DOI: 10.3217/978-3-85125-528-7-10

Graph-Laplacian minimisation for surface smoothing in 3D finite element tetrahedral meshes

Richard Huber, Martin Holler and Kristian Bredies*

University of Graz, Institute for Mathematics and Scientific Computing

Abstract
We propose a new method to improve surface regularity of 3D tetrahedral meshes associated with finite element simulations of the heart. Our approach is to minimise the graph-Laplacian subject to suitable point constraints. These constraints are computed from the whole triangulation and prevent a worsening of mesh quality that would otherwise be caused by the smoothing. The resulting minimisation problem is solved via a primal-dual algorithm, leading to a method that globally updates vertex coordinates in each iteration. Experiments confirm that our method reduces surface oscillations of the mesh while preventing degeneration of the triangulation as indicated by mesh quality metrics.

1. Introduction

In biomedical engineering, the development of a realistic 3D simulation framework for the human heart is currently an active research topic. Such a framework would allow, for example, patient-specific models and more individualised treatment [6]. In order to carry out such simulations, 3D meshes are typically created from segmented magnetic resonance (MR) images, using mesh-generation software such as described in [12]. In view of the subsequent simulations, these procedures ensure a sufficient quality of the triangulation, as indicated by quality metrics, and prevent the creation of degenerate elements. However, due to physical limitations in the image acquisition and, consequently, a low resolution of the image data, such meshes often suffer from artifacts. Those appear in particular in form of oscillations on the otherwise smooth surface (see Section 5.).

It is the goal of this work to provide a method that reduces these oscillations, but maintains high mesh quality. To this aim, we minimise the graph-Laplacian under suitable constraints and adapt the mesh coordinates accordingly. The constraints are computed from the whole initial triangulation and ensure non-degeneracy of the resulting triangulation and maintenance of a high mesh quality, the latter being indicated by quality metrics.

As the computation of meshes from segmented image data and a subsequent reduction of mesh artifacts is a challenge that commonly appears in mesh generation for finite element simulations in many different contexts, a lot of research has already been carried out in that direction. Different to classical mesh improvement dedicated to enhancing the quality of the triangluation, that often focus on a local adaption of nodes [8, 9, 11], our method aims at reducing mesh artifacts and hence is more related to mesh denoising approaches. For the latter, we exemplary refer to [10, 13, 14] and the references therein for recent methods. For a general overview on mesh related topics see [2, 3].

*University of Graz, Institute for Mathematics and Scientific Computing, Heinrichstrasse 36, A-8010 Graz, Austria. Email: `richard.huber@edu.uni-graz.at`, `martin.holler@uni-graz.at`, `kristian.bredies@uni-graz.at`

2. Model problem

The initial setting is as follows: The 3D tetrahedral finite element mesh is given in form of a triangulation. In particular, the coordinates of points of the triangulation, together with edge information, and a masking of surface points is given. The triangulation is assumed to be regular, in particular, all tetrahedra are non-degenerate and disjoint except for their boundaries.

Since the above-described oscillatory artifacts appear on the surface of the mesh, we will only adapt surface points and use the position of interior points only to determine suitable point constraints. This also reduces the computational cost and memory requirements, however, will have some drawbacks as discussed in Section 6.

The triangulation of the surface induces a graph $G = (V, E)$ with vertices $V = \{v_1, \dots, v_N\}$, where N is the number of vertices, and edge set E such that there is an edge between v_i and v_j in G if, and only if, $\{v_i, v_j\} \in E$. We define $U := \mathbb{R}^{3\times N}$ to be the space of point-coordinates of the triangulation, where for $u \in U$, the jth coordinate of the vertex v_i is denoted by $u_i^j \in \mathbb{R}$. Further, we will use the notation $u^j \in \mathbb{R}^N$ for the vector containing all jth coordinates of u and $u_i \in \mathbb{R}^3$ for the coordinates of v_i. With this notation, we define the graph-Laplacian operator as the componentwise matrix-multiplication operator according to

