A Comparison of Mesh Morphing Methods for Shape Optimization

Steven J. Owen[1] and Matthew L. Staten[2]

[1] Sandia National Laboratories†, Albuquerque, NM, U.S.A. sjowen@sandia.gov
[2] Sandia National Laboratories, Albuquerque, NM, U.S.A. mlstate@sandia.gov

1 Introduction

The modeling and simulation process for design often involves many small iterations through multiple changes to geometry. In an automated setting, geometric characteristics are represented as parameters that are driven via an optimization procedure. The ability to automatically adjust and update an existing mesh to conform to modifications in the geometry is a necessary capability that can enable rapid prototyping of many alternate geometric designs. It is clear however, that a mesh morph or update capability will have its limitations based upon the magnitude of the geometric changes required, however extending the range in which the connectivity of a mesh may be reused through many iterations of the design process is the desired outcome of this research.

In this research note we propose three different methods for morphing a mesh onto an updated geometry: 1. Smoothing, 2. Weighted Residuals, and 3. Simplex-based transformations. In each case we assume the geometric topology description of the model through all iterations of the optimization, remains constant. This assumption allows for the geometric ownership of nodes and elements in the mesh to also remain unchanged through design iterations.

For our purposes we begin with a geometry domain, $\Omega_G^n = \{G_i^r | r = 0, 1, 2, 3\}$ at iteration n, with geometry entities of dimension r. An existing finite element mesh, $\Omega_M^n = \{M_i^r | r = 0, 1, 2, 3\}$, also at iteration n, is assumed to have been generated and associated with its owning geometry in Ω_G^n. We seek a transformation $\Omega_M^n \Rightarrow \Omega_M^{n+1}$ given a new Ω_G^{n+1} such that the element quality in Ω_M^{n+1} is usable and preferably optimal. Since the connectivity of mesh entities in Ω_M^n and Ω_M^{n+1} will be identical, we seek only the nodal transformation $M_0^n \Rightarrow M_0^{n+1}$

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

2 Mesh Morphing Methods

In defining the transformation $M_0^n \Rightarrow M_0^{n+1}$ we address each dimension independently. For example, nodes are first transformed from vertices of Ω_G^n to Ω_G^{n+1}, followed by curves, surfaces and finally volumes. For vertices, knowing the new location of vertex j represented as $\mathbf{X}\left(\left[G_j^0\right]^{n+1}\right)$, the corresponding location of its associated node k can be represented as:

$$\mathbf{X}_0\left(\left[M_k^0\right]^{n+1}\right) = \mathbf{X}\left(\left[G_j^0\right]^{n+1}\right) \tag{1}$$

and for curves:

$$\mathbf{X}_1\left(\left[M_k^0\right]^{n+1}\right) = \left[G_j^1\right]^{n+1}(t_k) \tag{2}$$

where $\left[G_j^1\right]^{n+1}(t_k)$ is a simple parametric evaluation of curve j at parameter t_k where we have assumed $t_k = t_k^n = t_k^{n+1}$.

2.1 Smoothing

The first method employed, utilizes existing smoothing techniques incorporated in the Mesquite [1] toolkit. Using the nodal coordinates established on the curves as fixed, an optimization based smoothing technique is used to establish coordinate locations for interior nodes on surfaces. Mesquite's mean ratio, condition number [2] and untangling smoothing procedures are used adaptively based on local mesh quality. The same smoothing techniques are also employed for volumes, where the node locations established from the surface smoothing operations are now fixed and used as input to the volume smoothing operation.

2.2 Weighted Residuals

Also implemented is a method for morphing nodes on surfaces and volumes is a technique based upon the weighted residual method employed for hexahedral sweeping. In this method, the initial surface mesh, Ω_M^n, is first transformed using an affine transformation, $\mathbf{M}$, as described in [3], A correction is then applied to the transformed coordinate based upon the weighted sum of the nearby residual vectors of boundary nodes. The location of an interior surface node k, can be represented as:

$$\mathbf{X}_{2,3}\left(\left[M_k^0\right]^{n+1}\right) = \mathbf{M}\cdot\mathbf{X}_{2,3}\left(\left[M_k^0\right]^{n}\right) + \sum_{i=1}^{npts} w_i R_i \tag{3}$$

