INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng. **45**, 37–45 (1999)

APPLICATIONS OF MESH SMOOTHING: COPY, MORPH, AND SWEEP ON UNSTRUCTURED QUADRILATERAL MESHES

PATRICK M. KNUPP*

Parallel Computing Sciences Department, Sandia National Laboratories, M/S 0441, P.O. Box 5800, Albuquerque, NM 87185-0441, U.S.A.

SUMMARY

Mesh smoothing is demonstrated to be an effective means of copying, morphing, and sweeping unstructured quadrilateral surface meshes from a source surface to a target surface. Construction of the smoother in a particular way guarantees that the target mesh will be a 'copy' of the source mesh, provided the boundary data of the target surface is a rigid body rotation, translation, and/or uniform scaling of the original source boundary data and provided the proper boundary node correspondence between source and target has been selected. Copying is not restricted to any particular smoother, but can be based on any locally elliptic second-order operator. When the bounding loops are more general than rigid body transformations the method generates high-quality, 'morphed' meshes. Mesh sweeping, if viewed as a morphing of the source surface to a set of target surfaces, can be effectively performed via this smoothing algorithm. Published in 1999 by John Wiley & Sons, Ltd. This article is a U.S. government work and is in the public domain in the United States.

KEY WORDS: unstructured grid generation; hexahedral meshing; mesh smoothing; mesh sweeping; mesh copying

1. INTRODUCTION

Smoothing techniques are widely used on unstructured meshes to improve mesh quality (see, for example, References 1–3). It is shown here that they may also be used to perform a number of other critical tasks, namely mesh copying, morphing, and sweeping. Copying and morphing of meshes arises from the need to rapidly transfer an existing mesh that has been created on a particular 'source' surface of a complex multi-surface geometry onto one or more other 'target' surfaces of similar shape and size.[†]

The smoothing method presented here has been implemented in the CUBIT[4] hexahedral meshing code as part of the Morph[5] and Sweep[6] capabilities. Two major methods in CUBIT for generating a

* Correspondence to: Patrick M. Knupp, Parallel Computing Sciences Department, Sandia National Laboratories, M/S 0441, P.O. Box 5800, Albuquerque, NM 87185-0441, U.S.A. E-mail: pknupp@sandia.gov

[†] To clarify subsequent discussion, copy is a special case of morph. 'Copy' is used if the source and target are of identical geometric shape and 'morph' if the source and target are similar in shape

Contract/grant sponsor: Department of Energy; Contract/grant number: DE-AC04-94Al85000

Published in 1999 by John Wiley & Sons, Ltd.

Received 7 August 1998
Revised 17 August 1998

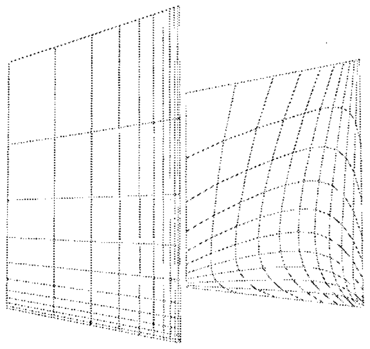

Figure 1. Original mesh copy algorithm: source (left) – target (right)

surface mesh on the source surface are 'mapping'[7] (i.e. transfinite interpolation) and paving.[8] These techniques cannot be used directly for mesh morphing because the correspondence between the source and target geometry is not determined by these algorithms. Furthermore, while mapping is fast, it is limited to structured meshes and so is insufficiently general. Paving has other limitations: it is slow (relative to mapping) and it is ill-posed, i.e. small variations in the boundary data may cause large changes in the interior mesh (including changes in nodal connectivity) and so is unsuitable for morphing.

Because of these limitations, a special copy/morph capability was developed in CUBIT that semi-automatically sets up a correspondence between source and target nodes.[5] To determine the co-ordinates of the interior target nodes, the copy capability uses an simple affine transformation based on a small subset of the source and target boundary nodes. If the source mesh can be copied onto the target mesh, the affine transformation copies correctly and quickly. However, for the more general morphing situation, the affine transformation may produce badly folded meshes due to insufficient regard for the complete target boundary dataset. To correct this situation, the Laplace smoother is applied in Reference 5 after the affine transformation. (In effect, the affine transformation serves merely as an initial guess for the Laplace smoother.) The smoother accounts for all the boundary data, but at the price of failing to produce exact copies. Figure 1 shows a 'copied' mesh that is produced by this technique (the two squares differ only by a translation). The Laplace smoothing has destroyed the exact copying that would have otherwise occured had only the affine transformation been used. It is clear that no smoothing should be used for pure copy operations while general morphing requires use of a different smoother, as is done in this paper.

