Labeled parameterization for high-quality surface fitting

Conference Paper · January 2015

2 authors:

Márton Vaitkus
Budapest University of Technology and Economics

Tamas Varady
Budapest University of Technology and Economics

Labeled parameterization for high-quality surface fitting*

Márton Vaitkus[1] and Tamás Várady[2]

[1] Budapest University of Technology and Economics
s.high.hopes@gmail.com
[2] Budapest University of Technology and Economics
varady@iit.bme.hu

Abstract. Fitting of point clouds or triangle meshes with parametric surfaces is a fundamental task in many applications, including the reverse engineering of CAD models and geometric modeling with curve networks. These applications require that the fit must adhere to tight tolerances, should be sufficiently smooth, and should have a small amount of control points at the same time. This necessitates a complicated, iterative fitting algorithm, with many sensitive parameters. The initial parameterization of the data points tends to have a drastic effect on the quality of the fit - however, finding the optimal parameterization for the purposes of surface fitting in a completely automatic way is still a difficult, open problem, especially for surfaces representing trimmed patches with complex geometry. Thus, the currently known automatic parameterization methods might fail to give acceptable results for certain surfaces, forcing us to rely on user interaction. In this paper, we consider parameterization algorithms capable of respecting a labeling of the surface boundary provided by the user (mapping parts of the boundary to certain sides of the domain rectangle) - this is achieved by a variant of the As-Rigid-As-Possible parameterization method. We compare the final fit quality initialized with the output of our algorithm.

1 Motivation

In geometry processing applications, it is often required to fit a point cloud or a triangle mesh with a parametric surface, e.g. a tensor-product B-spline. A well-known example is the *reverse engineering* of CAD models [39], where an unorganized reconstruction of the model geometry (a point cloud or a triangle mesh) is first segmented into parts representing geometric primitives (plane, cylinder, etc.), procedurally generated parts (sweeps, surfaces of revolution, blends, etc.) and free-form patches; and later each part is fitted with a surface of its presumed type. Surfaces identified as free-form are to be represented as tensor-product B-splines, so their reconstruction naturally leads to the parametric surface fitting problem. The other major application area where surface fitting might be

* This work is a reduced version of a paper to be submitted elsewhere.

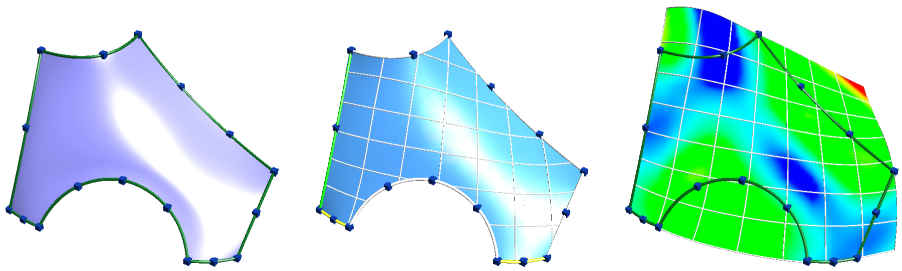
necessary is geometric modelling with curve networks - modern design methods employ n-sided surface patches (where $n \neq 4$, potentially) [40], requiring a non-standard representation which must be converted to traditional 4-sided tensor-product B-splines to ensure compatibility with common CAD/CAM software.

Our task then is to fit a set of points with a tensor product B-spline surface. With the number of control points fixed beforehand, this leads to a least-squares fitting problem - in practice, however, the optimal size of the control net, i.e. the optimal number of degrees of freedom is unknown. An additional difficulty - rarely addressed in published research on surface fitting - is that the majority of free-form surfaces arising in practice are best represented by *trimmed* patches, meaning the surface is assumed to be only a subset of some larger, unknown 4-sided surface, see Figure 1. So, the problem one must solve involves finding the parameter domain corresponding to the trimmed surface patch, choosing the number of control points and computing their positions in space, while adhering to several requirements:

- the data should be fitted with tight tolerance;
- the number of control points should be as small as possible;
- the surface should have a fair curvature distribution, without unexpected oscillations, and must extend naturally beyond the trimming curves;
- the trimmed out area should be as small as possible.