$$\Delta u := \begin{pmatrix} \hat{\Delta} u^1 \\ \hat{\Delta} u^2 \\ \hat{\Delta} u^3 \end{pmatrix}, \text{with the matrix } \hat{\Delta} \in \mathbb{R}^{N\times N}, \text{ given as } \big(\hat{\Delta}\big)_{i,j} := \begin{cases} \mathrm{Deg}(v_i) & \text{if } i = j, \\ -1 & \text{if } \{v_i, v_j\} \in E, \\ 0 & \text{else,} \end{cases} \quad (1)$$

where $\hat{\Delta} u^j$ is a matrix-vector multiplication and $\mathrm{Deg}(v_i)$ denotes the degree of v_i, i.e., the number of neighbours of v_i in G.

In order to smooth the surface, new coordinates of the surface points are computed by minimising the graph-Laplacian under constraints designed to maintain the original mesh structure and to ensure non-degeneracy of the mesh. The minimisation problem is

$$u^+ \in \operatorname*{argmin}_{u\in U} \frac{1}{2}\|\Delta u\|_2^2, \qquad \text{subject to } u \in \Omega, \quad (2)$$

where the feasible set has the form $\Omega = \{u \in U \colon u_i \in \Omega_i \text{ for } i = 1, \dots, N\}$, with pointwise feasible sets Ω_i as defined in the next section. A solution u^+ corresponds to the coordinates of the nodes of the smoothed surface. Note that the topology of the mesh, and in particular the set of edges E, does not change and Δ is linear. A minimisation of $\|\Delta u\|_2^2$ results in the node coordinates adapting to the means of the surrounding ones, and thus, reduces the curvature of the surface. Hence, minimising the graph-Laplacian operator is expected to imply a smoothing of the surface mesh.

Well-posedness. As we will see in the next section, it is reasonable to choose Ω to be non-empty, bounded and closed. Hence, existence of a solution to (2) follows directly from continuity of $u \mapsto \|\Delta u\|_2^2$ and finite dimensionality of U.

3. Suitable constraints

Naturally, the solution of (2) should be close to the original data. Further, the choice of Ω is driven by two requirements, the convexity of Ω and the maintenance of mesh quality:

Mesh quality. An important aim is to keep mesh quality high, since this is needed for the finite element simulation to work. A high mesh quality means that the tedrahedra are non-degenerate, disjoint, and that there is only a small number of very flat tetrahedra. The latter is important since many flat tetrahedra would cause numerical problems in the simulations. Our assumption is that the quality of the original mesh is sufficiently high, therefore we design the constraints such that the movement of the nodes does not significantly worsen mesh quality. In particular, we want to guarantee that no self-intersection of the surfaces of the tetrahedra occurs.

Convexity of Ω. Convexity yields several advantages in optimisation, such as allowing to apply a large range of optimisation methods and ensuring that indeed global optima are approximated. Thus, we aim to define constraints which can be represented as a family of point constraints given in a way that the set of admissible point-coordinates is convex.

In summary, to achieve the best results with our method, we look for a convex set of constraints which allows sufficient movement of the nodes while maintaining a high mesh quality.

Adaptive constraints. We define Ω by fixing an individual radius r_i for each node v_i, and allowing the node only to move within a ball of this radius centered at its original location. Our approach to choose r_i is as follows: Let us fix a surface vertex v in a tetrahedron T. Since the goal is to avoid degenerate tetrahedra, one must in particular prevent self-intersection. Geometrically interpreted, this means that each of the nodes must not pass to the opposite side of T. This motivates the incorporation of the heights on the nodes in T. Indeed, if the other nodes did not change, the distance of v to the opposite side of T would be determined by the corresponding height h of the tetrahedron and one could use h as a limitation on how far the vertex is allowed to move. But since the movement of the other points of the tetrahedron also affects this consideration, and since the node v is not only a node of T, but of several neighbouring tetrahedra, we use all heights h of all tetrahedra containing v to define the constraints. Indeed, for a fixed node v_i, we denote by h_T the minimum of the four heights of a tetrahedron T and $\hat{h}_i = \min\{h_T : v_i \text{ contained in } T\}$. We limit the movements of v_i by $\alpha\hat{h}_i$ with a parameter $0 < \alpha < 1/2$, which is expected to ensure that, even though all nodes move simultaneously, no self-intersections occur. Let u_{0i} denote the original coordinates of the vertex v_i. Thus, the corresponding radii and the resulting feasible sets are given by

