Optimizing Surface Triangulation via Near Isometry with Reference Meshes

Xiangmin Jiao, Narasimha R. Bayyana, and Hongyuan Zha

College of Computing
Georgia Institute of Technology, Atlanta, GA 30332, USA
{jiao,raob,zha}@cc.gatech.edu

Abstract Optimization of the mesh quality of surface triangulation is critical for advanced numerical simulations and is challenging under the constraints of error minimization and density control. We derive a new method for optimizing surface triangulation by minimizing its discrepancy from a virtual reference mesh. Our method is as easy to implement as Laplacian smoothing, and owing to its variational formulation it delivers results as competitive as the optimization-based methods. In addition, our method minimizes geometric errors when redistributing the vertices using a principle component analysis without requiring a CAD model or an explicit high-order reconstruction of the surface. Experimental results demonstrate the effectiveness of our method.
Keywords: mesh optimization, numerical simulations, surface meshes, nearly isometric mapping

1 Introduction

Improving surface mesh quality is important for many advanced 3-D numerical simulations. An example application is the moving boundary problems where the surfaces evolve over time and must be adapted for better numerical stability, accuracy, and efficiency while preserving the geometry. Frequently, the geometry of the evolving surface is unknown a priori but is part of the numerical solution, so the surface is only given by a triangulation without the availability of a CAD model. The quality of the mesh can be improved by mesh adaptation using edge flipping, edge collapsing, or edge splitting (see e.g. [1,2,3,4,5,6]), but it is often desirable to fix the connectivity and only redistribute the vertices, such as in the arbitrary Lagrangian-Eulerian methods [7]. In this paper, we focus on mesh optimization (a.k.a. mesh smoothing) with fixed connectivity.

Mesh smoothing has a vast amount of literature (for example, see [8,9,10,11,12]). Laplacian smoothing is often used in practice for its simplicity, although it is not very effective for irregular meshes. The more sophisticated methods are often optimization based. An example is the angle-based method of Zhou and Shimada [12]. Another notable example is the method of Garimella et al. [8], which minimizes the condition numbers of the Jacobian of the triangles against some reference Jacobian matrices (RJM). More recently, the finite-element-based method is used in [9], but their method is relatively difficult to implement. Note that some mesh smoothing methods (such as the angle-based method) are designed for two-dimensional meshes, and the conventional wisdom is to first parameterize the surface locally or globally and then optimize the

flattened mesh [2,13,14,15]. To preserve the geometry, these methods typically require a smooth or discrete CAD model and associated point location procedures to project the points onto the surface, which increase the implementation complexity and may lose the optimality after projection.

The goal of this paper is to develop a mesh smoothing method that is as simple as Laplacian smoothing while being as effective as the sophisticated optimization-based methods without parameterizing the mesh. The novelty of our method is to formulate the problem as a near isometic mapping from an ideal reference mesh onto the surface and to derive a simple iterative procedure to solve the problem. Due to its variational nature, the method can balance angle optimization and density control. It also eliminates the needs of the pre-processing step of surface parameterization and the post-processing step of vertex projection in the conventional methods, making it easy to implement even on parallel computers. These properties make our method well-suited for integration into numerical simulations, such as those involving moving boundary problems.

The remainder of the paper is organized as follows. In Section 2, we formulate the mesh optimization problem and explain its relationship to isometric mappings. In Section 3, we describe a simple discretization of our method for triangulated surfaces with adaptive step-size control, which can be easily implemented and parallelized. In Section 4, we present some experimental results. Section 5 concludes the paper with discussions of future research directions.

2 Mesh Optimization and Isometric Mappings

Given a mesh with a set of vertices and triangles, the problem of mesh optimization is to redistribute the vertices so that the shapes of the triangles are improved in terms of angles and sizes. Surface parameterization is the procedure to map the points on one surface onto a parameter domain (such as a plane or sphere) under certain constraints such as preservation of areas or angles [16]. Although surface parameterization has been widely used in computer graphics and visualization [17,18,19,20,21,22], in this section we explore an interesting connection between it and mesh optimization.