Finding a compromise between all these requirements is highly nontrivial in practice and the seemingly straightforward least-squares fitting procedure must be incorporated into a complicated iterative algorithm, with numerous sensitive parameters [43]. Given the large number and variety of published fitting algorithms we make no assumptions about how the fit will be computed, instead we aim at developing tools that could benefit any fitter.

One of the most important input parameters of any fitting algorithm is the choice of *initial parameterization*, which could have a drastic effect on the surface quality. For simple geometries one might get a sufficiently good parameterization by simply *projecting* to some *reference* or *carrier surface*, such as the least-squares plane, cylinder or even some simple parametric patch. In the case of more complex surfaces however, such methods might not give satisfactory results and a *flattening* of the mesh might be required - i.e. the vertices should be mapped to the $u - v$ parameter plane in an optimized way. Although the flattening of triangle meshes is a problem with a large and rich literature, the special needs and requirements of surface fitting have rarely been addressed before. In particular, finding a parameterization that is "optimal" for trimmed surface fitting in a completely automatic manner is a hard, open problem - at the time of writing this paper, we are not aware of any published work addressing it explicitly. As a starting point for future research on this topic, it should be noted that the usual objectives of mesh parameterization methods - minimizing the distortion and aligning the isolines with the principal curvatures - are not as universally beneficial for the purposes of free-form surface fitting as they tend to for those of other applications, such as texturing and quadrilateral remeshing. We do not attempt to solve this general problem in this paper, instead we try to alleviate the difficulty in finding the optimal parameterization of a trimmed surface by exploiting additional information provided by the user.



(a) (b) (c)

Fig. 1: Trimmed surface fitting and labeled parameterization. (a): Original surface. (b): Labels: west (green), south (yellow) and isolines. (c): Full fitted surface with mean curvature map.

In a process we call *labeling*, the user is asked to put labels (North, East, South, West) on parts of the surface boundary, suggesting that a given part is to be mapped to a certain side of the domain rectangle. Unlabeled parts are free to be optimized during the parameterization. The process is illustrated in Figure 1. We present a method for computing a parameterization which respects the constraints implied by labeling with small geometric distortion. Our constrained parameterization method is based on a two-step procedure: we first map the individual triangles to the plane simply by rotations; then we "stitch" these rotated triangles together into a consistent planar mesh. (Note: a similar labelling concept has been formerly developed in the reverse engineering system *Geomagic Studio* [1], however, the solution published in this paper is a new approach.)

In what follows, we first survey the relevant literature, then describe our two-step algorithm in detail. We describe how to enforce the constraints implied by the labeling and we demonstrate the effectiveness of our approach by examining the effect it has on the fitting of trimmed tensor product B-spline surfaces.

2 Related work

Initial parameterization for surface fitting. We do not attempt to give a survey of surface fitting algorithms, the interested reader is referred to Weiss et al. [43] and the references therein. In our experiments we use a variant of the iterative least-squares fitter described by Weiss et al. [43], but otherwise we make no assumptions about the way the surface is being fitted. As we have mentioned earlier, most methods for initial parameterization project the data points onto some simple surface - projecting onto a least-squares plane or a cylinder are popular choices in practice. A more sophisticated approach is described by Ma and Kruth [25], who project onto a rough parametric fit, constructed from the boundaries and interior section curves. Piegl and Tiller [28] project onto Coons patches or tensor product splines, depending on the complexity of the geometry. Neither of these methods are applicable to trimmed patches, as they assume that all 4 boundary curves are available. Pottmann and Leopoldseder [30] and Azariadis [2] both describe generalizations of the *active contours* idea in image processing, which means iteratively deforming a reference surface until it approximates the given data. Krishnamurthy et al. [18] optimize a set of isolines by minimizing a spring-like energy - while this is yet again restricted to 4-sided patches, a similar method potentially applicable to the trimmed case has been proposed by Wallner et al. [41]. Floater [12] parameterizes a point cloud with a meshless variant of Tutte-type mesh parameterization methods, with the same limitations regarding convex boundary curves. The use of mesh parameterization methods in the surface fitting context has been described in detail by Weiss et al. [43], but said work has preceded the development of modern free-boundary parameterization methods and the labeling problem is not addressed. While the method of Kós et al. [17] can be extended to handle labeling constraints for trimmed patches[1], it is still based on a fixed boundary scheme and its implementation is highly complex compared to the method we propose. More recently Lai et al. [19] and Wang et al. [42] parameterize triangle meshes in a feature-sensitive way to facilitate surface fitting - however the problem of trimming is not addressed.