It is emphasized that, if all that is wanted is mesh copying, then smoothing is not needed and would, in addition, be inefficient. The most efficient means of copying meshes is to establish a linear transformation matrix between source and target (a method for doing so is given in Reference 6). Such a transformation can be rapidly evaluated to copy meshes. However, if general morphing is wanted, then mesh smoothing becomes necessary. The cost of this added generality is that smoothing is much slower than evaluating a linear transformation.

Another application of morphing/smoothing in CUBIT is the Sweeping algorithm[6, 9] for two and a half-dimensional geometries. In this application, tube-like geometries are meshed by a sequence of morphed meshes forming a 2-D layer and leading from source to target. In the original CUBIT approach, each layer of nodes is Laplace smoothed. Figure 2 shows an example of how this approach is inadequate, i.e. can produce poor quality or even folded meshes. A variation in Sweep

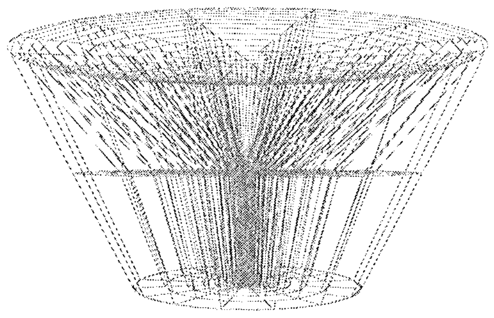

Figure 2. Sweep distortion from Laplace smoothing

is that the target surface may already be meshed and hence the morph needs to smoothly blend source and target meshes.

The ideas contained in this paper come from techniques in structured grid generation but are motivated by problems in unstructured meshing. Most significant in the pedigree of these ideas are the papers by Roache *et al.*[10] and Warsi.[11] The former is important in showing how to do mesh copying on structured grids for adaptivity while the latter is important in proposing the form of the weighted right-hand side of the smoother in this paper.

As a note of clarification, in the unstructured mesh literature, 'smoothing' generally refers to adjustment of nodal coordinates to improve mesh quality. In this paper, it is additionally meant by smoothing that some notion of ellipticity of the mesh generation operator is implicitly involved so that regularity of node spacing results.

In Section 2 a mathematical approach is described to show that mesh copying can be achieved via smoothing. In Section 3 a weighted form of the Winslow smoother applicable to unstructured meshes is presented while, in Section 4, examples of mesh Copying, Morphing, and Sweeping are given. Section 5 summarizes results and conclusions.

2. MESH COPY/MORPH

Let Ω_S and Ω_T be two bounded, topologically identical surfaces in R^3 (for copying geometrically identical surfaces are required). Call the former surface the 'Source' surface and the latter the 'Target' surface. Let $\partial\Omega_S$ and $\partial\Omega_T$ be the boundaries of the respective surfaces. It is assumed that the source surface has been meshed, i.e. there exists a set of points $\mathbf{x}_S \in \Omega_S$ with known connectivities and co-ordinates forming a quadrilateral mesh. In the following theory, the quality of the source mesh is irrelevant but, in most cases, the user of this method will want to ensure that the source mesh be of good quality before it is copied/morphed (why copy a bad mesh?). Assume that the target has been initially meshed in such a way that a one-to-one correspondence exists between the source and target nodes, both on the boundary and on the interior. Details of how this correspondence can be set up and established is discussed elsewhere.[5] It is desired to apply a 'smoother' to the initial target mesh to re-position the target nodes so that the target mesh is an exact copy of the Source mesh or, in the case of morph, the target is a high-quality, near-copy thereof.