More formally, given a 2-manifold surface $M \subset \mathbb{R}^3$ and a parameter domain $\Omega \subset \mathbb{R}^2$, the problem of *surface parameterization* is to find a mapping $f : \Omega \to M$ such that f is one-to-one and onto. Typically, the surface M is a triangulated surface, with a set of vertices $\boldsymbol{P}_i \in \mathbb{R}^3$ and triangles $T^{(j)} = (\boldsymbol{P}_{j1}, \boldsymbol{P}_{j2}, \boldsymbol{P}_{j3})$. The problem of *isometric parameterization* (or mapping) is to find the values of $\boldsymbol{p}_i$ such that $f(\boldsymbol{p}_i) = \boldsymbol{P}_i$, the triangles $t^{(j)} = (\boldsymbol{p}_{j1}, \boldsymbol{p}_{j2}, \boldsymbol{p}_{j3})$ do not overlap in Ω, and the angles and the areas of the triangles are preserved as much as possible.

We observe that the constraints of angle and area preservation in isometric parameterization are similar to angle optimization and density control in mesh optimization, except that the requirement of $\Omega \subset \mathbb{R}^2$ is overly restrictive. However, by replacing Ω with the input surface and replacing M by a "virtual" reference mesh with desired mesh quality and density (for example be composed of equilateral triangles of user-specified sizes, where the triangles need not fit together geometrically and hence we refer to it as a "virtual" mesh), we can then effectively use isometric parameterization as a computational framework for mesh optimization, which we describe as follows.

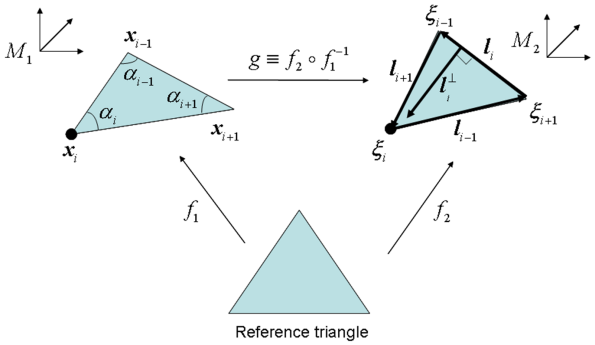


Figure1. Atomic mapping from triangle on M_1 to triangle on M_2.

Let M_1 denote an ideal virtual reference mesh and M_2 the triangulated surface to be optimized. Let us first assume that M_2 is a planar surface with a global parameterization $\boldsymbol{\xi}$, and we will generalize the framework to curved surfaces in the next section. In this context, the problem can be considered as reparameterizing M_2 based on the triangles and the metrics of M_1. Consider an atomic linear mapping $g \equiv f_2 \circ {f_1}^{-1}$ from a triangle on M_1 onto a corresponding triangle on M_2, as shown in Figure 1, where f_1 and f_2 are both mappings from a reference triangle. Let $\boldsymbol{J}_1$ and $\boldsymbol{J}_2$ denote their Jacobian matrices with respect to the reference triangle. The Jacobian matrix of the mapping g is $\boldsymbol{A} = \boldsymbol{J}_2 \boldsymbol{J}_1^{-1}$. For g to be nearly isometric, $\boldsymbol{A}$ must be close to be orthogonal. We measure the deviation of $\boldsymbol{A}$ from orthogonality using two energies: angle distortion and area distortion. Angle distortion depends on the condition number of the matrix $\boldsymbol{A}$ in 2-norm, i.e., $\kappa_2(\boldsymbol{A}) = \|A\|_2 \|A^{-1}\|_2$. Area distortion can be measured by $\det(\boldsymbol{A}) = \det(\boldsymbol{J}_2)/\det(\boldsymbol{J}_1)$, which is equal to 1 if the mapping is area preserving.