Mesh parameterization. Mesh parameterization has a very extensive and rich literature - the unacquainted reader should first refer to the thorough surveys of Sheffer et al. [34] and Hormann et al. [15]. The simplest methods are based on Tutte's theorem on planar graphs [38], which allows us to compute a valid parameterization as long as the boundary curve is mapped to a fixed convex polygon [11,13]. While there are ways to generalize these methods to non-convex boundaries [20,17], the majority of modern parameterization methods allow the boundary curve to be optimized as part of the algorithm. A popular approach is to minimize the angular distortion, i.e. compute a *conformal map* by optimizing some functional of the positions [21,10], angles [33,45] or edge lengths [36,4], as this problem usually has a well-defined, unique solution easily found by solving a linear system. Optimizing some more general distortion measure tends to require expensive non-linear optimization [14,31] - an important exception is As-Rigid-As-Possible (ARAP) parameterization [24], which employs an iterative two-phase optimization scheme, and thus directly inspired the constrained parameterization method described in this paper. Our approach is also very closely related to vector field design [9] and field-aligned global parameterization algorithms [16,6,27].

[1] Personal communication of Géza Kós.

3 Constrained parameterization algorithm

The parameterization methods we propose can be considered as a variant of ARAP parameterization or more precisely the local-global iterative algorithm for finding the local minimum of the ARAP energy, first proposed by Sorkine and Alexa [35]. The basic observation is that a parameterization is - up to translation - nothing else but a collection of linear maps, representable by 2×2 matrices (Jacobians) in arbitrary per-triangle reference frames. When the parameterization is without distortion, i.e. it is isometric, all of these maps are rigid transformations (rotation matrices); and thus a natural measure of distortion is the "distance" (e.g. in Frobenius norm) between the map and the rotation closest to it. This suggests the following two-step method for computing the parameterization (see Figure 2):

1. Compute a set of rigid transformations for the triangles (disregarding their connectivity)
2. Stitch together the rotated triangles triangles into a valid planar mesh

We will see that a great advantage of this method is that we can enforce constraints (e.g. those arising from a labeling) in a natural manner.

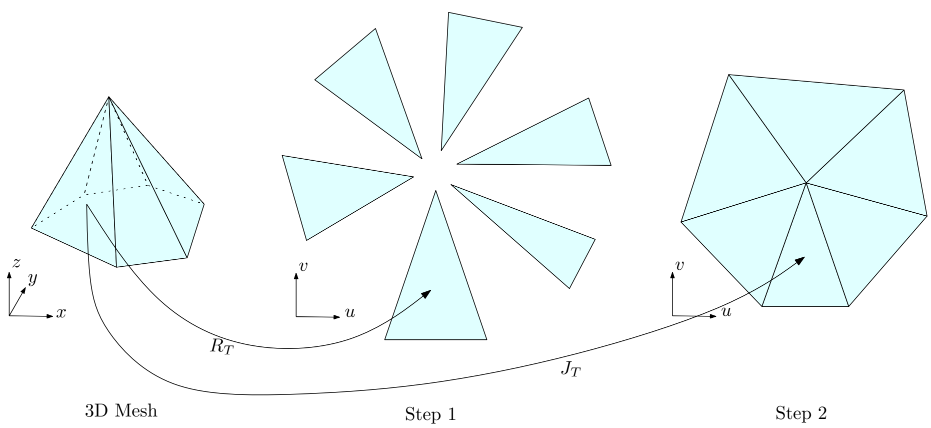


Fig. 2: Illustration of our parameterization method for a triangle fan. Step 1: the 3D triangles are flattened independently in the $u - v$ plane with pure rotations R_T. Step 2: the rotations are "stitched" together in a least-squares manner to give the Jacobians J_T of a planar mesh. See also Section 3.2.