Logically rectangular meshes are frequently determined by requiring them to globally satisfy a discretized elliptic equation. To extend such methods to unstructured meshes requires only that one relax the global requirement and impose the same discretized equation locally. To fix notation, let $\mathbf{x}_n = [x_n, y_n, z_n]^{\mathrm{T}} \in R^3$ be the position of the nth interior node in an unstructured quadrilateral mesh and let $\mathbf{x}_{n,m}$, $m = 0, 1, \ldots, M-1$ be the M nodes to which node n is connected by 'edges' (assume $M \geqslant 3$). For each of the M quadrilateral faces connected to $\mathbf{x}_n$, let $\hat{\mathbf{x}}_{n,m}$ be the node on the mth quadrilateral, opposite to $\mathbf{x}_n$. Let e_n, e_m, and d_m be scalars and define $D_n\mathbf{x} \in R^3$ to be the vector

$$D_n\mathbf{x} = e_n\,\mathbf{x}_n - \sum_{m=0}^{M-1} e_m\mathbf{x}_{n,m} - \sum_{m=0}^{M-1} d_m\,\hat{\mathbf{x}}_{n,m} \tag{1}$$

A mesh is determined by solving $D_n\mathbf{x} = 0$ for $\mathbf{x}_n$ when the coefficients e_n, e_m, and d_m are given. The vector $D_n\mathbf{x}$ serves as a discrete 'second-order' operator analogous to those used in structured mesh smoothing PDEs. For the theory that follows, assume that

$$e_n - \sum_{m=0}^{M-1} (e_m + d_m) = 0 \tag{2}$$

which is the 'row-sums equals zero' property. Note that the scalars in the relations above may depend upon the mesh itself, in which case $D_n\mathbf{x}$ will be a non-linear, discrete operator.

By analogy with logically rectangular surface mesh generators, the vector $D_n\mathbf{x}$ lies in the *tangent plane* of the surface to be meshed and can thus be resolved by any pair of linearly independent vectors that also lie in the tangent plane. Define 'tangent' vectors $\mathbf{u}_n \in R^3$, $\mathbf{v}_n \in R^3$ by

$$\mathbf{u}_n = \sum_{m=0}^{M-1} \alpha_m\mathbf{x}_{n,m} \tag{3}$$

$$\mathbf{v}_n = \sum_{m=0}^{M-1} \beta_m\mathbf{x}_{n,m} \tag{4}$$

with $\alpha_m, \beta_m \in R$ and

$$\sum_{m=0}^{M-1} \alpha_m = 0 \tag{5}$$

$$\sum_{m=0}^{M-1} \beta_m = 0 \tag{6}$$

To make the notation more precise, let $\mathbf{x}_n^{\mathrm{S}}$ signify the nth node of the source mesh and $\mathbf{x}_n^{\mathrm{T}}$ the nth node of the target mesh. A one-to-one correspondence exists between $\mathbf{x}_n^{\mathrm{S}}$ and $\mathbf{x}_n^{\mathrm{T}}$. The final target mesh is required to satisfy the initial Dirichlet data on the boundary of the target. In the interior, require the final Target mesh to satisfy the following discrete equations for each node $\mathbf{x}_n$:

$$D_n\mathbf{x}^{\mathrm{T}} = P_n|\mathbf{u}_n^{\mathrm{T}}|^{\gamma-1}\,\mathbf{u}_n^{\mathrm{T}} + Q_n|\mathbf{v}_n^{\mathrm{T}}|^{\gamma-1}\,\mathbf{v}_n^{\mathrm{T}} \tag{7}$$

where γ is a scalar to be defined later, and P_n and Q_n are weight functions that vary with node n to enable mesh copying. The weight functions are determined from the source mesh by replacing $\mathbf{x}^{\mathrm{T}}$ with $\mathbf{x}^{\mathrm{S}}$ in (7) and solving for P_n and Q_n:

$$P_n = \{(\mathbf{v}_n^{\mathrm{S}} \cdot \mathbf{v}_n^{\mathrm{S}})(\mathbf{u}_n^{\mathrm{S}} \cdot D_n\mathbf{x}^{\mathrm{S}}) - (\mathbf{u}_n^{\mathrm{S}} \cdot \mathbf{v}_n^{\mathrm{S}})(\mathbf{v}_n^{\mathrm{S}} \cdot D_n\mathbf{x}^{\mathrm{S}})\}/|\mathbf{u}_n^{\mathrm{S}}|^{\gamma-1}|\mathbf{u}_n^{\mathrm{S}} \times \mathbf{v}_n^{\mathrm{S}}|^2 \tag{8}$$

$$Q_n = \{(\mathbf{u}_n^{\mathrm{S}} \cdot \mathbf{u}_n^{\mathrm{S}})(\mathbf{v}_n^{\mathrm{S}} \cdot D_n\mathbf{x}^{\mathrm{S}}) - (\mathbf{u}_n^{\mathrm{S}} \cdot \mathbf{v}_n^{\mathrm{S}})(\mathbf{u}_n^{\mathrm{S}} \cdot D_n\mathbf{x}^{\mathrm{S}})\}/|\mathbf{v}_n^{\mathrm{S}}|^{\gamma-1}|\mathbf{u}_n^{\mathrm{S}} \times \mathbf{v}_n^{\mathrm{S}}|^2 \tag{9}$$