Let us define a function

$$\tau_p(s) \equiv \begin{cases} s^p + s^{-p} & \text{if } s > 0 \\ \infty & \text{otherwise,} \end{cases} \tag{1}$$

where $p > 0$, so τ_p is at the minimum if $s = 1$ and approaches ∞ if $s \leq 0$ or $s \to \infty$. To combine the angle- and area-distortion measures, we propose to use the energy

$$E_I(T, \mu_A) \equiv (1 - \mu_A)\tau_1(\kappa_2(\boldsymbol{A})) + \mu_A \tau_{\frac{1}{2}}(\det(\boldsymbol{A})), \tag{2}$$

where μ_A is between 0 and 1 and indicates the relative importance of area preservations versus angle preservation. For feasible parameterizations, E_I is finite because $\det(\boldsymbol{A}) > 0$ and $\kappa_2(\boldsymbol{A}) \geq 1$. To obtain a nearly isometric mapping, we must find $\boldsymbol{\xi}_i = g(\boldsymbol{x}_i)$ for each vertex $\boldsymbol{x}_i \in M_1$ to minimize the sum of E_I over all triangles on M_1, i.e.,

$$E_I(M_1, \mu_A) \equiv \sum_{T \in M_1} E_I(T, \mu_A). \tag{3}$$

We refer to this minimization as *nearly isometric parameterization of surfaces* (*NIPS*), which balances the preservation of angles and areas. This formulation shares similarity

with that of Degener et al. for surface parameterization [13], who also considered both area and angle distortions but used a different energy. Note that the special case with $\mu_A = 0$ would reduce to the most isometric parameterization (MIPS) of Hormann and Griener [19] and is closely related to the condition-number-based optimization in [8].

The direct minimization of E_I may seem difficult because of the presence of $\kappa_2(\boldsymbol{A})$. However, by using a result from [23] regarding the computation of Dirichlet energy $E_D(g) = \text{trace}(\boldsymbol{A}^T\boldsymbol{A})\det(J_1)/4$ as well as the fact that

$$\tau_1(\kappa_2(\boldsymbol{A})) = \kappa_F(\boldsymbol{A}) = \frac{\text{trace}(\boldsymbol{A}^T\boldsymbol{A})}{\det(\boldsymbol{A})} = \frac{4E_D(g)}{\det(\boldsymbol{J}_2)}, \tag{4}$$

one can show that

$$\tau_1(\kappa_2(\boldsymbol{A})) = \frac{1}{\det(\boldsymbol{J}_2)}\sum_i \cot\alpha_i \|\boldsymbol{l}_i\|_2^2, \tag{5}$$

where α_i and $\boldsymbol{l}_i \equiv \boldsymbol{\xi}_{i-} - \boldsymbol{\xi}_{i+}$ are defined as in Figure 1, and $i-$ and $i+$ denote the predecessor and successor of i in the triangle.

To minimize E_I, we must evaluate its gradient with respect to $\boldsymbol{\xi}_i$. For a triangle T, let $\boldsymbol{l}_i^\perp \equiv \widehat{\boldsymbol{n}} \times \boldsymbol{l}_i$ denote the 90° counter-clockwise rotation of the vector $\boldsymbol{l}_i$ on M_2. It can be verified that

$$\frac{\partial E_I(M,\mu_A)}{\partial \boldsymbol{\xi_i}} = \sum_{i\in T^{(j)}} s_{i+}^{(j)}\boldsymbol{l}_{i+}^{(j)} - s_{i-}^{(j)}\boldsymbol{l}_{i-}^{(j)} + t_i^{(j)}\boldsymbol{l}_i^{(j)\perp}, \tag{6}$$

where

$$s_\pm = (1-\mu_A)\frac{2\cot\alpha_\pm}{\det(\boldsymbol{J}_2)} \text{ and } t_i = (1-\mu_A)\frac{\kappa_F}{\det(\boldsymbol{J}_2)} + \frac{1}{2}\mu_A\frac{\det(\boldsymbol{J}_1)-\det(\boldsymbol{J}_2)}{\det(\boldsymbol{J}_1)^{\frac{1}{2}}\det(\boldsymbol{J}_2)^{\frac{3}{2}}}. \tag{7}$$

Equation (6) is a simple weighted sum of its incident edges and 90° counter-clockwise rotation of opposite edges over all its incident triangles $T^{(j)}$.