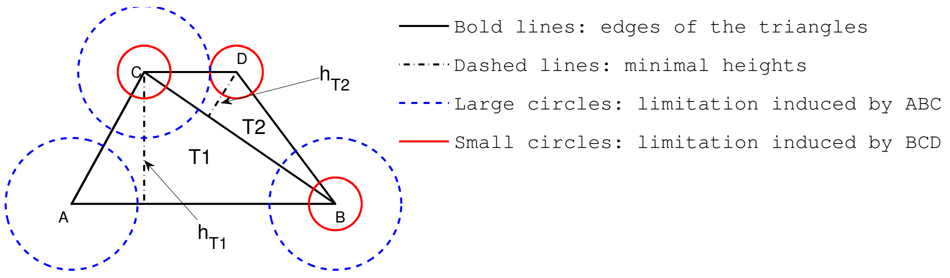
$$r_i = \alpha\hat{h}_i \quad \text{with } \hat{h}_i = \min\{h_T \colon v_i \in T\} \text{ and } h_T = \min\{h \colon h \text{ height of } T\}, \tag{3}$$

$$\Omega = \{u \in U \colon \|u_i - u_{0i}\| \le r_i \text{ for } i = 1, \dots, N\}. \tag{4}$$

This ensures that no self-intersection occurs and mesh quality is maintained. Figure 1 illustrates such constraints for the case of 2D triangles. In the three-dimensional setting, also the interior vertices adjacent to the surface of the mesh will be incorporated in the computation of constraints.

4. Numerical solution

The aim in this section is to describe an algorithmic framework for the solution of (2) with Ω as in (4). For this purpose, we will use the primal-dual algorithm described in [5], which is an iterative method that allows to solve convex-concave saddle-point problems with non-smooth structure. A non-smooth optimisation method is required to incorporate the proposed point constraint, however, due to differentiability of the graph-Laplacian regularisation and simplicity of the feasible set, also other methods, such as FISTA [1], could be used.



C
D
h_T2
T2
T1
A
B
h_T1
Bold lines: edges of the triangles
Dashed lines: minimal heights
Large circles: limitation induced by ABC
Small circles: limitation induced by BCD

Figure 1: Triangle $T1 = (A, B, C)$ with minimal height h_{T1} on C and triangle $T2 = (C, D, B)$ with minimal height h_{T2} on D. Limiting the movement of all nodes in a triangle by α times the minimum of the heights induces circles for each node, of which the smallest one is chosen as constraints.

In order to apply the primal-dual algorithm, Problem (2) is reformulated as a saddle-point problem according to

$$\min_{u\in\Omega} F(\Delta u) \quad \Longleftrightarrow \quad \min_{u\in U} F(\Delta u) + I_\Omega(u) \quad \Longleftrightarrow \quad \min_{u\in U} \sup_{w\in U} \langle w, \Delta u\rangle - F^*(w) + I_\Omega(u), \tag{5}$$

where $F(u) = \frac{1}{2}\|u\|_2^2$, the indicator function of Ω, i.e., $I_\Omega(u) = 0$ for $u \in \Omega$ and ∞ otherwise, and F^* is the convex conjugate of F, defined as $F^*(w) := \sup_{u\in U}\langle w, u\rangle - F(u)$. Explicitly, we get