Equation (7) is a system of equations for the unknown target node locations $\mathbf{x}_n^{\mathrm{T}}$. Discussion of the exact form of the discrete operator and tangent vectors is defered to Section 3 to emphasize that the mesh copying property is independent of the operator.

It is first shown that, under appropriate constraints, the target mesh (i.e. the solution to (7)) is a copy of the source mesh. Let $\mathcal{R}$ be a 3×3 orthogonal transformation such that $\det \mathcal{R} = 1$ and $\mathcal{R}^{\mathrm{T}}\mathcal{R} = \mathcal{I}$. Let $\mathbf{w}$ be a 3×1 translation vector. The target mesh is a *copy* of the source mesh provided there exists $\mathcal{R}$, $\mathbf{w}$ such that

$$\mathbf{x}_n^{\mathrm{T}} = \mathcal{R}\,\mathbf{x}_n^{\mathrm{S}} + \mathbf{w} \tag{10}$$

for all nodes n on Ω_{T}.

By construction, the source mesh identically satisfies

$$D_n\mathbf{x}^{\mathrm{S}} = P_n\,|\mathbf{u}_n^{\mathrm{S}}|^{\gamma-1}\,\mathbf{u}_n^{\mathrm{S}} + Q_n\,|\mathbf{v}_n^{\mathrm{S}}|^{\gamma-1}\,\mathbf{v}_n^{\mathrm{S}} \tag{11}$$

for every node on Ω_{S}.

Theorem 1. Assume conditions (2), (5), *and* (6) *hold. Assume, further, that* (10) *holds between boundary nodes of the source and target. Then* (10) *holds on the interior if and only if* (7) *holds on the interior.*

Proof. For the first part of the proof, assume (10) holds on the interior. From the definitions it is straightforward to show the following:

$$\mathcal{R}\,\mathbf{u}_n^{\mathrm{S}} = \mathbf{u}_n^{\mathrm{T}} \tag{12}$$

$$\mathcal{R}\,\mathbf{v}_n^{\mathrm{S}} = \mathbf{v}_n^{\mathrm{T}} \tag{13}$$

$$\mathcal{R}(D_n\mathbf{x}^{\mathrm{S}}) = D_n\mathbf{x}^{\mathrm{T}} \tag{14}$$

Because the vector norm is rotationally invariant, $|\mathbf{u}_n^{\mathrm{S}}| = |\mathbf{u}_n^{\mathrm{T}}|$ and $|\mathbf{v}_n^{\mathrm{S}}| = |\mathbf{v}_n^{\mathrm{T}}|$. Applying $\mathcal{R}$ to both sides of (11) gives the first part of the theorem.

To prove the second part of the theorem, assume (7) holds on the interior and (10) holds on the boundary. Then one can solve (7) for P_n and Q_n:

$$P_n = \{(\mathbf{v}_n^{\mathrm{T}}\cdot\mathbf{v}_n^{\mathrm{T}})(\mathbf{u}_n^{\mathrm{T}}\cdot D_n\mathbf{x}^{\mathrm{T}}) - (\mathbf{u}_n^{\mathrm{T}}\cdot\mathbf{v}_n^{\mathrm{T}})(\mathbf{v}_n^{\mathrm{T}}\cdot D_n\mathbf{x}^{\mathrm{T}})\}/|\mathbf{u}_n^{\mathrm{T}}|^{\gamma-1}|\mathbf{u}_n^{\mathrm{T}}\times\mathbf{v}_n^{\mathrm{T}}|^2$$

$$Q_n = \{(\mathbf{u}_n^{\mathrm{T}}\cdot\mathbf{u}_n^{\mathrm{T}})(\mathbf{v}_n^{\mathrm{T}}\cdot D_n\mathbf{x}^{\mathrm{T}}) - (\mathbf{u}_n^{\mathrm{T}}\cdot\mathbf{v}_n^{\mathrm{T}})(\mathbf{u}_n^{\mathrm{T}}\cdot D_n\mathbf{x}^{\mathrm{T}})\}/|\mathbf{v}_n^{\mathrm{T}}|^{\gamma-1}|\mathbf{u}_n^{\mathrm{T}}\times\mathbf{v}_n^{\mathrm{T}}|^2$$

