Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Measured boundary parameterization based on Poisson's equation[*]

Jun-jie CAO, Zhi-xun SU[†‡], Xiu-ping LIU, Hai-chuan BI
(*School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, China*)
[†]E-mail: zxsu@hotmail.com

Received July 27, 2009; Revision accepted Nov. 11, 2009; Crosschecked Dec. 30, 2009

Abstract: One major goal of mesh parameterization is to minimize the conformal distortion. Measured boundary parameterizations focus on lowering the distortion by setting the boundary free with the help of distance from a center vertex to all the boundary vertices. Hence these parameterizations strongly depend on the determination of the center vertex. In this paper, we introduce two methods to determine the center vertex automatically. Both of them can be used as necessary supplements to the existing measured boundary methods to minimize the common artifacts as a result of the obscure choice of the center vertex. In addition, we propose a simple and fast measured boundary parameterization method based on the Poisson's equation. Our new approach generates less conformal distortion than the fixed boundary methods. It also generates more regular domain boundaries than other measured boundary methods. Moreover, it offers a good tradeoff between computation costs and conformal distortion compared with the fast and robust angle based flattening (ABF++).

Key words: Mesh parameterization, Poisson's equation, Measured boundary
doi:10.1631/jzus.C0910460 **Document code:** A **CLC number:** TP391.41

1 Introduction

Surface parameterization is a fundamental and vital issue in science and engineering. It facilitates most forms of surface processing, such as texture map, morphing, remeshing, surface reconstruction, and surface editing. At the same time, these applications foster the growth of efficient parameterization techniques that can produce less parameterization distortion and also run faster.

Gaussian curvature and the boundary map are two major concerns when seeking parameterization with less distortion. There are three techniques to minimize the Gaussian curvature: cutting the mesh into disk-like charts, introducing seams (Sheffer and Hart, 2002; Zhu *et al*., 2003), and recurring to cone singularities (Kharevych *et al*., 2006; Ben-Chen *et al*., 2008; Springborn *et al*., 2008; Yang *et al*., 2008). As to the boundary map, a common belief is that setting the boundary free absorbs serious distortion caused by convex boundary condition. There are three approaches to setting the boundary free, which are free boundary parameterization, virtual boundary parameterization, and measured boundary parameterization defined in Section 3. Among the three methods, the free boundary parameterization, taking the boundary map as a part of a whole optimization, often leads to the least distortion at the price of longer running time, and the measured boundary parameterization, separating the optimization of finding a better boundary map from the whole optimization, is easier to implement and more efficient with little higher distortion than the free boundary methods.

In this paper, we define a Poisson distance map (PDM) and a boundary Poisson distance map (BPDM) based on the Poisson's equation. With the two newly defined distance map, a novel measured boundary method is presented. We also discuss the choice of center vertex, which plays an important role in all

[‡] Corresponding author

[*] Project supported by the Program for New Century Excellent Talents in University (No. NCET-05-0275), the National Natural Science Foundation of China (Nos. 60533060 and 60673006), and the IDeA Network of Biomedical Research Excellence Grant (No. 5P20RR 01647206) from NIH, USA

measured boundary methods. The main process and results of our method are shown in Fig. 1. Our algorithm is as follows:

1. Find the center vertex by boundary geodesic distance map (BGDM) or BPDM.

2. Compute PDM from the center vertex to all boundary vertices.

3. Compute the boundary map according to the ‘distance’ computed.

4. Parameterize the remaining vertices by a discrete conform parameterization (DCP).

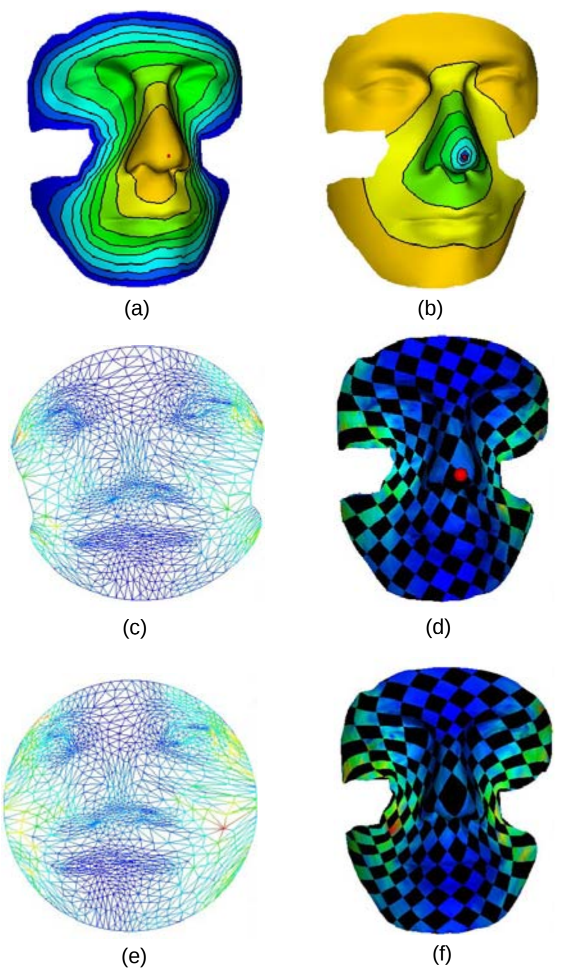

Fig. 1 Overview of our method
(a) shows the boundary Poisson distance map (BPDM) and the center vertex found; (b) shows the Poisson distance map (PDM) from the center vertex; (c) and (d) are the parameter domain and texture mapping of our method; (e) and (f) are those of discrete conform parameterization (DCP). The conformal distortion and L_2 stretch distortion of our method are 0.4493 and 1.3896, respectively, and those of DCP are 0.7778 and 1.6753, respectively

2 Related works and contribution

2.1 Mesh parameterization

Overwhelming algorithms are devoted to surface parameterization in the last decade, targeting diverse parameter domains and focusing on minimizing different distortions. We refer readers to Floater and Hormann (2005), Sheffer *et al*. (2006) and Hormann *et al*. (2007) for surveys about theories and recent advances. Here we briefly discuss some later works mostly related to our contribution.

Free boundary techniques treat all inner and boundary vertices in the same way, and obtain the boundary map and the mesh parameterization simultaneously. The more ‘natural’ (respecting the 3D boundary) the 2D boundary is, the less the distortion of the parameterization is introduced. Yet, more computations are often needed to derive the boundary fully free. Thus, unlike fast and easy-implement fixed boundary methods, free boundary techniques generally produce significantly less distortion at the price of a higher computational effort. In the last decade, there are many researches devoted to how to reduce distortion by setting the boundary free (Hormann and Greiner, 1999; Sheffer and de Sturler, 2000; Desbrun *et al*., 2002; Lévy *et al*., 2002; Karni *et al*., 2005; Zayer *et al*., 2005; Dong and Garland, 2007; Lee and Lee, 2007). Most of the above approaches are non-linear. Linear methods (Desbrun *et al*., 2002; Lévy *et al*., 2002) may lead to artifacts for some obscure choice of point constraints in practice, especially for meshes with geometrically complex boundaries. Mullen *et al*. (2008) overcame this problem by finding an eigenvector rather than solving a linear system with point constraints. The most recent linear free boundary method was proposed by Liu *et al*. (2008). They utilized a local/global approach to parameterize a mesh patch and claimed that their approach generates more shape-preserving results than other state-of-the-art methods.

Virtual boundary parameterization methods, first proposed by Lee *et al*. (2002), relax the 2D boundary in a less flexible way. It can also be seen as a fixed boundary method with a better boundary map. Thus, the computational cost is far less than that of the free boundary methods. Lee *et al*. (2002) used edge adjustment to parameterize the boundary in a more ‘natural’ way, and constructed a virtual boundary to guarantee the validity of the parameterization. But as they have declared, the effect is poor when the 3D boundary severely runs in and out. A simplified version of virtual boundary is often conducted as a post-processing for free boundary methods and measured boundary methods to prevent flipped triangles in parameter domain.

Measured boundary parameterization methods compute the boundary map before parameterization. The boundary is mapped onto a polygon using some kind of distance field from a center vertex (Lee *et al*., 2006; Lee and Lee, 2007; Shapira and Shamir, 2008). The distortion and computation time generated by the measured boundary parameterization methods are generally a trade-off between fixed boundary and free boundary methods.

