
| 180        | -          | - | -  | -  | -  | -  | H  | H  | F   | B   | B   | -   | -   | -   |

### Table 7.1

| Mesh name | # bdry nodes | # nodes | \(\alpha_{\text{max}}^{\text{FEMWARP}}\) | \(\alpha_{\text{max}}^{\text{ssFEMWARP}}\) | NCHOLVS | \(\alpha_{\text{max}}^{\text{Opt-MS}}\) | \(\alpha_{\text{max}}^{\text{hybrid}}\) | \(\Delta\alpha\) |
|-----------|--------------|---------|----------------------------------------|----------------------------------------|---------|----------------------------------------|----------------------------------------|----------------|
| foam5     | 1048         | 1337    | 0.7                                    | 1.0                                    | 4       | 0                                      | 1.2                                    | 0.1            |
| gear      | 606          | 866     | 3.5                                    | 3.5                                    | 4       | 0.6                                    | 3.5                                    | 0.1            |
| hook      | 790          | 1190    | 0.16                                   | 0.16                                   | 2       | 0                                      | 0.16                                   | 0.01           |
| tire      | 1248         | 2570    | 0.15                                   | 1.60                                   | 13      | 0                                      | 2.0                                    | 0.05           |
An Algorithm for Warping Tetrahedral Meshes

The first step in the simulation of the ventricular movement was to generate a mesh for the initial position of the ventricles. In order to do this, we first noted that the 192 points we were given were arranged in eight slices with 24 points each. Thus, in order to generate the initial mesh, we decided to create a mesh for the top slice and then use FEMWARP to do the mesh warping necessary to create meshes for the remaining slices. Note that this uses FEMWARP in one and two dimensions as we describe in detail below. Once we have the meshes for all of the levels, we connect the triangular meshes into a tetrahedral mesh for the ventricles. The procedure to do this is also described below.

We now give a more detailed description of the method we used to create the initial mesh of the canine ventricles. We first used Triangle to generate an initial mesh of the top slice (after projecting it into the x-y plane). Note that this yielded a good-quality mesh in the x-y plane with several additional nodes. Second, we computed the z-coordinates for the new points on the boundary of the top slice using 1D FEMWARP. Third, we used the weight-finding portion of our FEMWARP algorithm to compute the weights for the appropriate 2D linear system obtained from the x- and y-coordinates. Fourth, we determined the z-coordinates for the mesh of the top slice by forcing the z-coordinates to satisfy the appropriate 3D linear system using the 2D weights. At this point, we had the mesh for the top slice.

Our second task was to generate meshes for each of the remaining seven slices. This was done using our FEMWARP algorithm to warp the mesh for the top slice into meshes for each of the remaining slices. In order to accomplish this, the first step was to determine the coordinates of the additional boundary nodes for the mesh of the appropriate slice. This was done using 1D FEMWARP. Then, the (x, y) coordinates of the interior nodes of that slice were determined using 2D FEMWARP. The z-coordinates for the interior nodes were found by forcing them to satisfy the appropriate 3D linear system using these weights.

The third step was to connect the triangular meshes for each of the eight slices into one tetrahedral mesh for the canine ventricles. To do this, the corresponding triangles between two slices were connected to form a triangular prism. After a temporary mesh of triangular prisms was created, the triangular prisms were subdivided into tetrahedron using the method outlined in [7].

After the initial tetrahedral mesh was created, we checked its quality using the inverse mean-ratio mesh-quality metric. (The mean ratio was defined in [23] and was adapted for tetrahedral elements in [12].) Using this test, it was determined that the initial mesh was of poor quality. This was because the eight slices were equally-spaced even though the curvature of the ventricles changes much more rapidly near the bottom of the heart. Thus, we decided to use linear interpolation in conjunction with FEMWARP in the obvious way in order to add two additional slices of nodes near the bottom of the heart. Because the curvature of the ventricles changes more rapidly near the bottom of the heart, the first additional slice was placed halfway between levels 7 and 8, and the second additional slice was placed halfway between the first new level and level 8. The resulting initial mesh is shown in Figure 8.1.

After the initial mesh was created, the heart data was used to move the 192 data points on the boundary of the mesh to their new positions. The same process as above (i.e., FEMWARP in 1D, FEMWARP in 2D, and using 1D FEMWARP to add the two new levels of data near the bottom) was used in order to reposition the remaining boundary points. Once the boundary nodes were relocated to their positions for timestep \( t = 2 \), the 3D version of FEMWARP was used to move the interior nodes to their new positions for this timestep. This process was performed iteratively in order to study the movement of the heart at timesteps \( t = 3, \ldots, 32 \).

