Multimaterial Mesh-Based Surface Tracking: Supplemental Material

Fang Da* Columbia University
Christopher Batty† University of Waterloo
Eitan Grinspun‡ Columbia University

Abstract

This supplementary document summarizes additional details of our multimaterial mesh-based surface tracking method, including multimaterial enhancements for collision-safe mesh improvement, expanded discussion of the zipper-based merging used as a comparison against snap-based merging, results of two scaling experiments on our current implementation, and a discussion of the parameters affecting the behavior of the method.

1 Multimaterial Mesh Improvement

Our multimaterial mesh-based surface tracking algorithm employs collision-safe, local remeshing [Brochu and Bridson 2009] to maintain the quality of the triangle mesh, while preserving intersection-safety. This is achieved via a set of local multimaterial remeshing operations which are direct extensions of classic mesh operations, including edge flipping, edge splitting, edge collapsing, and vertex smoothing. For completeness, we describe the details of these operations and their multimaterial modifications below.

1.1 Background

Remeshing of surface meshes is a very broad topic, with many applications and goals. Alliez et al. [2008] provide an overview of recent developments; we will briefly review work that is relevant to our setting. We focus on incremental local remeshing, since it allows each remeshing operation to be robustly checked for collision-safety [Brochu and Bridson 2009]. Various authors have applied this style of remeshing for surfaces undergoing gradual but large deformations (or *mesh adaptation*), and high quality results have been achieved by considering anisotropy, curvature, and feature-detection [Wicke et al. 2010; Jiao et al. 2010; Clark et al. 2012; Narain et al. 2012; Clausen et al. 2013]. We employ uniform resolution isotropic meshes with one particular choice of feature detection; combining our method with more advanced remeshing strategies is a key future direction. Several authors have also studied remeshing, subdivision, and fairing targeted at non-manifold surfaces [Hubeli and Gross 2000; Ying and Zorin 2001; Zilske et al. 2008; Pellerin et al. 2011].

1.2 Our approach

At each iteration, we perform passes of edge splitting, edge collapsing, edge flipping, and vertex smoothing. We cancel any operations that would introduce collisions, as well as those that would yield unacceptably poor quality angles, inverted normals (relative to the original local patch), tiny-area triangles, and large volume loss; following Brochu and Bridson [2009] we expose the target angle bounds, edge length bounds, minimum triangle area bound, and volume change bound as user parameters, as these may be problem-dependent (see also §3). These bounds are respected by each of our topological operations as well. (In particular, the maximum volume change bound should be chosen with this in mind; an overly strict limit can hinder merging.)

*e-mail:fang@cs.columbia.edu
†e-mail:christopher.batty@uwaterloo.ca
‡e-mail:eitan@cs.columbia.edu

We detect feature edges by thresholding dihedral angles between triangle pairs sharing an edge [Botsch and Kobbelt 2004; Dunyach et al. 2013], rather than analyzing the local quadric metric tensor [Brochu and Bridson 2009; Jiao et al. 2010]; we found this to be more robust and intuitive to control, and it extends naturally to the non-manifold case. We used a threshold of $30°$. Vertices lying on one or two feature edges are considered to belong to a feature curve or *ridge*, and vertices lying on three or more feature edges are identified as *peaks*.

1.3 Edge Flipping

The primary subtlety in performing edge flips is to ensure that triangle labeling remains correct, since two consistently labeled triangles sharing a manifold edge may nevertheless have opposing orientations. We pick one of the two incident triangles as a reference triangle, and use it to construct the resulting flipped two-triangle patch with the *same vertex winding order* (i.e., orientation) for both. The new triangles can then simply be assigned the label of the reference triangle. To preserve sharp features we disallow flipping of feature edges, though we do allow flipping of smooth *non*-feature edges that connect two feature vertices. We do not perform flipping on non-manifold edges, as this operation is not well-defined in general. Collision-checking follows Brochu and Bridson [2009].