$$R_i = \mathbf{X}_1\left(\left[M_i^0\right]^{n+1}\right) - \mathbf{M}\cdot\mathbf{X}_1\left(\left[M_i^0\right]^{n}\right) \tag{4}$$

$$w_i = \frac{d_i^{-2}}{\sum_{j=1}^{npts} d_j^{-2}} \tag{5}$$

where R_i in equation (4) is the vector from boundary node i, where $\mathbf{M}$ has been applied at iteration n, to the location of the same boundary node i at iteration $n+1$ as determined from equation (2). The weight w_i in equation (5), is defined as the normalized inverse distance squared from the current node being computed to its surrounding boundary nodes. For efficiency, only a selection of the closest nodes is used to influence the weight w_i. Equation (3) can also be used to compute the interior node locations for volumes ($r = 3$).

2.3 Simplex-Based Transformations

The final method we propose is based upon a mesh sweeping method introduced by the authors in [4]. For this method we utilize a Delaunay tessellation of the boundary nodes [5]. For purposes of this study, only the 3D simplex-based transformation case was implemented, but used a weighted residual implementation for surface node transformations. For an interior node i in the volume, its enclosing tetrahedron, T_i^n is determined and its corresponding Barycentric coordinates, B_i is computed. We also define the requirement that connectivity for tetrahedron T_i^n defined at iteration n remains the same for all T_i^{n+1} at iteration $n+1$ and make the assumption that Barycentic coordinates, B_i, for node i with respect to T_i^n and T_i^{n+1} will be the same:

$$B_i = T_i^n \left(\left[M_{j=1,..4}^0 \right]^n \right) = T_i^{n+1} \left(\left[M_{j=1,..4}^0 \right]^{n+1} \right) \tag{6}$$

We can therefore define the interior node locations for a volume at iteration $n+1$ as:

$$\mathbf{X}_3 \left(\left[M_i^0 \right]^{n+1} \right) = \sum_{j=1}^{4} \left(B_i \right)_j \cdot \mathbf{X}_2 \left(T_i^{n+1} \right)_j \tag{7}$$

where $(B_i)_j$ is defined as the j^{th} component of B_i and $\mathbf{X}_2 \left(T_i^{n+1} \right)_j$ is the j^{th} vertex of tetrahedron T_i^{n+1}.

3 Examples

Two example problems are illustrated in figures 1-2 that focus on morphing of hexahedral meshes. In these examples, an initial geometry configuration is defined using parametric dimensioning. Figure 1 illustrates the parameters used in this model. A similar set of parameters was also defined for the other example as well as a range of acceptable values for each parameter. For each iteration, a perturbation of the parameters was defined and geometry was reconstructed using the new parameters. Mesh morphing was then performed to apply the mesh from the previous iteration to the new geometry. For both example cases the three proposed mesh morphing techniques were applied and resulting mesh quality tabulated. Initially, small perturbations were applied to the parameters to simulate small geometry changes over many design iterations. Secondly, we also looked at larger parametric changes, still within acceptable ranges, that may simulate fewer, but grosser changes to the geometry.

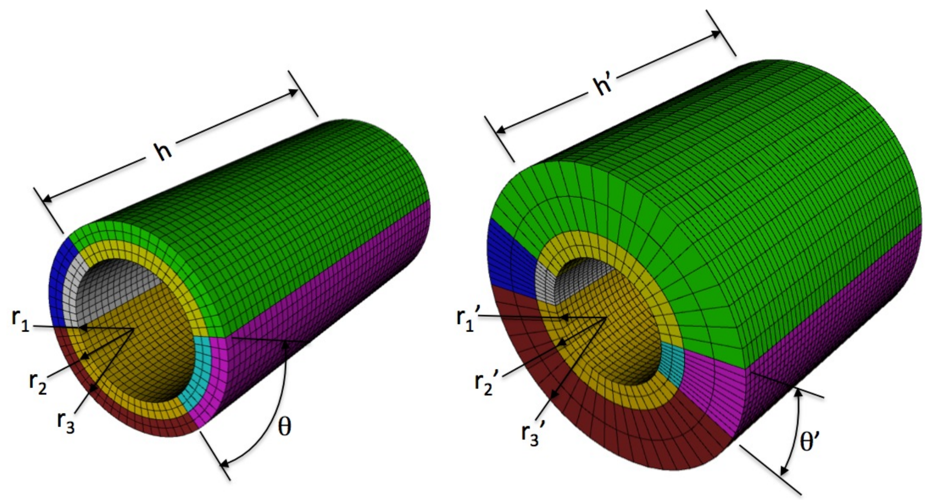

Fig. 1. Example problem "bore". Dimensions represent parameters which may change on the geometry

