Deformation and Smooth Joining of Mesh Models for Cardiac Surgical Simulation

Hao Li[1], Wee Kheng Leow[1], Ing-Sh Chiu[2], and Shu-Chien Huang[2]

[1] Dept. of Computer Science, National University of Singapore,
Computing 1, Singapore 117590
{lihao,leowwk}@comp.nus.edu.sg
[2] Dept. of Surgery, National Taiwan University Hospital,
No. 7 Chung San South Road, Taipei, Taiwan, R.O.C.
ingsh@ntu.edu.tw, dtsurg99@yahoo.com.tw

Abstract. This paper focuses on an important aspect of cardiac surgical simulation, which is the deformation of mesh models to form smooth joins between them. A novel algorithm based on the Laplacian deformation method is developed. It extends the Laplacian method to handle deformation of 2-manifold mesh models with 1-D boundaries, and joining of 1-D boundaries to form smooth joins. Test results show that the algorithm can produce a variety of smooth joins common in cardiac surgeries, and it is efficient for practical applications.

Keywords: predictive surgical simulation, Laplacian mesh deformation, smooth join, discrete differential geometry.

1 Introduction

3D modeling and model deformation are indispensable components in computer simulation of surgery. In *reactive* surgical simulation [1,5,10], 3D models of body tissues deform in real-time according to applied forces provided by user inputs. Reactive simulation is suitable for surgical training and preoperative planning that involves only basic surgical operations.

In *predictive* surgical simulation [2, 4, 15], 3D models of body tissues may deform according to applied forces or geometrical constraints on the body tissues. The simulation system predicts surgical results given minimum user inputs. In this way, a surgeon can perform preoperative planning of complex surgical procedure without going through the entire procedure in detail.

To achieve the goal of predictive simulation of complex cardiac surgery, it is necessary to devise novel 3D model manipulation algorithms that satisfy the requirements of cardiac surgery and the physical properties of cardiac tissues. One of these important algorithms is the deformation of 3D models of cardiac tissues to form smooth joins, which is the focus of this paper.

This paper develops a novel algorithm that extends the Laplacian deformation method [8] to handle deformation of 2-manifold mesh models with 1-D boundaries, and joining of 1-D boundaries to form smooth joins. Test results show

F. Chen and B. Jüttler (Eds.): GMP 2008, LNCS 4975, pp. 483–490, 2008.
© Springer-Verlag Berlin Heidelberg 2008

that the algorithm can produce a variety of smooth joins common in cardiac surgeries, and is efficient for practical applications.

2 Related Work

Four main approaches exist for 3D model deformation: free-form deformation (FFD), finite element method (FEM), mass spring model (MSM), and differential geometry (DG) approach. FFD [7, 12] does not work directly on the geometric shape of an object. So it is difficult for FFD to manage geometrical constraints defined on the mesh in surgical simulation. FEM [2, 4, 15] offers accurate and realistic modeling of soft tissue deformation. However, it is computationally too expensive for real-time or near-real-time simulation. MSM [1, 5, 10] is efficient and easy to implement, so it is useful for real-time surgical simulation. However, its simulation results are not necessarily accurate because its behavior is highly dependent on the mesh topology and resolution.

DG approach directly computes the positions of the mesh points after deformation, given the initial configuration of the object and predefined constraints on the geometric properties of the mesh surface [8, 9, 13, 16]. DG offers more accurate and stable simulation results than MSM and is computationally less expensive than FEM. The standard DG method applies Laplacian operators to estimate the mean-curvature normals of the surface. As the Laplacian operators are ill-defined on mesh boundaries, existing Laplacian methods assume that the models are 2-manifold surfaces without boundaries. In contrast, our applications require the joining of meshes with boundaries.

One DG method for achieving smooth joins is to apply Laplacian filtering to smoothen the joined meshes [6]. This method is, however, not favored in our application because surface smoothing or fairing may cause the models to violate physical properties of cardiac tissues (see Section 3 for physical constraints).