3.1 Computing the rotation matrices

Imagine that we flatten the mesh isometrically, face-by-face, by placing a single arbitrary triangle on the plane and then traversing a dual spanning tree of the triangles, flattening each new triangle in the obvious way. Such a simple procedure will obviously fail to provide a natural layout in the plane, due to the fact that most surfaces have non-zero Gaussian curvature, which for a triangle mesh means that around vertices the triangle fans might have non-zero angle defect. This already suggests a solution to the problem: instead of simply flattening the triangles along the edges, apply additional rotations, in a way that distributes the angle defect evenly around each vertex (and around each homology generator for multiply connected surfaces). This leads to an underdetermined set of linear equations $F\omega = K$ for the rotation angles ω, where F is the vertex-edge adjacency matrix and K is the vector of Gaussian curvatures at the vertices[2], which we want solve in a least-norm manner:

$$\begin{array}{l} \underset{\omega}{\text{minimize}} \quad \|\omega\|^2 \\ \text{subject to } F\omega = K \end{array}.$$

This is a standard optimization problem [7], which can be solved e.g. by sparse QR factorization. Note that the method we described is practically equivalent to the so-called *discrete connection* method for vector field design on discrete surfaces [9]. Given the additional rotations, the rotation matrices are computed by traversing the faces as described above.

Enforcing constraints During this phase of the parameterization, we can obviously enforce any constraint which can be expressed as a set of linear equations for the rotation angles. For labeling, we might either need two adjacent edges to make a 0 or 90 degree angle or we might want two distant edges to do the same. The first case simply means adding an extra equation for the boundary vertex, while the latter is equivalent to an equation involving an arbitrary dual path between the two edges. To get a proper global orientation, the first triangle (the root of the spanning tree) must contain one of the labelled edges and must be placed on the plane according to its assigned label. Also, while redundant constraints might simplify the implementation, they are to be avoided in practice as they often become contradictory due to floating point errors.

3.2 Computing the planar mesh from the rotations

Given the set of rotation matrices, we wish to compute a parameterization, which has as its Jacobians, matrices that are close to the rotations as possible in Frobenius norm. This means solving the following optimization problem for the vertex parameters u and v:

$$\underset{u,v}{\text{minimize}} \sum_T \|J_T(u,v) - R_T\|_F^2 .$$

[2] For multiply connected surfaces, the system also includes equations corresponding to inner boundary loops.

Solving this optimization problem directly leads to a pair of linear systems, equivalent to discrete Poisson equations (see [8] or [3]):

$$\begin{aligned} Lu &= b_u \\ Lv &= b_v, \end{aligned}$$

where L is the well-known cotangent Laplacian [26,29] and b_u and b_v are the discrete divergences of the columns of the Jacobian matrices - see [24] for explicit formulas.

Enforcing constraints As we minimize a quadratic form of the positions, we can enforce any linear constraint by Lagrange multipliers. For labeling we could simply require that the u and v coordinates of vertices lying on the same sides of the domain rectangle to be the same. This is somewhat redundant, however, as the only constraints which are not satisfied already for the rotations, are that disjoint segments with the same label should lie on the same line in the plane. Thus a more natural alternative is to map the labeled edges with *exactly* the previously computed rotations, while allowing them to be scaled by an arbitrary factor. This means adding equations of the following form:

$$J_T|_e = \alpha_T R_T,$$

i.e. that the Jacobian J_T restricted to the labeled edge e is some (unknown) scalar α_T times the rotation R_T. We regularize the scaling factors by augmenting the minimized energy with terms $|\alpha_T - 1|^2$. Now, to force disjoint segments to lie on the same line, we only have to require the equality of the u or v coordinates of a single vertex from each segment. In any case, the problem is of the form

$$\begin{aligned} &\underset{x}{\text{minimize}} \;\; \tfrac{1}{2}x^{\mathrm{T}}Ax - b^{\mathrm{T}}x \\ &\text{subject to } Cx = d \end{aligned},$$

which is solved by the linear system

$$\begin{bmatrix} A & C^{\mathrm{T}} \\ C & 0 \end{bmatrix} \begin{bmatrix} x \\ \nu \end{bmatrix} = \begin{bmatrix} b \\ d \end{bmatrix},$$

where ν is a vector of Lagrange multipliers. One might object, that we could simply constrain the labeled vertices to have the same u or v coordinates, without even enforcing any constraints in the first phase. While this is indeed possible, our method can be considered as a priori more robust then this naive approach: by first computing a natural rigid layout for the triangles, enforcing the constraints require less drastic deviation from the rotations, which leads to smaller distortion and a reduced chance of triangle flips. Note that instead of trying to be close to the rotation matrices, we could in principle minimize *any* quadratic functional of the positions in this phase.