$$F^*(w) = \sup_{u\in U} \langle w, u\rangle - \frac{1}{2}\|u\|_2^2 = \langle w, w\rangle - \frac{1}{2}\|w\|_2^2 = \frac{1}{2}\|w\|_2^2, \tag{6}$$

where the second equality is due to $u = w$ being the unique critical point of $u \mapsto \langle w, u\rangle - \frac{1}{2}\|u\|_2^2$, which can be confirmed by differentiation, and hence, $u = w$ being the unique global maximiser. Thus, (2) is reformulated as the following saddle point problem

$$\min_{u\in U} \max_{w\in U} L(u, w), \quad \text{where } L(u, w) = \langle w, \Delta u\rangle - \frac{1}{2}\|w\|_2^2 + I_\Omega(u). \tag{7}$$

The following proposition shows that by solving (7), we indeed obtain a solution of the original problem (2).

Proposition. *The saddle point problem* (7) *with feasible set* Ω *defined as in* (4) *admits at least one solution and for any saddle point* (u^+, w^+) *of* (7)*,* u^+ *is a solution of the original minimisation problem* (2).

Proof. Due to [7, VI Prop 2.4, p. 176], it is sufficient to show that for $L\colon U \times U \to \overline{\mathbb{R}}$ defined as in (7), for $u \in U$ fixed, $w \mapsto L(u, w)$ is concave and upper semi-continuous on U, and for $w \in U$ fixed, $u \mapsto L(u, w)$ is convex and lower semi-continuous on U. Further, we need to show that $u \mapsto L(u, w)$ is coercive for fixed w and that

$$\lim_{\substack{\|w\|\to\infty \\ w\in U}} \inf_{u\in U} L(u, w) = -\infty. \tag{8}$$

The convexity/concavity and l.s.c./u.s.c. assumptions are satisfied, in particular due to Ω being convex and closed, and $u \mapsto L(u, w)$ is coercive due to Ω being bounded. Further, for fixed $u \in \Omega$,

$$\lim_{\|w\|\to\infty} \langle w, \Delta u\rangle - \|w\|_2^2 \leq \lim_{\|w\|\to\infty} \|w\|\|\Delta u\| - \frac{1}{2}\|w\|_2^2 = \lim_{\|w\|\to\infty} \|w\|\Big(\|\Delta u\| - \frac{1}{2}\|w\|\Big) = -\infty$$

and hence, (8) holds, yielding the existence of a saddle point (u^+, w^+). Due to [7, III Prop 3.1, p. 57], the optimality of u^+ for (2) is a direct consequence of (5). □

The primal-dual algorithm for the solution of (7) will also require knowledge of the operator norm $\|\Delta\|$. An estimate can be found via power iteration [4], which computes λ_{max}, the eigenvalue of Δ with the greatest modulus if it is well separated from other eigenvalues. Note that this eigenvalue λ_{max} equals $\|\Delta\|$ due to Δ being symmetric and positive semidefinite.

The iteration steps of the primal-dual algorithm are given, in the abstract form, as

$$\begin{cases} w_{k+1} = (\operatorname{id} + \sigma \partial F^*)^{-1}(w_k + \sigma \Delta \bar{u}_k) \\ u_{k+1} = (\operatorname{id} + \tau \partial I_\Omega)^{-1}(u_k - \tau \Delta w_{k+1}) \\ \bar{u}_{k+1} = 2u_{k+1} - u_k \end{cases} \tag{9}$$

for suitable parameter $\tau, \sigma \in (0, \infty)$ such that $\|\Delta\|^2 \tau\sigma < 1$. Since $F^*(u) = \frac{1}{2}\|u\|_2^2$ is differentiable, a simple computation shows that $\partial F^*(u) = u$, thus

$$z = (\operatorname{id} + \sigma \partial F^*)^{-1}(u) \iff z + \sigma z = u \iff z = \frac{u}{1+\sigma}.$$