Another DG method is to model the mesh surfaces using Poisson method [11, 16], which describes the differential properties of mesh surfaces using continuous partial differential equations. The Poisson method requires boundary conditions on all boundaries, which may not be known a priori in our applications.

3 Simulation of Smooth Joins

Suturing of cardiac tissues to form smooth joins are common operations in cardiac surgery. Figure 1 illustrates two typical scenarios: the suturing of coronary button to the aorta (Fig. 1(a)), and end-to-end joining of two arteries (Fig. 1(b)).

In this paper, the two anatomical parts to be joined are modeled as 3D meshes with zero surface thickness. One of the two parts is modeled as a fixed, rigid object called the *host* H. The other one is modeled as a flexible object called the *guest* G that is deformed to fit the host to form a smooth join.

The boundary curves of G and H to be joined, denoted as U and V, are identified by corresponding starting points $\mathbf{b}_s$ and $\mathbf{q}_s$, and end points $\mathbf{b}_e$ and $\mathbf{q}_e$ (Fig. 1). For closed curves, the starting points are also the end points. The curves

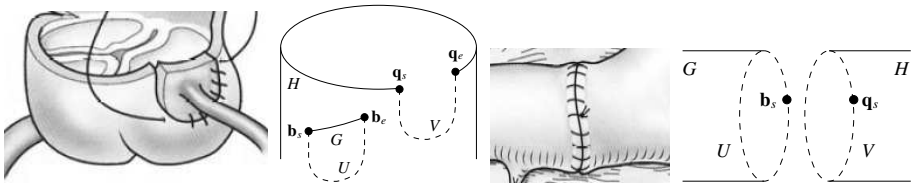

Fig. 1. Two kinds of smooth joins in cardiac surgery. (a, b) Partial patching. (c, d) End-to-end join. (a, c) Schematic drawings from [3]. Joins are indicated by the stitch marks. (b, d) Computational models. Dashed curves denote boundaries to be joined.

U and V may have different shape and length. The host is fixed and the guest is allowed to stretch or shrink. The minimum amount of deformation required to produce a smooth join is computed and displayed. In this way, a surgeon can assess whether the amount of deformation is acceptable empirically.

Now, we can define the joining problem as follows: Given two triangulated 2-manifold mesh models with 1-D boundaries, of which one is a rigid host H and the other a flexible guest G, and their corresponding boundary curves V and U to be joined, determine the shape of the deformed guest that forms smooth, continuous, and seamless join with the host. That is, the shape and length of the deformed U, denoted as U', should be identical to those of V.

The deformation of the guest model G is subjected to three soft constraints:

A. The shape of the deformed guest should be similar to that of the original guest, and its surface should be smooth.
B. The stretching or shrinking of the deformed guest should be minimized.
C. The join should be smooth and continuous.

4 Smooth Joining Algorithm

The algorithm for smoothly joining two mesh models consists of two steps:

1. Map boundary points $\mathbf{b}$ on U to the new positions $\mathbf{b}'$ on U'.
2. Deform G into G'.

Step 1 maps points $\mathbf{b}$ on U to the new positions $\mathbf{b}'$ on U' (Section 4.1). This mapping is imposed as positional hard constraints in the algorithm. Step 2 deforms the guest model G subject to the positional hard constraints and the three soft constraints. The algorithm is extended from the Laplacian method [8] (Section 4.2) to handle minimum stretching and shrinking (Section 4.3) and smooth joining of meshes (Section 4.4).

4.1 Mapping Corresponding Points

This step determines the corresponding point of each mesh point $\mathbf{b}$ on U. First, the lengths $l(U)$ of U and $l(V)$ of V are measured, and a scaling factor k is

computed as the ratio $l(U)/l(V)$. Then, each point $\mathbf{b}$ on U is mapped to a position $\mathbf{p}$ on V such that

$$d_U(\mathbf{b}_s, \mathbf{b}) = k\, d_V(\mathbf{q}_s, \mathbf{p}). \tag{1}$$

So, the new position $\mathbf{b}'$ of $\mathbf{b}$ is $\mathbf{p}$.