Rather than flipping based on a Delaunay(-like) criterion, we follow Botsch and collaborators in seeking a mesh with more regular connectivity [Botsch and Kobbelt 2004; Dunyach et al. 2013]. That is, we perform flips that drive valences towards six for interior manifold vertices and four for boundary vertices. To handle manifold patches that border on non-manifold edges and vertices, we simply ignore triangles that are connected to the patch only through a non-manifold edge for the purposes of valence-counting, so that non-manifold vertices are seen as boundary vertices (i.e., with an ideal valence of four, within the manifold patch in question). We perform the flip if it reduces the total least squares difference between the ideal valences and the current valences.

$$\min \sum_{i=1}^{4} \left(\deg(\mathsf{v_i}) - \deg(\mathsf{v_i})^{opt}\right)^2 \tag{1}$$

In the above, $\mathsf{v_i}$ is one of the four vertices of the two-triangle patch, $\deg$ indicates the vertex valence (or degree), and $\deg(\mathsf{v_i})^{opt}$ is the optimal valence of $\mathsf{v_i}$ (either four or six).

1.4 Edge Splitting and Collapsing

Splitting and collapsing of non-manifold edges are straightforward extensions of the manifold case (Figure 1), and can be checked for collisions in the same way [Brochu and Bridson 2009]. We use an upper and lower edge-length bound to trigger splitting and collapsing, respectively. Child triangles created by an edge split inherit the labels of their parents. Edge collapses do not require relabeling, since the two triangles bordering the edge are deleted and the surrounding labels remain correct.

To generate smooth new vertex positions when splitting or collapsing, we use the modified butterfly scheme [Zorin et al. 1996; Brochu and Bridson 2009; Wojtan et al. 2010]. We treat both non-manifold edges and feature edges in the same manner as Zorin's original scheme treats boundary edges. This has two benefits. First, sequences of non-manifold or feature edges are subdivided by fitting a smooth cubic curve, thereby better preserving their shape. Second, a smooth region separated by a non-manifold curve or sharp feature curve from an adjacent smooth region uses information only from the "same side" to perform subdivision; as in the regular boundary edge case, reflection is used to derive ghost-data for "missing" triangles. This preserves the smoothness of patches on either side, such as in the case of the sharp intersection curve produced by the merging of two spheres, seen in our supplemental video. Similarly, for situations in which two or more surface patches are connected at only a single non-manifold vertex (i.e., share no edges or triangles), each patch is treated as a distinct manifold.

When collapsing an edge to a point, we can choose the position of the final point to be either one of the existing endpoints or a new point computed using subdivision. In smooth regions, subdivision is preferred, whereas near sharp features selecting one of the end points better maintains the shape. Following Brochu and Bridson [2009], we preserve peak vertices over ridge vertices, and ridge vertices over smooth non-feature vertices. In our non-manifold setting, when two vertices have the same feature type, we break the tie by preferring to keep a non-manifold vertex over a manifold one, to avoid perturbing triple- and higher-order junctions. If two vertices are equivalent in terms of features and manifoldness, we simply use the midpoint.

Taken together, these choices minimize the disruption of features and non-manifold boundaries that collapses can cause. For example, even for our simple example of merging of two spheres expanding and shrinking under normal flow, nïve remeshing can cause the intersection triple-curve to drift substantially, both normally and tangentially relative to the outer surface.

1.5 Vertex Smoothing

We apply smoothing of vertex positions to improve the shape of mesh triangles. Because naïve Laplacian smoothing tends to rapidly destroy volume, particularly in high curvature regions, we follow previous authors in applying *tangential* or *null-space* smoothing, which removes the normal component of the displacement induced by smoothing [Botsch and Kobbelt 2004; Jiao and Alexander 2005; Jiao et al. 2010]. For feature ridge vertices, smoothing-induced displacement perpendicular to the ridge direction is projected out instead so that smoothing occurs only along the ridge. For peak vertices, no smoothing is applied. We compute the vertex normal and ridge direction per Jiao and Bayyana [2008]. No special treatment is required for non-manifold geometry.