Further,

$$\begin{aligned} z = (\operatorname{id} + \tau \partial I_\Omega)^{-1}(u) &\iff z + \tau \partial I_\Omega(z) \ni u \iff 0 \in \partial\Big(\frac{1}{2}\|u - \cdot\|_2^2 + \tau I_\Omega(\cdot)\Big)(z) \\ &\iff z \in \operatorname*{argmin}_{v \in U} \|u - v\|_2^2 + \tau I_\Omega(v) \iff z \in \operatorname*{argmin}_{v \in \Omega} \|v - u\|_2^2 \\ &\iff z = P_\Omega(u), \end{aligned} \tag{10}$$

where $P_\Omega(u)$ denotes the projection of u onto Ω, i.e., onto the element in Ω with minimal distance to u. Hence, (10) can be solved by projecting onto the closest feasible point. We can compute this projection for each node individually since only point constraints are considered, i.e., whether or not $\|u_i - u_{0i}\| \le r_i$ does not depend on the other nodes' locations. The projection for each node is simply the projection on the ball of radius r_i centered at the original location u_{0i}, i.e.,

$$P_\Omega(u)_i = p(u_i, u_{0i}, r_i), \quad \text{with} \quad p(x, y, r) = \begin{cases} x & \text{if } \|x - y\| \le r, \\ \frac{r(x-y)}{\|x-y\|} + y & \text{else.} \end{cases} \tag{11}$$

Note that Ω, and hence, u_{0i} and r_i, do not change during the iteration and r_i is determined according to (3) and (4). By inserting (10) and (11) into (9), the iterations can be computed by simple arithmetic operations resulting in Algorithm 1.

Algorithm 1 Primal-Dual algorithm for minimising graph-Laplacian with adaptive constraints

Input: Original point-coordinates $\tilde{u}_0$ of mesh, edge information $\mathcal{E}$, masking of surface points S.

```
 1: $u_0 \leftarrow$ extract_surf_coo($\tilde{u}_0$), $r \leftarrow$ get_radii($\tilde{u}_0, \mathcal{E}, S$), $\Omega \leftarrow$ get_$\Omega(r, u_0)$     ▷ constraints
 2: $\Delta \leftarrow$ get_$\Delta(\mathcal{E}, S)$,   $\|\Delta\| \leftarrow$ powiter($\Delta$)     ▷ inititialisation of Laplacian
 3: $u \leftarrow u_0$, $\bar{u} \leftarrow u_0$, $w \leftarrow 0 \in \mathbb{R}^{3\times N}$, $\tau \leftarrow \|\Delta\|^{-1}$, $\sigma \leftarrow \|\Delta\|^{-1}$
 4: repeat
 5:     $w \leftarrow \frac{(w + \sigma\Delta\bar{u})}{(1+\sigma)}$     ▷ update of the dual variable
 6:     $\bar{u} \leftarrow P_\Omega(u - \tau\Delta w)$     ▷ update of the primal variable
 7:     $u \leftarrow 2\bar{u} - u$     ▷ update of the extragradient
 8:     $(u, \bar{u}) \leftarrow (\bar{u}, u)$     ▷ interchange of $u$ and $\bar{u}$
 9: until maximal number of iterations is reached
10: return $u$
```

Output: $u^+ = u$ surface point-coordinates of smoothed mesh.

Note that this is a global method, i.e., it updates the positions of all surface vertices in each iteration, unlike many other surface smoothing algorithms which operate pointwise.

Reiteration. In some situations, the proposed constraints are too restrictive, and hence, the smoothing results are not satisfactory. To overcome that, the point-constraints for each single point would need to be updated iteratively with the position of all other points. This would, however, result in a non-convex problem, preventing the computation of global optima.