3 Mesh Optimization for Curved Surfaces

From Eqs. (6) and (7), it is obvious that $\partial E_I/\partial\boldsymbol{\xi}_i$ does not depend on the underlying parameterization $\boldsymbol{\xi}_i$ of M_2, so it becomes straightforward to evaluate it directly on a curved surface. In particular, at each vertex i on M_1, let $\boldsymbol{V}_i \equiv (\widehat{\boldsymbol{t}}_1|\widehat{\boldsymbol{t}}_2)_{3\times 2}$ denote the matrix composed of the unit tangents of M_2 at the point. To reduce error, we constrain the point to move within the tangent space of M_2 so that the displacements would be $\boldsymbol{V}_i\boldsymbol{u}_i$, where $\boldsymbol{u}_i \in \mathbb{R}^2$. Because of the linearity, from the chain rule we then have

$$\frac{\partial E_I(M_1,\mu_A)}{\partial \boldsymbol{u}_i} = \boldsymbol{V}_i^T\sum_{i\in T^{(j)}} s_{i+}^{(j)}\boldsymbol{l}_{i+}^{(j)} - s_{i-}^{(j)}\boldsymbol{l}_{i-}^{(j)} + t_i^{(j)}\boldsymbol{l}_i^{(j)\perp}, \tag{8}$$

where $\boldsymbol{l}_i \in \mathbb{R}^3$ as defined earlier. This equation constrains the search direction within the local tangent space at vertex i without having to project its neighborhood onto a plane.

We estimate the tangent space using a principal component analysis as in [24]. In particular, at each vertex v, suppose v is the origin of a local coordinate frame, and m is the number of the faces incident on v. Let $\boldsymbol{N}$ be an $3 \times m$ matrix whose ith column vector is the unit outward normal to the ith incident face of v, and $\boldsymbol{W}$ be an $m \times m$ diagonal matrix with W_{ii} equal to the face area associated with the ith face. Let $\boldsymbol{A}$ denote $\boldsymbol{N}\boldsymbol{W}\boldsymbol{N}^T$, which we refer to as the *normal covariance matrix*. $\boldsymbol{A}$ is symmetric positive semi-definite with real eigenvalues. We use the vector space spanned by the eigenvectors corresponding to the two smaller eigenvalues of $\boldsymbol{A}$ as the tangent space. If the surface contains ridges or corners, we restrict the tangent space to contain only the eigenvector corresponding to the smallest eigenvalues of $\boldsymbol{A}$ at ridge vertices and make the tangent space empty at corners.

To solve the variational problem, one could use a Gauss-Seidel style iteration to move the vertices. This approach was taken in some parameterization and mesh optimization algorithms [3,8]. For simplicity and ease of parallelization, we use a simple nonlinear Jacobi iteration, which moves vertex i by a displacement of

$$\boldsymbol{d}_i \equiv -\boldsymbol{V}_i \boldsymbol{V}_i^T \frac{\sum_{i \in T^{(j)}} \left(s_{i+}^{(j)} \boldsymbol{l}_{i+}^{(j)} - s_{i-}^{(j)} \boldsymbol{l}_{i-}^{(j)} + t_i^{(j)} \boldsymbol{l}_i^{(j)\perp} \right)}{\sum_{i \in T^{(j)}} \left(s_{i+}^{(j)} + s_{i-}^{(j)} \right)}. \tag{9}$$

This Jacobi-style iteration may converge slower than the Gauss-Seidel style iteration but it can be more efficient in practice since in the Gauss-Seidel iteration after moving a vertex v one needs to reestimate the local tangent space at the neighbor vertices of v.