But, in view of (8) and (9), (10) must hold on the interior. □

This theorem shows that a copied mesh must satisfy equation (7) for any $\gamma \in R$, provided P_n and Q_n satisfy (8) and (9), respectively.

Mesh scaling: The previous result can be generalized to include changes of scale. Let $\rho \in R^+$ be a uniform stretching factor. The operator in (1) is required possess the following property for some $\gamma \in R$:

$$D_n(\rho\mathbf{x}) = \rho^{\gamma} D_n\mathbf{x} \tag{15}$$

Theorem 2. Assume conditions (2), (5), *and* (6) *hold. Assume, further, that*

$$\mathbf{x}_n^{\mathrm{T}} = \rho\,\mathcal{R}\,\mathbf{x}_n^{\mathrm{S}} + \mathbf{w} \tag{16}$$

holds between boundary nodes of the source and target. Then (16) *holds on the interior if and only if* (7) *holds on the interior.*

Proof. The proof goes the same, with only the minor change that $|\mathbf{u}_n^{\mathrm{T}}| = \rho\, |\mathbf{u}_n^{\mathrm{S}}|$ and $|\mathbf{v}_n^{\mathrm{T}}| = \rho\, |\mathbf{v}_n^{\mathrm{S}}|$. □

Solving (7) with appropriate boundary conditions will thus result in a target mesh that is a scaled copy of the source mesh.

3. MESH SMOOTHING

As noted in the previous section, the mesh copying method just presented is independent of the exact form of the discrete operator. To ensure well-posedness of (7) and hence, that small changes in the boundary data give rise to small changes in the target mesh, it is required that $D_n\mathbf{x}$ be a discrete approximation to an elliptic operator.

For unstructured meshes, the most commonly used 'smoother' is 'Laplace Smoothing.' Then

$$D_n\mathbf{x} = M\,\mathbf{x}_n - \sum_{m=0}^{M-1} \mathbf{x}_{n,m} \tag{17}$$

i.e. $e_n = M$, $e_m = 1$, and $d_m = 0$ for all m, n. Note that, due to the copying capability of (7), the Target Mesh will not be 'smooth' unless the Source Mesh is 'smooth,' even though an elliptic operator is used (this point is discussed in detail in Reference 10). For mesh scaling, the Laplace smoother requires $\gamma = 1$ to satisfy (15).

For mesh copying, the Laplacian smoothing operator is adequate (but unneccesary). For morph, because this smoother has no guarantee against mesh folding (i.e. negative Jacobians), it is possible for folded meshes to be the result of solving (7) with (17), especially if the target is a non-convex region. To avoid this situation an operator with a folding guarantee is needed. For mapped meshes this is given by the Winslow operator[12]

$$\mathcal{Q}_w\mathbf{x} = g_{22}\mathbf{x}_{\xi\xi} - 2\,g_{12}\mathbf{x}_{\xi\eta} + g_{11}\mathbf{x}_{\eta\eta} \tag{18}$$

To derive the corresponding $D_n\mathbf{x}$, discretize (18) assuming an unstructured, quadrilateral mesh. Perhaps the most natural approach to doing this is to use the Finite Element method, approximating (18) with linear elements, applying them to each quadrilateral cell (see Reference 13 for an example of this approach). As an alternative, it is possible to derive finite difference approximations, as the author showed in.[14]