Shapira and Shamir (2008) introduced a method named 'local geodesic parameterization', which respects the geodesic distances (calculated by a modified fast marching algorithm) from a specified source vertex to its geodesic neighboring vertices in a specified radius. The resulting 2D boundary is a kind of measured boundary. The algorithm lowers conformal distortion dramatically. They also presented an application: mean-shift on manifolds, where preserving the local geodesic distances is a key constraint. As they declared, their further work is to constrain the source vertex to stay exactly in the middle of the patch, which can reduce distortion even more.

To make the boundary more 'natural', Lee *et al*. (2006) and Lee and Lee (2007) computed the straightest geodesic distance as a better approximation of the geodesic distance. Distance from an interior source vertex to every boundary vertex was generated by cutting planes, which defines the straightest geodesic distance map. Then they parameterized the mesh's boundary by a polar map with angles and distances determined by the source vertex and every border vertex. They used the phrase measured boundary to name the 2D boundary generated. They showed that the method produces more competitive results than mean value parameterization (Floater, 2003). Their method has two serious problems. One is the resulting path from the source vertex may not always reach the boundary vertex if there is a hole or gap between them, or the path forms a closed cycle, as shown in Fig. 2. Even it reaches the boundary vertex, as shown in Fig. 3, they may introduce a large error. The other is that they declared that the source vertex should be a center vertex of the mesh from the boundary vertices, whilst they did not mention how it is determined.

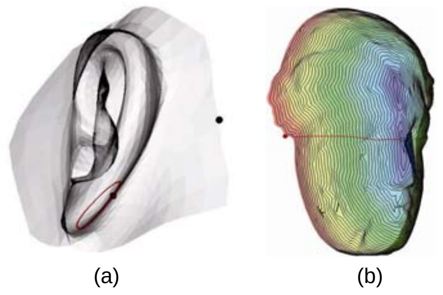
Fig. 2 Two failed cases in the generation of the straightest geodesic distance map: (a) The path is self-closed; (b) The path failed to reach the destination

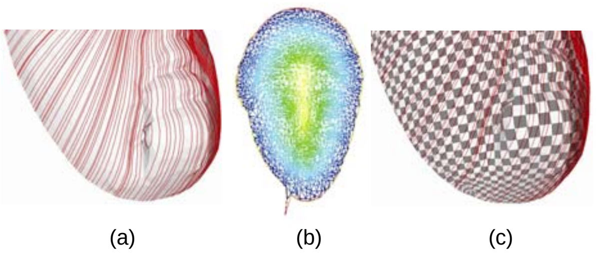
Fig. 3 A bad case of parameterization using the straightest geodesic distance map: (a) shows the straightest geodesic path; (b) is the resulting parameter domain; (c) is the corresponding texture mapping

2.2 Geodesic path in mesh processing

Just like surface parameterization, calculating geodesic paths and distances on meshes is also a fundamental problem and various applications benefit from it, such as parameterization (Lee *et al*., 2005; Shapira and Shamir, 2008), mesh flattening (Wang *et al*., 2004), mesh skeleton extraction (Tierny et al., 2006; Aujay *et al*., 2007), and remeshing (Sifri *et al*., 2003).

There are different definitions of geodesic path for a polyhedral surface. Considering a geodesic as a shortest path between two vertices on a graph corresponding to the mesh is maybe the most widely accepted definition. Dijkstra's world-famous algorithm is classified by this definition. There are also many Dijkstra-like algorithms generating better results, such as fast marching (Kimmel and Sethian, 1998). A detailed survey can be seen in Mitchell (2000).

Polthier and Schmies (2006) defined the straightest geodesics inspired by the quality of smooth geodesics. Their straightest geodesic path was uniquely defined with a source vertex and direction. The algorithm failed to compute the straightest geodesic with boundary condition, i.e., a source and a destination. For this, the first algorithm to compute the straightest path with boundary conditions on open mesh using a cutting plane was introduced in Lee *et al*. (2006). But as stated in the previous section, the path satisfying the boundary condition may not exist at all.

Methods above deal with the single source geodesic path problem. Wang *et al*. (2004) defined a boundary geodesic distance map (BGDM), which encapsulates the geodesic distance from every node to the boundary of a mesh, using a front propagation method. The algorithm deals with both uniform and non-uniform meshes. They used its gradient field as a guidance field to minimize the length of cutting paths to the mesh boundary.