The simulation of the canine ventricular movement produced a series of meshes that show the ventricles twisting, expanding, and then contracting over the cardiac cycle. This is consistent with what occurs in nature. Because the dynamic range of the motion is small, it cannot be detected in single figures (separate from an animation).
During each timestep, the inverse mean ratios of the tetrahedra in the mesh were computed. The inverse mean ratio computations showed that the heart remained untangled throughout the entire simulation. This is not very surprising since the heart mesh is composed of elliptical rings that seem to undergo less movement on each timestep than the circular rings in the test cases. However, the motion of the heart is anisotropic which makes it difficult to predict in advance how it will tolerate deformations. Interestingly enough, the values of the minimum and average inverse mean ratios were relatively constant across all timesteps. Only the value of the maximum inverse mean ratio changed a significant amount. However, it only increased by as much as four percent of its initial value and decreased by as much as seven percent of that value. Thus the inverse mean ratio computations indicate that the heart meshes are of sufficiently good quality for use with a numerical PDE solver that requires moving meshes.

In order to further test our mesh warping algorithm, the motion of the ventricles was exaggerated by a factor of 3. In this case, the motion was large enough to detect in separate figures and is shown in Figure 8.2. We note that the FEMWARP algorithm also performed successfully in this case, which is encouraging given the much larger deformations. Small-step FEMWARP was not needed for this problem.

9. Conclusions. We studied an algorithm called FEMWARP for warping triangular and tetrahedral meshes. The first step in the algorithm is to determine a set of local weights for each interior node using finite element methods. Second, a user-supplied deformation is applied to the boundary notes. The third and final step is to solve a system of linear equations based upon the weights and the new positions of the boundary nodes to determine the final positions of the interior nodes.

FEMWARP falls into a more general class of methods that we call linear weighted laplacian smoothing (LWLS). LWLS methods have several advantages. First, if a continuous boundary deformation is given, then methods within the framework are valid for computing the resulting trajectory that specifies the movement of the interior nodes. In addition, these trajectories will be continuous, which is vital for applications where continuity of motion is required. Second, sparse matrix algorithms may be used to solve the linear system which determines the final positions of the interior nodes. Third, the methods are exact if the boundary deformation is affine.

We then analyzed the case that FEMWARP fails, i.e., produces element reversals. Some workarounds include: taking smaller steps, using a finer mesh, and finally, a hybrid of FEMWARP and Opt-MS.

We tested the robustness of FEMWARP, small-step FEMWARP, and hybrid FEMWARP/Opt-MS on 2D annulus test cases and 3D general unstructured meshes. The latter two algorithms generally outperform plain FEMWARP, sometimes significantly. We also used FEMWARP to study the motion of the canine ventricles under normal conditions.
Fig. 8.2. Exaggerated Movement of the Heart as it Beats. Motion for timesteps $t = 1, 9, \text{ and } 25$. 
10. Acknowledgements. The authors wish to thank O. Faris and E. McVeigh from the Laboratory of Cardiac Energetics at the National Institutes of Health for the moving canine heart data that they provided. In addition, they wish to thank Dr. Lori Freitag-Diachin of Lawrence Livermore National Laboratory and Dr. Patrick Knupp of Sandia National Laboratories for providing us with the majority of the 3D test meshes. They benefited from conversations with G. Bailey and A. Schatz of Cornell, C. Patron of Risk Capital, T. Tomita of BRM Consulting, and especially H. Kesten of Cornell. The authors thank the two anonymous referees for their careful reading of an earlier version of this manuscript.