Summarizing the results in Reference 14, the discrete Winslow operator for unstructured quadrilateral meshes is

$$D_n\mathbf{x} = G_{22}\,D_{\xi\xi}\mathbf{x} - 2\,G_{12}D_{\xi\eta} + G_{11}D_{\eta\eta} \tag{19}$$

with

$$G_{11} = D_\xi\mathbf{x}\cdot D_\xi\mathbf{x} \tag{20}$$

$$G_{12} = D_\xi\mathbf{x}\cdot D_\eta\mathbf{x} \tag{21}$$

$$G_{22} = D_\eta\mathbf{x}\cdot D_\eta\mathbf{x} \tag{22}$$

and

$$D_\xi \mathbf{x} = \frac{2}{M} \sum_{m=0}^{M-1} (\mathbf{x}_{n,m} - \mathbf{x}_n) \cos\theta_m \tag{23}$$

$$D_\eta \mathbf{x} = \frac{2}{M} \sum_{m=0}^{M-1} (\mathbf{x}_{n,m} - \mathbf{x}_n) \sin\theta_m \tag{24}$$

The second derivatives are constructed similarly, but depend on whether or not $M = 4$. For $M \neq 4$,

$$D_{\xi\xi} \mathbf{x} = \frac{2}{M} \sum_{m=0}^{M-1} (\mathbf{x}_{n,m} - \mathbf{x}_n)(4\cos^2\theta_m - 1) \tag{25}$$

$$D_{\xi\xi} \mathbf{x} = \frac{8}{M} \sum_{m=0}^{M-1} (\mathbf{x}_{n,m} - \mathbf{x}_n) \cos\theta_m \sin\theta_m \tag{26}$$

$$D_{\xi\xi} \mathbf{x} = \frac{2}{M} \sum_{m=0}^{M-1} (\mathbf{x}_{n,m} - \mathbf{x}_n)(4\sin^2\theta_m - 1) \tag{27}$$

where

$$\theta_m = \frac{2\pi m}{M} \tag{28}$$

Conditions (2), (5) and (6) are satisfied using these definitions of the operator and 'tangent' vectors, thus assuring the mesh copying property.

To summarize, system (7) is solved using (19) for the operator (note that $\gamma = 3$ for proper scaling), and using $D_\xi \mathbf{x}$, $D_\eta \mathbf{x}$ for the 'tangent' vectors $\mathbf{u}_n$, $\mathbf{v}_n$, respectively. Weights are given by (8) and (9). This is analogous to solving the continuum system

$$\mathcal{Q}_w \mathbf{x} = P_n\, g_{11} \mathbf{x}_\xi + Q_n\, g_{22} \mathbf{x}_\eta \tag{29}$$

which has appeared previously in the literature,[11] but with different definitions of P_n and Q_n because mesh copying was not the goal in that paper.

4. APPLICATIONS: MORPH AND SWEEP

In this section a few applications are given to illustrate the theory. The first two examples concern mesh morphing while the third illustrates mesh sweeping. The smoothing equations are solved iteratively in CUBIT, at present using Gauss–Seidel relaxation with loose convergence criteria (in the future CUBIT will use modern relaxation techniques to obtain smoothed meshes faster).

There are five steps that are needed to do a morph or copy. Only the last two steps are discussed in detail in this paper:

(1) Generate the source mesh,
(2) Generate the target boundary mesh,
(3) Generate an initial target mesh in the interior (needed for the iterative solver),
(4) Establish a one-to-one correspondence between source and target boundary nodes,
(5) Compute P_n and Q_n from the source mesh using (8) and (9),
(6) Solve (7) to get the final, 'morphed' target mesh.

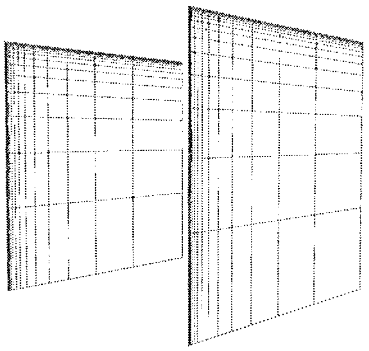
Figure 3. Mesh copying via smoothing: source (left) – target (right)

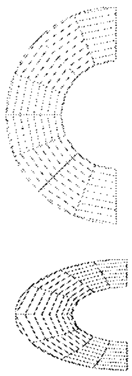
Figure 4. Mesh morphing via smoothing: source (top) – target (bottom)

Examples. Figure 3 shows how the weighted smoother is able to faithfully copy the source mesh from the example in Figure 1. Figure 4 shows how the weighted smoother can achieve a high-quality morphed mesh on a target surface that is similar to the source (the two horseshoe-shaped surfaces in the figure have different aspect ratios and are in the same plane). Figure 5 shows the mesh quality problem in the 'sweep' example of Figure 2 is fixed by the weighted smoother.