The concurrent motion of the vertices in the Jacobi-style iterations may lead to mesh folding. To address this problem, we introduce an asynchronous step-size control. For each triangle $\boldsymbol{p}_i \boldsymbol{p}_j \boldsymbol{p}_k$, we solve for the maximum $\alpha \leq 1$ such that the triangle $\boldsymbol{p}_i^{(\alpha)} \boldsymbol{p}_j^{(\alpha)} \boldsymbol{p}_k^{(\alpha)}$ does not fold, where $\boldsymbol{p}_i^{(\alpha)} \equiv \boldsymbol{p}_i + \alpha \boldsymbol{d}_i$. We reduce $\boldsymbol{d}_i$ at vertex i by a factor equal to the minimum of the αs of its incident faces. After rescaling the displacement of all the vertices, we then recompute α and repeat the rescaling process until $\alpha = 1$ for all vertices. This asynchronous step-size control avoids mesh folding while allowing the energy decrease quickly.

4 Experimental Results

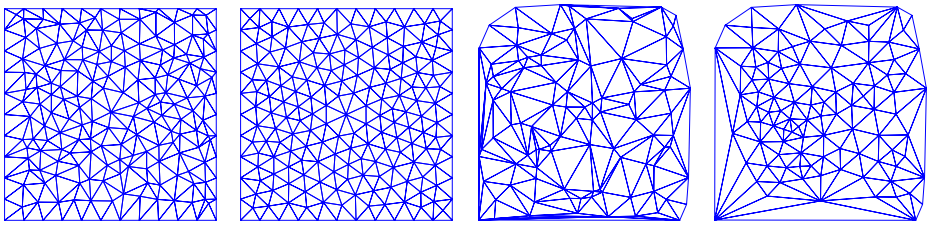
In this section, we present some preliminary results using our method for static 2-D meshes and dynamic 3-D meshes.

4.1 Optimization of 2-D Meshes

We first compare the effectiveness of our method against the length-weighted Laplacian smoothing and the angle-based method of Zhou and Shimada [12]. Because these existing methods are better established for planar meshes, we perform our comparisons in 2-D. Table 1 shows the minimum and maximum angles of six different meshes before and after mesh optimization, including three relatively uniform meshes (denoted by U1, U2, and U3) and three meshes with random points (denoted by R1, R2, and R3). In our

Table1. Comparative results of optimizing 2-D meshes. CN-based correspond to our method with $\mu_A = 0$ and CNEA-based with $\mu_A = 1$. Symbol '-' indicates occurrence of mesh folding.

	Minimum angle						Maximum angle					
	U1	U2	U3	R1	R2	R3	U1	U2	U3	R1	R2	R3
Original	12.0	8.8	8.5	0.11	1.1	0.043	129.5	156.9	147.5	179.7	176.9	179.9
Laplacian	33.1	31.4	30.8	5.8	4.9	4.5	99.4	105.9	109.2	168.3	170.8	171.0
Angle-based	32.9	30.9	29.5	3.9	-	-	96.0	103.7	105.9	170.3	-	-
CN-based	36.0	36.1	34.6	12.6	8.9	10.3	96.8	100.1	105.6	153.2	157.3	156.2
CNEA-based	35.7	35.3	34.2	12.7	7.8	8.9	97.0	101.3	105.8	153.9	163.4	160.6



(a) (b) (c) (d)

Figure2. Sample meshes (a,c) before and (b,d) after mesh optimization.

methods, we consider the virtual reference mesh to be composed of equilateral triangles with the average area of the triangles. Figure 2 shows the original and the optimized meshes using our method with $\mu_A = 1$ for U1 and R1. In nearly all cases, the condition-number based method (i.e., $\mu_A = 0$) performs substantially better in minimum angles for all cases, comparative or slightly better in maximum angles for uniform meshes, and significantly better in maximum angles for the random meshes. The area-equalizing optimization delivers slightly worse angles than condition-number based optimization, but the former still outperforms Laplacian smoothing and the angle-based methods while allowing better control of triangle areas. For the random meshes, the angle-based method caused folded triangles because unlike our method its minimum-energy state does not prevent mesh folding.