In real application, there may not be an existing mesh point at position $\mathbf{p}$ on V. To ensure a close fit between U and V, each mesh point $\mathbf{q}$ on V is also mapped to a corresponding position $\mathbf{a}$ on U according to Eq. 1. A new point and the associated edges are added to G at position $\mathbf{a}$.

4.2 Laplacian Mesh Deformation

In the Laplacian method of mesh deformation, the hard constraints on the positions of the mesh points on U', i.e., $\mathbf{b}' = \mathbf{q}$, are represented by the equation:

$$\mathbf{H}\,\mathbf{x} = \mathbf{d} \tag{2}$$

where $\mathbf{x}$ is a vector of all the mesh points in G' and $\mathbf{d}$ is a vector of the desired coordinates of $\mathbf{b}'$, and $\mathbf{H}$ is a matrix that relates $\mathbf{x}$ and $\mathbf{d}$.

Shape constraint (Constraint A) is imposed by minimizing the difference between the surface normals before and after deformation. The surface normal $\mathbf{n}(\mathbf{p})$ at mesh point $\mathbf{p}$ can be approximated by the Laplacian operator $\mathbf{l}(\mathbf{p})$:

$$\mathbf{l}(\mathbf{p}) = \sum_{\mathbf{v}_i \in N(\mathbf{p})} w_i\,(\mathbf{p} - \mathbf{v}_i) \tag{3}$$

where $N(\mathbf{p})$ is the set of neighboring vertices of $\mathbf{p}$, and $w_i = 1/|N(\mathbf{p})|$.

Shape constraint is imposed by minimizing the squared difference between the mean-curvature normals before and after deformation, i.e.,

$$\|\mathbf{l}(\mathbf{p}') - \mathbf{l}(\mathbf{p})\| \tag{4}$$

4.3 Minimum Stretching and Shrinking

Minimum stretching and shrinking (Constraint B) is achieved by preserving the distances between neighboring connected points. For each pair of connected points $\mathbf{p}_i$ and $\mathbf{p}_j$ in G, the stretching (and shrinking) energy to be minimized is:

$$k_{ij}\,(\|\mathbf{p}'_i - \mathbf{p}'_j\| - \|\mathbf{p}_i - \mathbf{p}_j\|)^2 \tag{5}$$

where k_{ij} is the stiffness coefficient. It is equivalent to minimizing the following:

$$k_{ij}\,\|\mathbf{d}_{ij} - \mathbf{s}_{ij}\|, \quad \text{where } \mathbf{d}_{ij} = \mathbf{p}'_i - \mathbf{p}'_j,\ \mathbf{s}_{ij} = \mathbf{d}_{ij}\|\mathbf{p}_i - \mathbf{p}_j\|/\|\mathbf{p}'_i - \mathbf{p}'_j\|. \tag{6}$$

4.4 Smooth Join

Our method of imposing Constraint C is to match the surface tangents of the corresponding boundary points on U' and V. As there is more than one surface tangent at a point, the one that is normal to the boundary curve is chosen.

Let $\mathbf{b}$ be a boundary point on U and $\mathbf{b}_1, \ldots, \mathbf{b}_n$ denote the neighboring points of $\mathbf{b}$ such that $\mathbf{b}_1$ is also a boundary point. Among these connected neighbors,

there is a neighbor $\mathbf{b}_t$ such that $\mathbf{b}-\mathbf{b}_t$ is (approximately) normal to the surface normal and U. We call this point the *tangential point*.