REFERENCES

[1] N. Amenta, M. Bern, and D. Eppstein, Optimal point placement for mesh smoothing, in Proceedings of 8th ACM-SIAM Symposium on Discrete Algorithms, 1997, pp. 528–537.
[2] E. Amezua, M. Hormaza, A. Hernández, and M. Ajuria, A method for the improvement of 3d solid finite-element meshes, Adv. Eng. Softw., 22 (1995), pp. 45–53.
[3] T. J. Baker, Mesh movement and metamorphosis, in Proceedings of the Tenth International Meshing Roundtable, Albuquerque, NM, 2001, Sandia National Laboratories, pp. 387–396.
[4] J. U. Brackbill and J. S. Saltzman, Personal Communication, (February 2003).
[5] S. A. Canann, M. B. Stephenson, and T. Blacker, Optissmoothing: An optimization-driven approach to mesh smoothing, Finite Elem. Anal. Des., 13 (1993), pp. 185–190.
[6] W. M. Cao, W. Z. Huang, and R. D. Russell, A study of monitor functions for two dimensional adaptive mesh generation, SIAM J. Sci. Comput., 20 (1999), pp. 1978–1994.
[7] J. Dompierre, P. Labbé, M. Vallet, and E. Camarero, How to subdivide pyramids, prisms and hexahedra into tetrahedra, in Proceedings of the 8th International Meshing Roundtable, Albuquerque, NM, 1999, Sandia National Laboratories, pp. 195–204.
[8] A. S. Dvinsky, Adaptive grid generation from harmonic maps on riemannian manifolds, J. Comput. Phys., 95 (1991), pp. 450–476.
[9] L. Freitag, On combining laplacian and optimization-based mesh smoothing techniques, in AMD - Vol. 220 Trends in Unstructured Mesh Generation, ASME, 1997, pp. 37–43.
[10] L. Freitag and P. Knupp, Tetrahedral element shape optimization via the jacobian determinant and condition number, in Proceedings of the 8th International Meshing Roundtable, Albuquerque, NM, 1999, Sandia National Laboratories, pp. 247–258.
[11] L. Freitag and P. Knupp, Tetrahedral mesh improvement via optimization of the element condition number, Internat. J. Numer. Methods Engrg., 53 (2002), pp. 1377–1391.
[12] L. Freitag, P. Knupp, T. Munson, and S. Shontz, A comparison of optimization software for mesh shape-quality improvement problems, in Proceedings of the Eleventh International Meshing Roundtable, Albuquerque, NM, 2002, Sandia National Laboratories, pp. 29–40.
[13] L. Freitag and C. Ollivier-Gooch, Tetrahedral mesh improvement using swapping and smoothing, Internat. J. Numer. Methods Engrg., 40 (1997), pp. 3979–4002.
[14] L. Freitag and P. Plassmann, Local optimization-based simplicial mesh untangling and improvement, Internat. J. Numer. Methods Engrg., 49 (2000), pp. 109–125.
[15] L. Freitag-Diachin, Personal Communication, (October 2002).
[16] G. H. Golub and C. F. Van Loan, Matrix Computations, John Hopkins University Press, third ed., 1996.
[17] W. Z. Huang and R. D. Russell, Moving mesh strategy based on a gradient flow equation for two-dimensional problems, SIAM J. Sci. Comput., 20 (1999), pp. 998–1015.
[18] C. Johnson, Numerical solutions of partial differential equations by the finite element method, Studentlitteratur, 1987.
[19] P. Knupp, Winslow smoothing on two-dimensional unstructured meshes, Eng. Comput., 15 (1999), pp. 263–268.
[20] P. Knupp, Tetrahedral mesh improvement using swapping and smoothing, in Proceedings of the First Workshop on Applied Computational Geometry, New York, NY, 1996, Association for Computing Machinery, pp. 124–133.
[21] L. Freitag-Diachin, Personal Communication, (October 2002).
[22] R. Li, T. Tang, and P. Zhang, Moving mesh methods in multiple dimensions based on harmonic maps, J. Comput. Phys., 170 (2001), pp. 562–588.
[23] A. Liu and B. Joe, Relationship between tetrahedron quality measures, BIT, 34 (1994), pp. 268–287.
[24] E. McVeigh and O. Faris, Personal Communication, (February 2003).
[25] V. N. Parthasarathy and S. Kodiyalam, A constrained optimization approach to finite element mesh smoothing, Finite Elem. Anal. Des., 9 (1991), pp. 309–320.
[26] J. Shewchuk, Triangle: Engineering a 2d quality mesh generator and delaunay triangulator, in Proceedings of the First Workshop on Applied Computational Geometry, New York, NY, 1996, Association for Computing Machinery, pp. 124–133.
[27] L. Freitag-Diachin, Personal Communication, (October 2002).
the Eleventh International Meshing Roundtable, Albuquerque, NM, 2002, Sandia National Laboratories, pp. 115–126.

[28] S. M. SHONTZ, Numerical Methods for Problems with Moving Meshes, PhD thesis, Cornell University, 2005.

[29] J. F. THOMPSON, Z. U. A. WARSI, AND C. W. MARTIN, Numerical Grid Generation, North-Holland, 1985.

[30] R. VARGA, Matriz Iterative Analysis, Prentice-Hall, 1963.

[31] D. WHITE AND G. RODRIGUE, Improved vector FEM solutions of Maxwell’s Equations using grid pre-
       conditioning, Internat. J. Numer. Methods Engrg., 40 (1997), pp. 3815–3837.

[32] A. WINSLOW, Numerical solution of the quasi-linear poisson equation in a nonuniform triangle mesh, J. Comput.
       Phys., 1967 (1967), pp. 149–172.

[33] P. ZAVATTIERI, E. DARI, AND G. BUSCAGLIA, Optimization strategies in unstructured mesh generation, Internat.
       J. Numer. Methods Engrg., 39 (1996), pp. 2055–2071.