A heuristic approach to still achieve some improvement, without re-designing the overall method, is to restart Algorithm 1 after convergence. To this aim, new constraints are computed from the output u^+ and the graph-Laplacian is optimised again subject to these updated constraints. This can be repeated a few times, e.g., 4 times, to allow some more flexibility in the constraint set. In practice, it can be reasonable to reduce the number of iterations performed in Algorithm 1, and do a few outer iterations in order to allow for more movement, while still guaranteeing that no self-intersection occurs and the mesh quality remains high.

Independent of such heuristics, the point-constraints of our method always ensure a non-degenerate triangulation. Also, the inner points of the mesh are not moved by our methods and hence limit the effect of the re-iteration. This, together with the point-constraints, in particular prevents a strong decrease of the volume of the shape, as frequently observed with unconstrained Laplacian smoothing.

5. Experimental results

The proposed method, although rather simple, is quite effective. It allows to smooth the surface and to reduce artifacts significantly while maintaining the original level of mesh quality. Figure 2 illustrates the effects of smoothing, with the original model on the left side, and the smoothed version on the right. The figure shows a mesh of a human heart, where the smoothed version was computed with 3 outer and 1000 inner iterations and with the constraint parameter $\alpha = 2/5$.

The effect of the proposed method on mesh quality can be evaluated quantitatively by measuring ρ, the skewness of a tetrahedron, i.e., the ratio of a tetrahedron's volume to its circumscribed ball's volume. Additionally, we quantify the change of the volume of each tetrahedron and identify changed orientations. This is done for each tetrahedron in the mesh by measuring the ratio of $\det(A)$ in the original and the smoothed mesh, denoted by θ, where A is a parallelepiped induced by a the tetrahedron.

Furthermore, one can observe maximal and minimal angles in the tetrahedra in order to find very flat tetrahedra. Table 1 depicts a quantitative evaluation of the effect of our method on mesh quality by comparing ρ for the original and the smoothed mesh and computing θ. As one can see, the number of flat structures does not increase significantly due to smoothing and for only 1% of the tetrahedra the volume reduced by more than one half. Further, we observed that no sign-flips of the determinant occurred, hence there are no self-intersections.

Percentiles of P	1%	5%	10%
Original mesh	0.0900	0.2350	0.3348
Smoothed mesh	0.0934	0.2047	0.2812

Percentiles of Θ	1%	5%
	0.5473	0.6648

Table 1: Mesh quality corresponding to mesh considered in Figure 2. Percentiles of P and Θ, where P is a vector of ρ for all tetrahedra and Θ is a vector of θ for all tetrahedra.

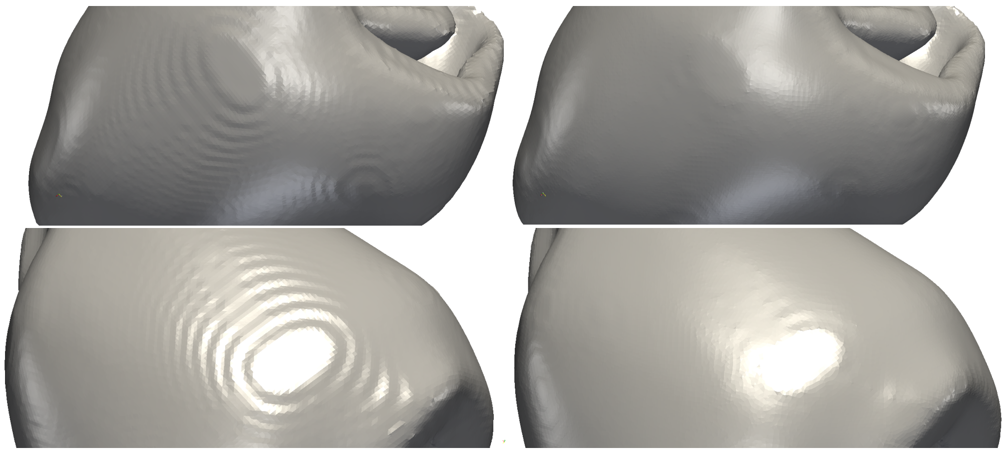


Figure 2: **3D triangulation of a human heart. The left figure shows the surface of the original mesh with artifacts, while the right shows the corresponding smoothed version where the artifacts are reduced.**