3.3 Ensuring local injectivity

A significant drawback of using the cotangent Laplacian for stitching the triangles together is that the triangles might reverse their orientation, i.e. the parameterization might not be *locally injective*, which is quite common in the presence of constraints - this is a limitation our method shares with the majority of published mesh parameterization algorithms. As the corresponding constraints are non-convex functions of the unknowns, we solve the problem heuristically using an iterative weighting scheme: if a triangle happens to flip, we give it a larger weight in the energy, and we iterate in this manner until all flips are resolved (for practical examples this happens within no more than a dozen or so iterations). While more robust alternatives have appeared in recent works [22,5,32], these usually require potentially expensive inequality-constrained optimization. A practically identical scheme is applied to the scaling factors, as they are also not guaranteed to be positive, unless we add explicit inequality constraints to the problem.

4 Results

We compare the fits computed by a black-box solver (based on the method of [43]) with a built-in projective parameterization and our own method for trimmed surfaces. Figure 3 shows results (parameterization, curvature and deviation maps) for the surface of Figure 1 for the projective parameterization (first row) and two different kinds of labeling (second and third rows). Figure 4 shows results for another test surface. The curvature maps indicate mean curvatures, the isolines of the surfaces have been superimposed. Observe how isolines get aligned to the prescribed labels. The deviation maps shows various colored areas, indicating 'within tolerance' (green), negative (blue) and positive (red) distances. The control nets of the surfaces have also been added.

In general, the labeled parameterization yields surface approximations, which have better curvature distribution (i.e. the surfaces are *fairer*). This may increase the approximation error to a minimal extent compared to the projective method, though this is a negligible trade-off, as the fairness of the final surface is of utmost importance. The surfaces beyond the trimmed areas should extend smoothly and naturally, being a crucial issue when further computations for intersections, fillet generations, offsetting, etc. need to be performed.

5 Limitations and Future Work

Besides the ad-hoc way we enforce local injectivity, an important limitation is that we cannot guarantee the *global* injectivity of the mapping, i.e. parts of the mesh might overlap, even with every triangle having the right orientation. While we have not yet observed this in the context of trimmed surface fitting, it might easily happen for surfaces with high curvature, complex boundary and disjoint segments getting the same label. Ensuring global injectivity is a much