The tangential point $\mathbf{b}_t$ of $\mathbf{b}$ is computed as follows. First, the normal $\mathbf{n}(\mathbf{b})$ at $\mathbf{b}$ is estimated by the weighted mean of the normals of the neighboring faces of $\mathbf{b}$. Next, the required surface tangent $\mathbf{t}(\mathbf{b})$ is computed as the cross-product $\mathbf{n}(\mathbf{b})\times(\mathbf{b}-\mathbf{b}_1)$. The difference vector $(\mathbf{b}-\mathbf{b}_1)$ approximates the tangent of U at $\mathbf{b}$, which is normal to the required surface tangent. Then, the tangential point $\mathbf{b}_t$ of $\mathbf{b}$ is the point whose difference vector $\mathbf{b}-\mathbf{b}_t$ is most parallel to $\mathbf{t}(\mathbf{b})$. This tangential point is used as an approximation of the corresponding tangential point of $\mathbf{b}'$, whose position is to be computed by the deformation algorithm.

The desired surface tangent $\mathbf{w}(\mathbf{b}')$ at a boundary point $\mathbf{b}'$ is given by the tangent $\mathbf{t}(\mathbf{q})$ at the corresponding boundary point $\mathbf{q}$ on V. This tangent $\mathbf{t}(\mathbf{q})$ is computed in the same way as described above.

Now, the smooth join constraint can be specified as

$$\mathbf{b}'-\mathbf{b}_t=\mathbf{w}(\mathbf{b}'),\quad \text{where } \mathbf{w}(\mathbf{b}')=-\mathbf{t}(\mathbf{q})\|\mathbf{b}-\mathbf{b}_t\|/\|\mathbf{t}(\mathbf{q})\|. \tag{7}$$

4.5 Constrained Minimization

Assembling Eq. 4, 6 and 7 gives the total energy to be minimized subjected to the hard constraints described in Eq. 2:

$$\min_{\mathbf{x}}\|\mathbf{A}\,\mathbf{x}-\mathbf{c}(\mathbf{x})\| \quad \text{such that } \mathbf{H}\,\mathbf{x}=\mathbf{d} \tag{8}$$

where $\mathbf{c}(\mathbf{x})$ collects the $\mathbf{l}(\mathbf{p})$, $\mathbf{s}_{ij}$ and $\mathbf{w}(\mathbf{b}')$ terms, and $\mathbf{A}$ relates $\mathbf{x}$ and $\mathbf{c}(\mathbf{x})$.

This is a non-linear optimization problem, which can be solved iteratively using Gauss-Newton method, as demonstrated in [14], in conjunction with the equality-constrained least square method described in [8].

5 Experiments and Discussion

This section evaluates the effectiveness and efficiency of the proposed algorithm in producing smooth joins of mesh models. Two test scenarios were constructed manually based on the application examples illustrated in Fig. 1. Two variants of the algorithm was tested for comparison: with and without smooth join constraints. Three test samples were constructed for each scenario, with the length of the guest's boundary smaller than, equal to, and greater than that of the host's boundary. For partial patching, the guest models were flat surfaces, which agree with real applications. The amount of deformation (stretching or shrinking) of the guest model was computed by measuring the amount of change in the areas of the triangles before and after deformation.

Figures 2–3 show that the smooth join constraint is necessary for the algorithm to produce a smooth join between the guest and the host. The constraint ensures that the algorithm achieves a balance between smooth joining and preservation of the orientation and shape of the guest model.