4.2 Optimization of Dynamic Meshes

To demonstrate our method for surface meshes, we optimize a mesh that is deformed by a velocity field. Our test mesh discretizes a sphere with radius 0.15 centered at $(0.5, 0.75, 0.5)$ and contains 5832 vertices and 11660 triangles. The velocity field is given by

$$\begin{aligned} u(x,y,z) &= \cos(\pi t/T)\sin^2(\pi x)(\sin(2\pi z) - \sin(2\pi y)), \\ y(x,y,z) &= \cos(\pi t/T)\sin^2(\pi y)(\sin(2\pi x) - \sin(2\pi z)), \\ z(x,y,z) &= \cos(\pi t/T)\sin^2(\pi z)(\sin(2\pi y) - \sin(2\pi x)), \end{aligned} \tag{10}$$

where $T = 3$, so that the shape is deformed the most at time $t = 1.5$ and should return to the original shape at time $t = 3$. We integrate the motion of the interface using the face-offsetting method in [24] while redistributing the vertices using the initial mesh as the reference mesh. In this test the angles and areas of the triangles were well preserved even after very large deformation. Furthermore, the vertex redistribution introduced very small errors and the surface was able to return to a nearly perfect sphere at time $t = 3$. This example demonstrates that our proposed technique is effective for optimizing dynamic surface meshes.

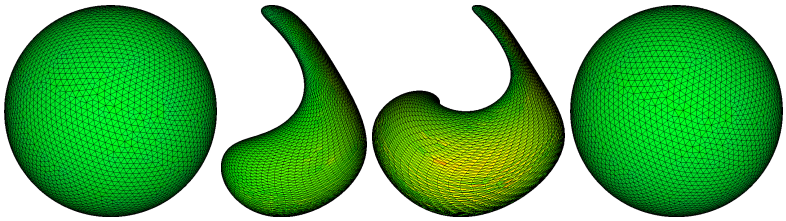


Figure3. Direct optimization of dynamic surface in reversible flow. Colors indicate triangle areas.

5 Conclusion

In this paper, we proposed a new method for optimizing surface meshes using a near isometry with reference meshes. We derived a simple discretization, which is easy to implement and parallelize and is well suitable for integration into large-scale numerical simulations. We compared our method with some existing methods, showed substantial improvements in the maximum and minimum angles, and demonstrated its effective use for moving meshes. As a future direction, we plan to extend our method to optimize quadrilateral meshes and 3-D volume meshes. It is also worth exploring more efficient procedures for solving the equations resulted from our variational formulation.

Acknowledgments This work was supported by a subcontract from the Center for Simulation of Advanced Rockets of the University of Illinois at Urbana-Champaign funded by the U.S. Department of Energy through the University of California under subcontract B523819 and in part by the National Science Foundation under award number CCF-0430349.

References

1. Alliez, P., Meyer, M., Desbrun, M.: Interactive geometry remeshing. ACM Trans. Graph. **21** (2002) 347–354 Proc. SIGGRAPH 2002.
2. Frey, P., Borouchaki, H.: Geometric surface mesh optimization. Comput. Visual. Sci. (1998) 113–121