For strongly folded geometry (dihedral angles exceeding $165°$) we instead perform Laplacian vertex smoothing in the average plane of the fold, as a special case. That is, we find the average normal of the incident triangles, and project out smoothing perpendicular to that axis. The goal is to widen the angle at the fold, under the assumption that such very sharp folds correspond to a crease that is in the process of merging. Treating this case via smoothing rather than relying entirely on collision-induced merging produces smoother intersection curves, for example during the initial collisions of spheres undergoing normal flow, as seen in our video.

Smoothing yields an updated position for each vertex; these displacements are treated as pseudo-motions allowing collisions to be detected and resolved in the same manner as time integration [Brochu and Bridson 2009].

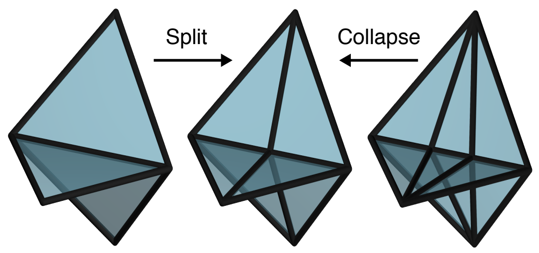

Figure 1: ***Non-manifold edge splits and collapses*** *closely parallel their manifold counterparts.*

1.6 Eliminating Very Poor Triangles

As discussed above, individual remeshing operations are canceled if they damage features or violate bounds on volume change, areas, angles, or edge lengths. However, in complex scenarios these potentially conflicting constraints can limit the remesher's ability to resolve poor triangles. For example, eliminating a poor quality triangle may require smoothing a vertex in a manner that damages a feature, or performing a collapse that temporarily introduces a very small angle. Therefore, if triangles with very poor angles (e.g., outside $[2°, 178°]$) remain after our standard remeshing pass, we turn to a more aggressive strategy: we apply our remeshing operations on those elements alone while ignoring all quality constraints *except intersection-safety*. We found this infrequently invoked fail-safe to be effective at eliminating poor triangles at the cost of additional localized regularization.

2 Multimaterial Zipper-based Merging

In the paper, we present a new snap-based merging approach that performs better than the zipper-based merging of Brochu and Bridson [2009]. This section describes how we extended zipper-based merging to the multimaterial case, in order to provide a valid practical comparison.

The basic two-material zipper-based merging scheme proceeds as follows. First, proximate edge-edge pairs are identified, and the two triangles incident to each edge are deleted. This deletion creates two quadrilateral holes which are zippered together using eight new triangles to create a seamless tube connecting the previously disjoint regions. (Stanculescu et al. [2011] similarly detected proximate vertices and zippered their entire one-rings.) If the connecting tube geometry induces new interpenetrations with itself or the existing mesh, the operation is canceled to preserve intersection-safety. We extended this zippering approach to multiple materials by simply leaving in place and appropriately *relabeling* one of the pre-existing two-triangle patches to create the separating interface (Figure 2).

While generally effective, the resulting zippering process has two flaws. First, retaining one of the two existing interfaces introduces a mild asymmetry in the merge operation; this could be corrected by adding the new separating interface at the midpoint of the tube, at the cost of additional geometry (i.e., a 16- rather than 8-triangle tube). More fundamentally, however, zippering relies on colliding geometries being in relatively ideal alignment in order for the proposed holes to be safely connected. We experimentally observed that many proposed merges are immediately canceled to preserve intersection-safety, and in complex collision scenarios there may simply not be sufficient space to safely zipper at all; this holds true for both the original two-phase and multimaterial versions. This has been known to delay or entirely prohibit topology change, leading

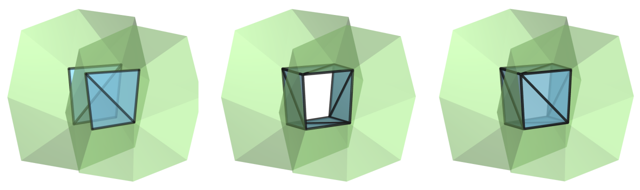

Figure 2: ***Zipper-based merging:*** *Left: Two nearby edges are identified, with incident triangles in blue. Deleting these triangles produces a quad-shaped hole in each surface. Middle: If the two outer materials match, the holes are zippered to form a connecting tube. Right: If the materials differ, one triangle-pair is preserved and its labels are modified to keep the two materials separated.*

to lingering surface noise in liquid animations [Brochu et al. 2010]. We therefore conclude that zippering's large edits limit its practical effectiveness.