When the guest's boundary was shorter than the host's boundary, the guest was stretched significantly to match the host (column (a)). When they had the

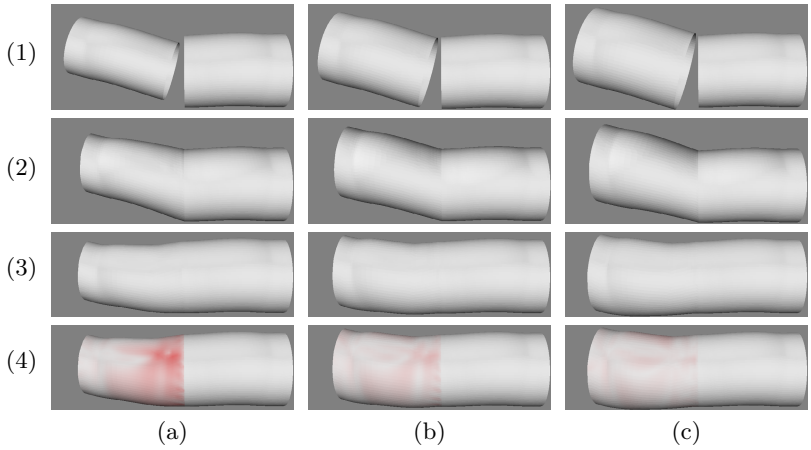

Fig. 2. Results of end-to-end joint. (1) Initial configurations, (2) without smooth join constraint, (3, 4) With smooth join constraint. The guest's boundary is (a) shorter than, (b) equal to, and (c) longer than the host's boundary. Red color indicates the amount of deformation of the guest model (white: no change, dark red: large change).

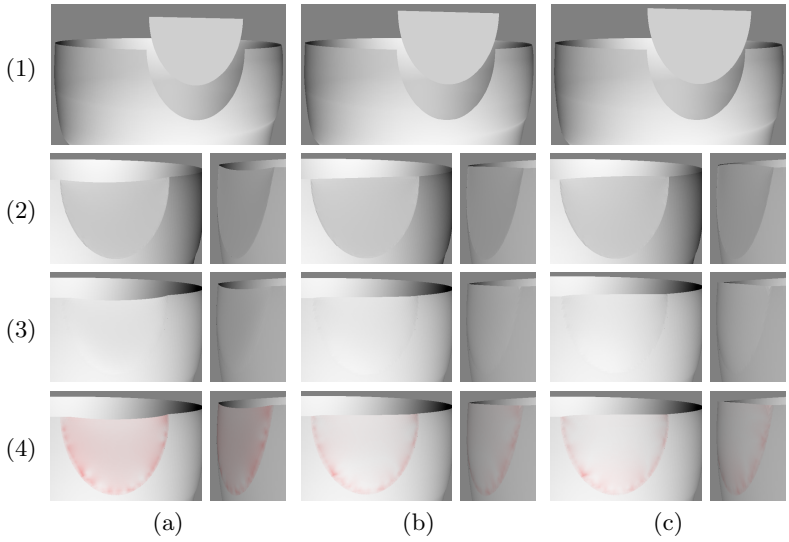

Fig. 3. Results of partial patching. (1) Initial configurations. (2) Without smooth join constraint. (3, 4) With smooth join constraint. The guest's boundary is (a) shorter than, (b) equal to, and (c) longer than the host's boundary. Red color indicates the amount of deformation of the guest model (white: no change, dark red: large change).

same length, the guest was minimally deformed (column (b)). When the guest's boundary was longer than the host's boundary (column (c)), the guest did not shrink significantly due to the minimum shrinking constraint. Instead, it bulged to match the shape of the host. This is an expected result in real applications.

The computation time of the algorithm was measured in an Intel Pentium 4 PC with 3GHz CPU and 1GB RAM to assess its efficiency. The algorithm took 70 seconds to solve the end-to-end join scenario (Fig. 2(4)), which was an optimization problem with 2,494 points, 522 hard constraints, and 29,688 soft constraints. For partial patching (Fig. 3(4)), with 1509 points, 1,218 hard constraints, and 17,967 soft constraints, the algorithm took 13 seconds to solve.