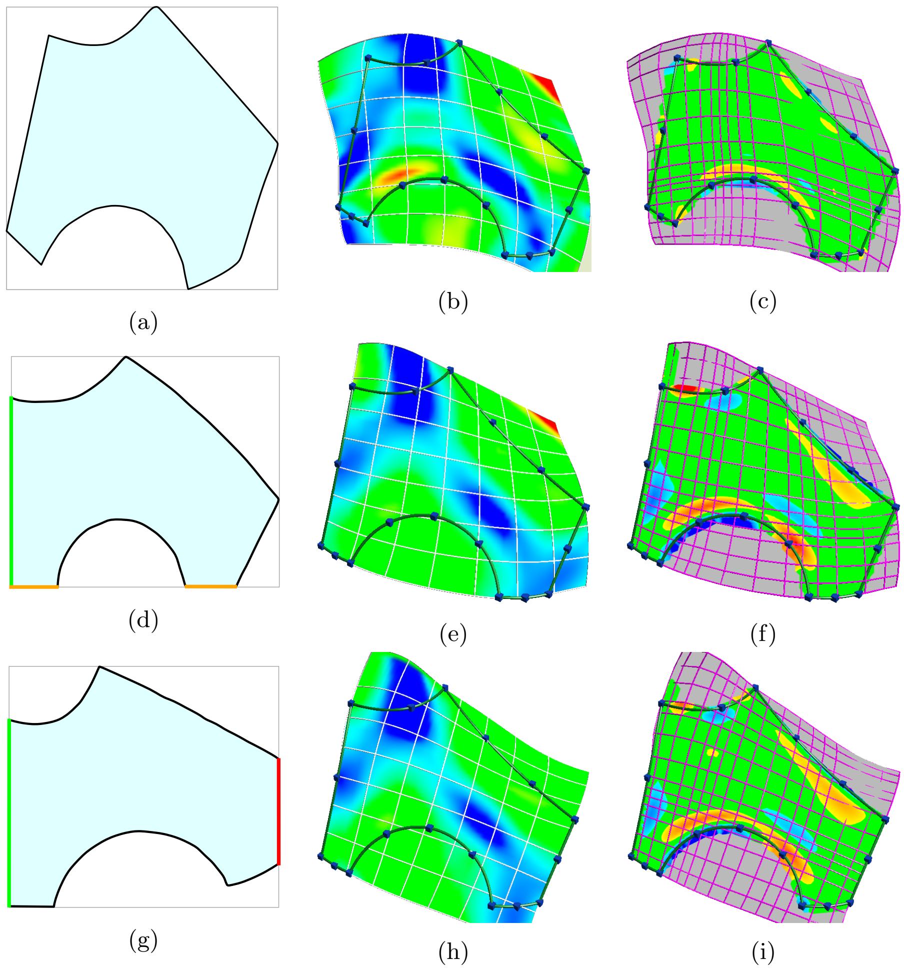


Fig. 3: Test example shown in Figure 1. First row: projective parameterization. Second row: Labels - west and south. Third row: Labels - west and east.

more difficult problem than ensuring local injectivity - at the time of writing, published methods only guarantee global bijectivity if the boundary of the domain is fixed [23] - for general, optimized boundaries the problem appears to be wide open. In our case one might attempt at an imperfect solution (which would nonetheless cover most practical situations) by prohibiting disjoint segments with the same label to intersect via simple linear inequalities. We also cannot guarantee that the entire mesh lies within the rectangle defined by the labeled boundaries - an example of this can be seen in Figure 4(e). This is even less likely to cause a serious problem, but its solution is very simple if we can enforce inequality constraints. While there are highly efficient general-purpose

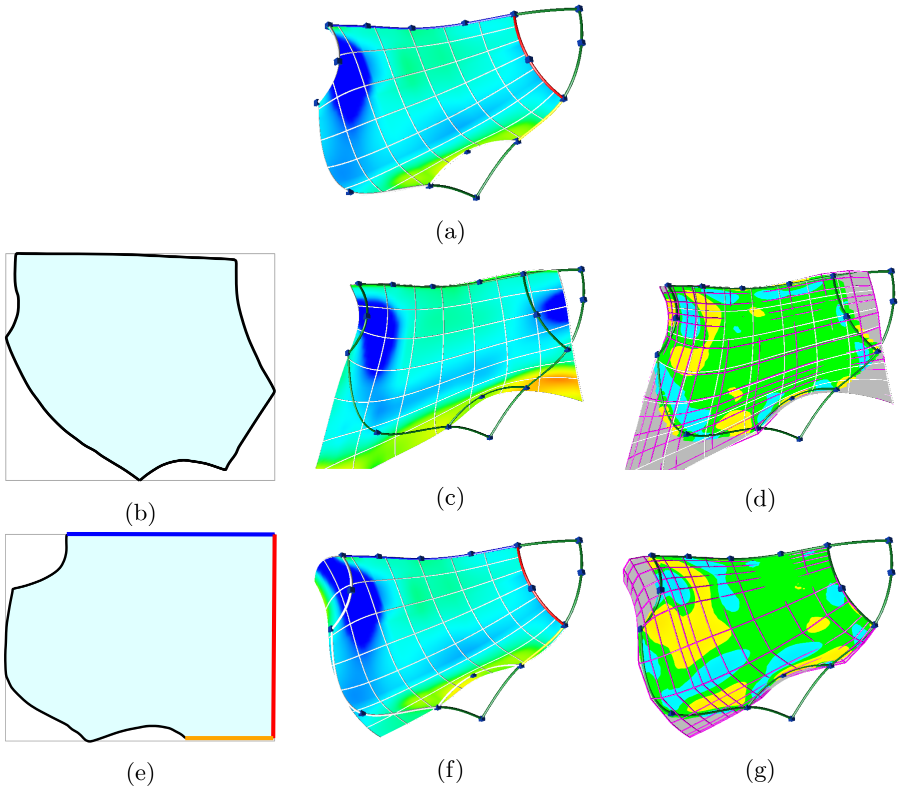


Fig. 4: Test example for a trimmed surface. (a): Original surface with labeling. (b)-(d): projective parameterization. (e)-(g): Labels - north, east and south.

solvers for inequality-constrained convex and even non-convex optimization, recently published results suggest that geometric problems might benefit greatly from customized solvers exploiting their peculiar structure [32].