6. Discussion and outlook

The proposed method allows for improvement of the visual surface quality in 3D tetrahedral meshes. However, the procedure does not always succeed in removing all artifacts as can be observed particularly when there is an area lying dominantly above its surrounding surface like a plateau. The reason for this might be that only the surface, and thus, the outermost tetrahedra are changed, while the layer below remains unchanged. The constraints that avoid the loss of mesh quality are disadvantageous in this regard, since the second layer prevents the outer layer from sinking. Therefore, the plateau might remain dominant above its surrounding. A possible solution is not only to change the outermost layer, but also a few layers inside as well. However, this would, of course, increase computational costs. Another possibility would be to modify the constraints to avoid such a problem, in particular, also consider non-convex bounds. Indeed, this would allow for more flexibility in choosing the constraints, however, at the cost of losing the advantageous properties gained due to convexity.

References

[1] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[2] M. Botsch, L. Kobbelt, M. Pauly, L. Alliez, and B. Lévy. *Polygon mesh processing*. CRC press, Taylor and Francis, 2010.

[3] M. Botsch, M. Pauly, L. Kobbelt, P. Alliez, B. Lévy, S. Bischoff, and C. Rossl. Geometric modeling based on polygonal meshes. In *ACM SIGGRAPH Course Notes*, 2007.

[4] S. Börm and C. Mehl. *Numerical Methods for Eigenvalue Problems*. Walter de Gruyter, 2012.

[5] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2010.

[6] A. Crozier, C. M. Augustin, A. Neic, A. J. Prassl, M. Holler, T. E. Fastl, A. Hennenmuth, K. Bredies, T. Kuehne, M. J. Bishop, S. A. Niederer, and G. Plank. Image-based personalization of cardiac anatomy for coupled electromechanical modeling. *Annals of Biomedical Engineering*, 44(1):58–70, 2016.

[7] I. Ekeland. *Convex analysis and variational problems*. SIAM, 1999.

[8] L. Freitag and C. Ollivier-Gooch. Tetrahedral mesh improvement using swapping and smoothing. *International Journal for Numerical Methods in Engineering*, 40(21):3979–4002, 1997.

[9] B. M. Klingne and J. R. Shewchuk. Aggressive tetrahedral mesh improvement. In *Proceedings of the 16th International Meshing Roundtable*, pages 3–23. Springer, 2008.

[10] X. Lu, Z.Deng, and W. Chen. A robust scheme for feature-preserving mesh denoising. *IEEE Transactions on Visualization and Computer Graphics*, 22(3):1181– 1194, 2016.

[11] M. Ohlsson, P. Toft, L.K. Hansen, and F.A. Nielsen. Active surface models for brain imaging. *In Proceedings of the Interdisciplinary Inversion Workshop*, 5:68–77, 1997.

[12] A. J. Prassl, F. Kickinger, H. Ahammer, V. Grau, J. E. Schneider, E. Hofer, E. J. Vigmond, N. A. Trayanova, and G. Plank. Automatically generated, anatomically accurate meshes for cardiac electrophysiology problems. *IEEE Transactions on Biomedical Engineering*, 56(5):1318ÔÇô30, 2009.

[13] H. Zhang, C. Wu, J. Zhang, and J. Deng. Variational mesh denoising using total variation and piecewise constant function space. *IEEE Transactions on Visualization and Computer Graphics*, 21(7):873–886, 2015.

[14] Y. Zheng, H. Fu, O.J.C. Au, and C.L. Tai. Bilateral normal filtering for mesh denoising. *IEEE Transactions on Visualization and Computer Graphics*, 17(10):1521–1530, 2011.