3. Hormann, K., Labsik, U., Greiner, G.: Remeshing triangulated surfaces with optimal parametrizations. Comput. Aid. Des. **33** (2001) 779–788
4. Jiao, X., Colombi, A., Ni, X., Hart, J.: Anisotropic mesh adaptation for evolving triangulated surfaces. In: Proc. 15th International Meshing Roundtable. (2006)
5. Praun, E., Hoppe, H.: Spherical parametrization and remeshing. ACM Trans. Graph. **22** (2003) 340–349 Proc. SIGGRAPH 2003.
6. Surazhsky, V., Gotsman, C.: Explicit surface remeshing. In: Eurographics Symposium on Geometric Processing. (2003) 20–30
7. Donea, J., Huerta, A., Ponthot, J.P., Rodriguez-Ferran, A.: Arbitrary Lagrangian-Eulerian methods. In Stein, E., de Borst, R., Hughes, T.J., eds.: Encyclopedia of Computational Mechanics. Volume 1: Fundamentals. John Wiley (2004)
8. Garimella, R., Shashkov, M., Knupp, P.: Triangular and quadrilateral surface mesh quality optimization using local parametrization. Comput. Meth. Appl. Mech. Engrg. **193** (2004) 913–928
9. Hansen, G.A., Douglass, R.W., Zardecki, A.: Mesh Enhancement. Imperial College Press (2005)
10. Knupp, P., Margolin, L., Shashkov, M.: Reference Jacobian optimization based rezone strategies for arbitrary Lagrangian Eulerian methods. J. Comput. Phys. **176** (2002) 93–128
11. Shashkov, M.J., Knupp, P.M.: Optimization-based reference-matrix rezone strategies for arbitrary lagrangian-Eulerian methods on unstructured grids. In: Proc. the 10th International Meshing Roundtable. (2001) 167–176
12. Zhou, T., Shimada, K.: An angle-based approach to two-dimensional mesh smoothing. In: Proceedings of 9th International Meshing Roundtable. (2000) 373–384
13. Degener, P., Meseth, J., Klein, R.: An adaptable surface parameterization method. In: Proc. the 12th International Meshing Roundtable. (2003) 227–237
14. Knupp, P.: Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. part i: a framework for surface mesh optimization. Int. J. Numer. Meth. Engrg. **48** (2000) 401–420
15. Sheffer, A., de Sturler, E.: Parameterization of faceted surfaces for meshing using angle-based flattening. Engrg. Comput. **17** (2001) 326–337
16. Floater, M.S., Hormann, K.: Surface parameterization: a tutorial and survey. In: Advances in Multiresolution for Geometric Modelling. Springer Verlag (2005) 157–186
17. Desbrun, M., Meyer, M., Alliez, P.: Intrinsic parameterizations of surface meshes. In: Proc. Eurographics 2002 Conference. (2002)
18. Gotsman, C., Gu, X., Sheffer, A.: Fundamentals of spherical parameterization for 3d meshes. ACM Trans. Graph. **22** (2003) 358–363 Proc. SIGGRAPH 2003.
19. Hormann, K., Greiner, G.: Mips: An efficient global parametrization method. In Laurent, P.J., Sablonniere, P., Schumaker, L.L., eds.: Curve and Surface Design: Saint-Malo 1999. (2000) 153–162
20. Khodakovsky, A., Litke, N., Schröder, P.: Globally smooth parameterizations with low distortion. ACM Trans. Graph. **22** (2003) 350–357 Proc. SIGGRAPH 2003.
21. Kos, G., Varady, T.: Parameterizing complex triangular meshes. In Lyche, T., Mazure, M.L., Schumaker, L.L., eds.: Curve and Surface Design: Saint-Malo 2002, Modern Methods in Applied Mathematics. (2003) 265–274
22. Sheffer, A., Gotsman, C., Dyn, N.: Robust spherical parametrization of triangular meshes. In: Proc. the 4th Israel-Korea Bi-National Conference on Geometric Modeling and Computer Graphics. (2003) 94–99
23. Pinkall, U., Polthier, K.: Computing discrete minimal surfaces and their conjugates. Exper. Math. **2** (1993) 15–36
24. Jiao, X.: Face offsetting: a unified framework for explicit moving interfaces. J. Comput. Phys. **220** (2007) 612–625