We only compute a single parameterization, by minimizing some kind of geometric distortion. The user might prefer a parameterization that has higher distortion but fills a larger percentage of the domain rectangle or maybe satisfies other application-specific requirements. Thus, it might be worthwhile to adapt recent results in shape-space exploration [44] and constrained modelling [37] to allow the user more freedom in choosing the parameterization, without sacrificing efficiency.

Finally, guessing a labeling without user assistance and defining what makes a parameterization "good" for surface fitting are challenging avenues for future research.

6 Conclusions

We have presented a method for computing a parameterization of a triangle mesh, useful for fitting the mesh with a trimmed tensor-product B-spline surface. The parameterization respects a boundary labeling provided by the user. The algorithm is based on a two-step iterative procedure, which is capable of enforcing the constraints implied by the labeling, with negligible additional cost over the unconstrained case. The parameterizations have been used to initialize a least-squares fitting algorithm and resulted in fits that are more natural and of higher quality than those computed from traditional projective parameterizations.

Acknowledgements

The authors would like to thank György Karikó for helping us interfacing with a surface fitter. This project is supported by the Hungarian Scientific Research Fund [OTKA, No. 101845].

References

1. www.geomagic.com. 3
2. P. N. Azariadis. Parameterization of clouds of unorganized points using dynamic base surfaces. *Computer-Aided Design*, 36(7):607–623, 2004. 4
3. J. A. Bærentzen, J. Gravesen, F. Anton, and H. Aanæs. *Guide to Computational Geometry Processing: Foundations, Algorithms, and Methods*. Springer, 2012. 7
4. M. Ben-Chen, C. Gotsman, and G. Bunin. Conformal flattening by curvature prescription and metric scaling. In *Computer Graphics Forum*, volume 27, pages 449–458. Wiley Online Library, 2008. 4
5. D. Bommes, M. Campen, H.-C. Ebke, P. Alliez, and L. Kobbelt. Integer-grid Maps for Reliable Quad Meshing. *ACM Trans. Graph.*, 32(4):98:1–98:12, July 2013. 8
6. D. Bommes, H. Zimmer, and L. Kobbelt. Mixed-integer quadrangulation. In *ACM Transactions on Graphics (TOG)*, volume 28, page 77. ACM, 2009. 4
7. S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004. 6
8. R. Courant and M. Schiffer. *Dirichlet's principle, conformal mapping, and minimal surfaces*, volume 3 of *Pure and Applied Mathematics*. Interscience Publishers, 1950. 7
9. K. Crane, M. Desbrun, and P. Schröder. Trivial connections on discrete surfaces. In *Computer Graphics Forum*, volume 29, pages 1525–1533. Wiley Online Library, 2010. 4, 6
10. M. Desbrun, M. Meyer, and P. Alliez. Intrinsic parameterizations of surface meshes. In *Computer Graphics Forum*, volume 21, pages 209–218. Wiley Online Library, 2003. 4
11. M. S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231–250, 1997. 4
12. M. S. Floater. Meshless parameterization and B-spline surface approximation. In *The Mathematics of Surfaces IX*, pages 1–18. Springer, 2000. 4