5. SUMMARY AND CONCLUSIONS

Although mesh copying is readily achieved via orthogonal transformations, mesh morphing between geometrically similar surfaces is more robustly achieved via weighted smoothing. Such a smoother has been constructed for unstructured quadrilateral meshes, based on ideas from structured mesh generation. The smoothing technique faithfully copies the source mesh onto the target when possible and produces near-copies when only morphing is possible. The weighted smoother, combined with morphing, is essential to successful implementation of the mesh sweeping algorithm. Examples from the CUBIT code illustrate these ideas.

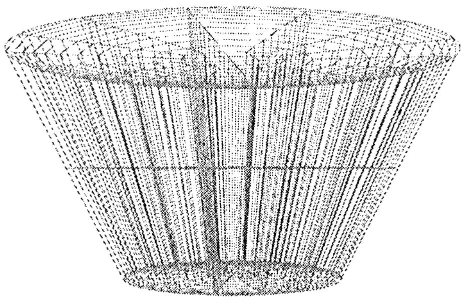

Figure 5. No sweep distortion using weighted morph smoother

Generalizations of this smoothing approach to hybrid quadrilateral/triangular meshes, three-dimensional hexahedral and hybrid hex/tet meshes, and to 'deforming' geometries are planned.

ACKNOWLEDGEMENTS

The author wishes to thank Stanly Steinberg, Tim Tautges, and the CUBIT gang. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-ACO4-94AL85000.

REFERENCES

1. R. Bank and B. Smith, 'Mesh smoothing using a posteriori error estimates', *SIAM J. Numer. Anal.*, **34**(3), 979–997 (1997).
2. L. Freitag, On Combining Laplacian and Optimization-based Mesh Smoothing Techniques, Trends in Unstructured Mesh Generation, AMD-Vol. 220, ASME, New York, 1997, pp. 37–43.
3. P. Zavattieri, 'Optimization strategies in unstructured mesh generation', *Int. J. Numer. Meth. Engng.*, **39,** 2055–2071 (1996).
4. T. Blacker *et al.*, *CUBIT Mesh Generation Environment, Vol. 1: User's Manual*, SAND94-1100, Sandia National Laboratories, Albuquerque, NM, May 1994.
5. D. Goodrich, 'Generation of all-quadrilateral surface meshes by mesh morphing', *M.Sc. Thesis*, Brigham Young University, 1997.
6. P. Knupp, 'Next-generation sweep tool: a method for generating All-Hex meshes on two-and-one-half dimensional geometries', *7th Int. Meshing Roundtable*, Detroit, MI, October 1998, pp. 505–513.
7. W. Cook and W. Oakes, 'Mapping methods for generating three-dimensional meshes', *Comput. Mech. Engng.*, **I,** 67–72 (1982).
8. T. Blacker and M. Stephenson, 'Paving: a new approach to automated quadrilateral mesh generation', *Int. J. Numer. Meth. Engng.*, **32,** 811–847 (1991).
9. L. Mingwu, S. Benzley, G. Sjaardema and T. Tautges, 'A multiple source and target sweeping method for generating all hexahedral finite element meshes', *Proc. 5th Int. Meshing Roundtable*, October 1996, Pittsburgh, pp. 217–225.
10. P. Roache, K. Salari and S. Steinberg, 'Hybrid adaptive Poisson grid generation and grid smoothness', *Commun. Appl. Numer. Meth.*, **7,** 345–354 (1991).
11. Z. U. A. Warsi, 'Basic differential models for coordinate generation', in J. F. Thompson (ed.), *Numerical Grid Generation*, North-Holland, New York, 1982, pp. 41–78.
12. A. Winslow, 'Numerical solution of the quasilinear Poisson equations in a nonuniform triangle mesh', *J. Comput. Phys.*, **2,** 149–172 (1967).
13. A. Allievi and S. Calisal, 'Application of Bubnov–Galerkin formulation to orthogonal grid generation', *J. Comput. Phys.*, **98,** 163–173 (1992).
14. P. Knupp, 'Winslow smoothing on two-dimensional unstructured meshes', *7th Int. Meshing Roundtable*, Detroit, MI, October 1998, pp. 449–457.
15. P. Knupp and S. Steinberg, *The Fundamentals of Grid Generation*, CRC Press, Boca Raton, FL, 1993.