3 Simulation Parameters

Our algorithm exposes a set of parameters that control the behavior of various mesh operations, allowing the method to cope with the diverse needs of different applications. Some of these parameters are drawn directly from the original *El Topo* method [Brochu and Bridson 2009], while a few are specific to our new multimaterial setting.

For most of the simulations presented in the paper, we used the following default set of parameter values. Maximum volume change allowed in any remeshing operation is set to $0.1\%\bar{L}^3$ where $\bar{L}$ is the mean of the upper and lower target bounds for edge lengths during remeshing. The minimum area triangle allowed to be created during remeshing is set to $2\%\bar{L}^2$. Vertex separation distance is set to $10\%\bar{L}$. The range of triangle internal angles allowed to be created by any remeshing operation is set to $[3°, 177°]$. In addition to edge length bounds, we also attempt to split edges whose opposing angle exceeds $160°$. The distance threshold that triggers proximity-based impulses during collision handling ranged from $10^{-4}\bar{L}$ to $10^{-3}\bar{L}$. The distance threshold at which merging is initiated ranged from $10^{-4}\bar{L}$ to $2 \times 10^{-2}\bar{L}$. The merge distance threshold should typically be set larger than the collision proximity threshold, since otherwise the collision code will act to keep surfaces too far apart to merge, which would clearly be counterproductive.

While these parameter settings were generally effective there are clear tradeoffs involved, which is why we chose to expose them to the user. For example, in situations where smaller perturbations at T1 transitions are desired, the user can specify a smaller separation distance for the *vertex separation* operation used to resolve irregular vertices. This reduces the size of the required instantaneous mesh modifications at the expense of reduced mesh regularity (i.e., a shorter new edge). Similarly, a low threshold on the allowable volume change during remeshing operations can better preserve volume, but requires canceling some operations that may have lead to better mesh quality.

4 Scaling Study

To experimentally examine how our method's performance currently scales with mesh size, we conducted a series of simulations with identical initial geometry, but varying remeshing resolution, for two different simulation scenarios:

1. face offsetting on two initially disjoint spheres, and

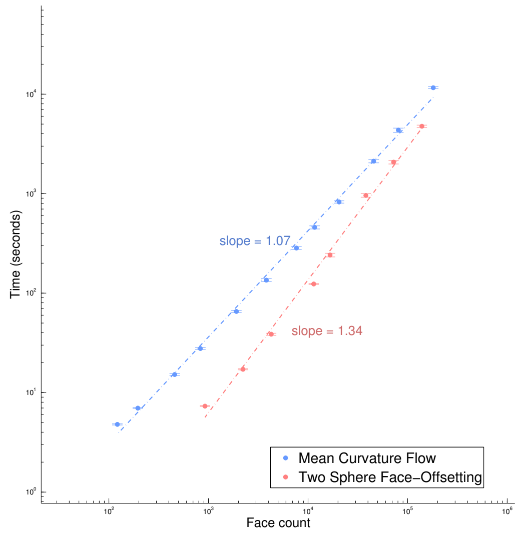

Figure 3: ***Scaling*** *of the total run time against mesh complexity.*

2. mean curvature flow on a cube divided into 20 Voronoi cells.

The total simulation time was recorded for each run and plotted against the average number of triangles throughout the simulation, shown in figure 3. The slope of the two fit lines suggest our overall method scales close to linearly with the number of triangles in the mesh. We nevertheless hope to improve scaling and extend our algorithm to even larger examples in the future by investigating parallelism, spatial adaptivity, and more advanced collision-culling strategies.