13. M. S. Floater. Mean value coordinates. *Computer Aided Geometric Design*, 20(1):19–27, 2003. 4
14. K. Hormann and G. Greiner. MIPS: An Efficient Global Parametrization Method. *Curve and Surface Design*, page 153, 2000. 4
15. K. Hormann, B. Lévy, and A. Sheffer. Mesh parameterization: Theory and practice. 2007. SIGGRAPH Course Notes. 4
16. F. Kälberer, M. Nieser, and K. Polthier. QuadCover-Surface Parameterization using Branched Coverings. In *Computer Graphics Forum*, volume 26, pages 375–384. Wiley Online Library, 2007. 4
17. G. Kós and T. Várady. Parameterizing complex triangular meshes. In T. Lyche, M.-L. Mazure, and L. L. Schumaker, editors, *Curve and surface design*, pages 265–274. Nashboro Press, 2003. 4
18. V. Krishnamurthy and M. Levoy. Fitting smooth surfaces to dense polygon meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 313–324. ACM, 1996. 4
19. Y.-K. Lai, S.-M. Hu, and H. Pottmann. Surface fitting based on a feature sensitive parametrization. *Computer-Aided Design*, 38(7):800–807, 2006. 4
20. Y. Lee, H. S. Kim, and S. Lee. Mesh parameterization with a virtual boundary. *Computers & Graphics*, 26(5):677–686, 2002. 4
21. B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 362–371. ACM, 2002. 4
22. Y. Lipman. Bounded distortion mapping spaces for triangular meshes. *ACM Transactions on Graphics (TOG)*, 31(4):108, 2012. 8
23. Y. Lipman. Bijective mappings of meshes with boundary and the degree in mesh processing. *SIAM Journal on Imaging Sciences*, 7(2):1263–1283, 2014. 9
24. L. Liu, L. Zhang, Y. Xu, C. Gotsman, and S. J. Gortler. A local/global approach to mesh parameterization. In *Computer Graphics Forum*, volume 27, pages 1495–1504. Wiley Online Library, 2008. 4, 7
25. W. Ma and J.-P. Kruth. Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces. *Computer-Aided Design*, 27(9):663–675, 1995. 4
26. R. H. MacNeal. *The solution of partial differential equations by means of electrical networks*. PhD thesis, California Institute of Technology, 1949. 7
27. A. Myles and D. Zorin. Global parametrization by incremental flattening. *ACM Transactions on Graphics (TOG)*, 31(4):109, 2012. 4
28. L. A. Piegl and W. Tiller. Parametrization for surface fitting in reverse engineering. *Computer-Aided Design*, 33(8):593–603, 2001. 4
29. U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics*, 2(1):15–36, 1993. 7
30. H. Pottmann and S. Leopoldseder. A concept for parametric surface fitting which avoids the parametrization problem. *Computer Aided Geometric Design*, 20(6):343–362, 2003. 4
31. P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe. Texture mapping progressive meshes. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 409–416. ACM, 2001. 4
32. C. Schüller, L. Kavan, D. Panozzo, and O. Sorkine-Hornung. Locally Injective Mappings. *Computer Graphics Forum*, 32(5):125–135, 2013. 8, 10
33. A. Sheffer and E. de Sturler. Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers*, 17(3):326–337, 2001. 4

34. A. Sheffer, E. Praun, and K. Rose. Mesh parameterization methods and their applications. *Foundations and Trends® in Computer Graphics and Vision*, 2(2):105–171, 2006. 4
35. O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *ACM International Conference Proceeding Series*, volume 257, pages 109–116. Citeseer, 2007. 5
36. B. Springborn, P. Schröder, and U. Pinkall. Conformal equivalence of triangle meshes. *ACM Transactions on Graphics (TOG)*, 27(3):77, 2008. 4
37. C. Tang, X. Sun, A. Gomes, J. Wallner, and H. Pottmann. Form-finding with polyhedral meshes made simple. *ACM Transactions on Graphics (TOG)*, 33(4):70, 2014. 10
38. W. Tutte. How to Draw a Graph. *Proceedings of the London Mathematical Society*, 3(1):743–767, 1963. 4
39. T. Varady and R. Martin. Reverse engineering. *Handbook of Computer Aided Geometric Design*, pages 651–681, 2002. 1
40. T. Várady, A. Rockwood, and P. Salvi. Transfinite surface interpolation over irregular n-sided domains. *Computer-Aided Design*, 43(11):1330–1340, 2011. 2
41. J. Wallner, H. Pottmann, and M. Hofer. Fair webs. *The Visual Computer*, 23(1):83–94, 2007. 4
42. Y. Wang and J. Zheng. Curvature-guided adaptive T-spline surface fitting. *Computer-Aided Design*, 45(8):1095–1107, 2013. 4
43. V. Weiss, L. Andor, G. Renner, and T. Várady. Advanced surface fitting techniques. *Computer Aided Geometric Design*, 19(1):19–42, 2002. 2, 3, 4, 8
44. Y.-L. Yang, Y.-J. Yang, H. Pottmann, and N. J. Mitra. Shape space exploration of constrained meshes. *ACM Trans. Graph.*, 30(6):124, 2011. 10
45. R. Zayer, B. Lévy, H.-P. Seidel, et al. Linear angle based parameterization. In *Fifth Eurographics Symposium on Geometry Processing-SGP 2007*, pages 135–141, 2007. 4