6 Conclusion

This paper presented a novel algorithm that extends the Laplacian deformation method to handle deformation of 2-manifold mesh models with 1-D boundaries, and joining of 1-D boundaries to form smooth joins. In addition to the positional and shape constraints in the original method, two other constraints are incorporated. The smooth join constraint ensures that the guest model is deformed properly to fit the host model and to produce a smooth join between their joining boundaries. The minimum stretching and shrinking constraint ensures that the deformation of the guest model is minimum. Test results show that the algorithm can produce a variety of smooth joins common in cardiac surgeries. Moreover, the algorithm can execute efficiently for practical applications.

References

1. Castañeda, M.A.P., Cosío, F.A.: Deformable model of the prostate for TURP surgery simulation. Computers & Graphics 28(5), 767–777 (2004)
2. Crouch, J.R., Merriam, J.C., Crouch, E.R.: Finite element model of cornea deformation. In: Duncan, J.S., Gerig, G. (eds.) MICCAI 2005. LNCS, vol. 3750, pp. 591–598. Springer, Heidelberg (2005)
3. Gardner, T.J., Spray, T.L.: Operative Cardiac Surgery, 5th edn. Arnold (2004)
4. Gladilin, E., Ivanov, A., Roginsky, V.: Generic approach for biomechanical simulation of typical boundary value problems in cranio-maxillofacial surgery planning. In: Barillot, C., Haynor, D.R., Hellier, P. (eds.) MICCAI 2004. LNCS, vol. 3217, pp. 380–388. Springer, Heidelberg (2004)
5. Kuhnapfel, U., Cakmak, H., Maass, H.: Endoscopic surgery training using virtual reality and deformable tissue simulation. Computers & Graphics 24, 671–682 (2000)
6. Liepa, P.: Filling holes in meshes. In: Proceedings of Eurographics/ACM SIGGRAPH symposium on Geometry processing, pp. 200–205 (2003)
7. MacCracken, R., Joy, K.I.: Free-form deformations with lattices of arbitrary topology. In: Proc. of ACM SIGGRAPH, pp. 181–188 (1996)
8. Masuda, H., Yoshioka, Y., Furukawa, Y.: Interactive mesh deformation using equality-constrained least squares. Computers & Graphics 30(6), 936–946 (2006)
9. Meyer, M., Desbrun, M., Schröder, P., Barr, A.H.: Discrete differential-geometry operators for triangulated 2-manifolds. In: Hege, H.-C., Polthier, K. (eds.) Visualization and Mathematics III, pp. 35–57. Springer, Heidelberg (2002)

10. Mosegaard, J.: LR-spring mass model for cardiac surgical simulation. In: Proc. of 12th Conf. on Medicine Meets Virtual Reality, pp. 256–258 (2004)
11. Park, S., Guo, X., Shin, H., Qin, H.: Surface completion for shape and appearance. The Visual Computer 22, 168–180 (2006)
12. Sederberg, T.W., Parry, S.R.: Free-form deformation of solid geometric models. In: Proc. of ACM SIGGRAPH, pp. 151–160 (1986)
13. Sorkine, O., Lipman, Y., Cohen-Or, D., Alexa, M., Rössl, C., Seidel, H.-P.: Laplacian surface editing. In: Proc. of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, pp. 175–184 (2004)
14. Weng, Y., Xu, W., Wu, Y., Zhou, K., Guo, B.: 2D shape deformation using nonlinear least squares optimization. The Visual Computer 22(9), 653–660 (2006)
15. Williams, C., Kakadaris, I.A., Ravi-Chandar, K., Miller, M.J., Patrick, J.C.W.: Simulation Studies for Predicting Surgical Outcomes in Breast Reconstructive Surgery. In: Ellis, R.E., Peters, T.M. (eds.) MICCAI 2003. LNCS, vol. 2878, pp. 9–16. Springer, Heidelberg (2003)
16. Yu, Y., Zhou, K., Xu, D., Shi, X., Bao, H., Guo, B., Shum, H.-Y.: Mesh editing with Poisson-based gradient field manipulation. ACM Trans. on Graphics 23(3), 644–651 (2004)