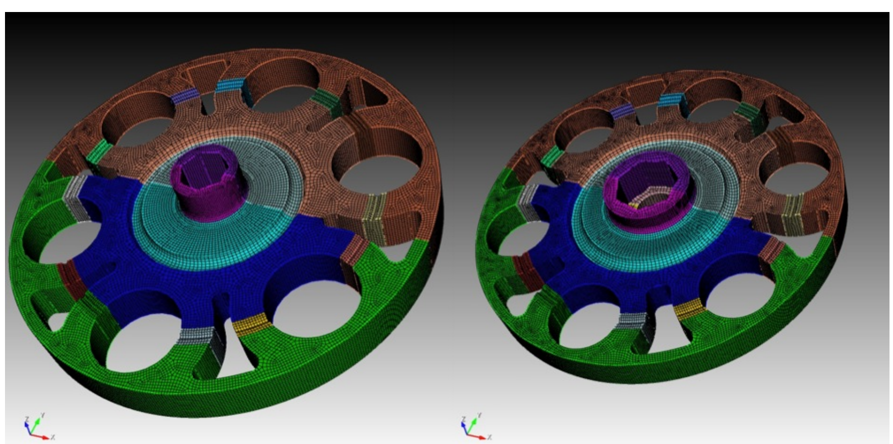

Fig. 2. Example problem "courier"

4 Summary of Results

Smoothing: Since the meshing package we used already incorporated the Mesquite smoothing tool, this method proved simplest to implement. Small changes in geometry were tracked very well, such as those represented by the "bore" model. Larger changes that required non-linear transformations, such as those in the "courier" model, proved impossible to capture. Inconsistent results were also noted depending on the starting iteration.

Weighted Residual: This method proved more efficient and more accurate than smoothing alone, with consistency improvements noted. Adding an additional iteration of smoothing following the weighted residual calculation proved effective in improving quality further. In general, we noted that the weighted residual calculation provided an effective initial guess to the smoothing process, making it much more efficient and consistent.

Simplex-Based Transformations: This method proved most effective for large transformations with complex geometry such as the "courier" model. For the selected models, we noted the greatest range in parameter changes with reasonable quality elements. We also noted that additional smoothing was not required for the models tested. Provided a simplex tessellation and point location tool is available, the implementation is also relatively simple.

In summary we have concluded that the simplex-based transformation method is the most effective for the widest range of geometries and parameter changes. Although efficiency of the simplex-based method is not quite as good for simple models when compared to smoothing and weighted residuals, larger, more complex models have proven faster and more accurate with this method.

References

1. Brewer, M., L. Diachin, P. Knupp, T. Leurent, and D. Melander, "The Mesquite Mesh Quality Improvement Toolkit", Proceedings, 12th International Meshing Roundtable, pp. 239-250 (2003)
2. Knupp, P., "Achieving Finite Element Mesh Quality via Optimization of the Jacobian Matrix Norm and Associated Quantities, Part I", Int. J. Num. Meth. Engr. Vol 48 pp. 401-420 (2000)
3. Knupp, P. "Next-Generation Sweep Tool: A Method For Generating All-Hex Meshes On Two-And-One-Half Dimensional Geomtries", Proceedings, 7th International Meshing Roundtable, pp.505-513 (1998)
4. Staten, M. L., S. A. Canann, and S. J. Owen, "BMSWEEP: Locating Interior Nodes During Sweeping," Engineering With Computers, Vol. 15 No. 3, pp.212-218 (1999)
5. Vavasis, S. "QMG: mesh generation and related software," http://www.cs.cornell.edu/home/vavasis/qmg-home.html (2010)