References

ALLIEZ, P., UCELLI, G., GOTSMAN, C., AND ATTENE, M. 2008. Recent advances in remeshing of surfaces. In *Shape Analysis and Structuring*, L. Floriani and M. Spagnuolo, Eds. Springer, Berlin, 53–82.

BOTSCH, M., AND KOBBELT, L. 2004. A remeshing approach to multiresolution modeling. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, ACM, New York, 185–192.

BROCHU, T., AND BRIDSON, R. 2009. Robust topological operations for dynamic explicit surfaces. *SIAM J. Sci. Comput. 31*, 4, 2472–2493.

BROCHU, T., BATTY, C., AND BRIDSON, R. 2010. Matching fluid simulation elements to surface geometry and topology. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 47.

CLARK, B., RAY, N., AND JIAO, X. 2012. Surface mesh optimization, adaption, and untangling with high-order accuracy. In *International Meshing Roundtable*, Springer, Berlin, X. Jiao and J.-C. Weill, Eds., 385–402.

CLAUSEN, P., WICKE, M., SHEWCHUK, J. R., AND O'BRIEN, J. F. 2013. Simulating liquids and solid-liquid interactions with Lagrangian meshes. *ACM Trans. Graph. 32*, 2, 17.

DUNYACH, M., VANDERHAEGHE, D., BARTHE, L., AND BOTSCH, M. 2013. Adaptive remeshing for real-time mesh deformation. In *Eurographics short papers*, Eurographics Association, Girona, Spain, 29–32.

HUBELI, A., AND GROSS, M. 2000. Fairing of non-manifold models for visualization. In *Proceedings of Visualization '00*, IEEE Computer Society Press, Salt Lake City, Utah, USA, 407–414.

JIAO, X., AND ALEXANDER, P. J. 2005. Parallel feature-preserving mesh smoothing. In *Proceedings of the 2005 international conference on Computational Science and Its Applications - Volume Part IV*, Springer-Verlag, Singapore, 1180–1189.

JIAO, X., AND BAYYANA, N. 2008. Identification of C1 and C2 discontinuities for surface meshes in CAD. *Computer Aided Design 40*, 2, 160–175.

JIAO, X., COLOMBI, A., NI, X., AND HART, J. 2010. Anisotropic mesh adaptation for evolving triangulated surfaces. *Engineering with Computers 26*, 4, 363–376.

NARAIN, R., SAMII, A., AND O'BRIEN, J. F. 2012. Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph. (SIGGRAPH Asia) 31*, 6, 147.

PELLERIN, J., LÉVY, B., AND CAUMON, G. 2011. Topological control for isotropic remeshing of nonmanifold surfaces with varying resolution: application to 3D structural models. In *IAMG*, International Association of Mathematical Geosciences, Salzburg, Austria, 678–688.

STANCULESCU, L., CHAINE, R., AND CANI, M.-P. 2011. Freestyle: Sculpting meshes with self-adaptive topology. *Computers and Graphics 35*, 3, 614–622.

WICKE, M., RITCHIE, D., KLINGNER, B. M., BURKE, S., SHEWCHUK, J. R., AND O'BRIEN, J. F. 2010. Dynamic local remeshing for elastoplastic simulation. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 49.

WOJTAN, C., THUEREY, N., GROSS, M., AND TURK, G. 2010. Physically-inspired topology changes for thin fluid features. *ACM Trans. Graph. (SIGGRAPH) 29*, 3, 50.

YING, L., AND ZORIN, D. 2001. Nonmanifold subdivision. In *Proceedings of Visualization '01*, IEEE, San Diego, CA, USA, 325 – 332.

ZILSKE, M., LAMECKER, H., AND ZACHOW, S. 2008. Adaptive remeshing of non-manifold surfaces. In *Eurographics short papers*, Eurographics Association, Crete, Greece.

ZORIN, D., SCHRÖDER, P., AND SWELDENS, W. 1996. Interpolating subdivision for meshes with arbitrary topology. In *SIGGRAPH 1996*, ACM, New Orleans, 189–192.