2.3 Poisson's equation in mesh processing

To our knowledge, the early works in computer graphics using the Poisson's equation are Pérez *et al*. (2003) and Yu *et al*. (2004). Pérez *et al*. (2003) successfully applied the Poisson's equation to approximate guidance fields with boundary conditions in image editing. However, to generalize the idea directly to mesh setting is nontrivial. Yu *et al*. (2004) presented several necessary techniques to make the Poisson's equation as a mesh processing engine.

Inspired by them, the Poisson's equation has become imperative in mesh processing. It has been successfully applied in image compression (Fattal *et al*., 2002), mesh optimization (Nealen *et al*., 2006), deformation (Zhou *et al*., 2005), reconstruction (Kazhdan *et al*., 2006), remeshing (Dong *et al*., 2005) and mesh parameterization (Zayer *et al*., 2005; Dong and Garland, 2007; Ben-Chen *et al*., 2008). Amongst these, our PDM and BPDM are most motivated by Dong *et al*. (2005), who constructed a Poisson field, by solving a Poisson's equation with the cotangent weights and the magnitude of mean curvatures of the mesh, to place extrema. This is commonly obtained by a geodesic distance map (GDM) or some combinations based on it, while our PDM and BPDM use different weights, right-hand side and number of constrained vertices. Also, we have to mention that the choice of weights has outstanding influences on the resulting map.

2.4 Contribution

Our contribution is twofold. Firstly, given a center vertex, we propose a competitive measured boundary parameterization using PDM. Although PDM has been proposed as a way to simulate a GDM in the context of feature vertices extraction (Dong and Garland, 2007), we demonstrate that a regular domain boundary shape can be derived by using PDM at the cost of a slightly higher distortion than putting the boundary freer by GDM.

Secondly, we present a good supplement to the existing measured boundary methods. As is evident from Fig. 4, the parameterization highly depends on the choice of the center vertex. A different center vertex leads to a quite different parameter domain shape and conformal distortion. Though it plays an important role in all measured boundary approaches, existing measured boundary methods determine it in an obscure way. This paper gives two algorithms of computation: the BGDM defined in Wang *et al*. (2004), and the BPDM we proposed. Both of them are robust under different mesh resolutions and sampling rates. Moreover, the BPDM respects the symmetry of the mesh patches more than the BGDM in the choice of the center vertex. If the BPDM is employed, it often introduces less distortion than the BGDM. In addition to this, the parameterization process in succession can be speeded up by reusing the matrix factorization generated when computing the BPDM.

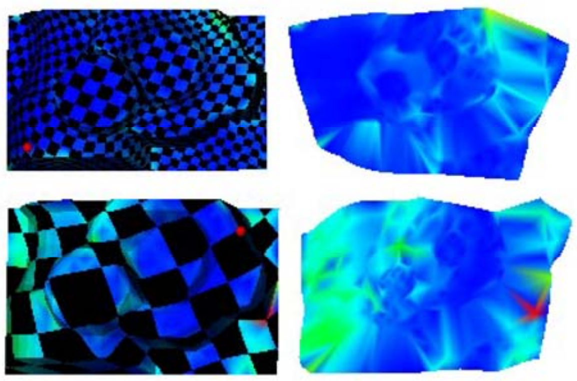

Fig. 4 Local geodesic parameterizations of the Stamp model with different center vertices
The color indicates the conformal distortion per vertex

3 Poisson distance map and boundary Poisson distance map

3.1 Poisson distance from the constrained vertices

When seeking to construct a smoother distance field on a mesh patch compared to the geodesic distance field in some aspects, the Poisson's equation comes under our attention inspired by Dong *et al*. (2005). Let $M=(V, F)$ be a triangulated mesh patch, composed of a set of vertices V and triangles F. C is a subset of V, called the constrained vertices. N_i represents the set of one-ring neighboring vertices of v_i, and v_i is the ith vertex. To find such a field, we solve the following Poisson's equation with the Dirichlet boundary conditions:

$$\begin{cases} \Delta f(v_i) = -g(v_i), & v_i \in V \setminus C, \\ f(v_i) = 0, & \text{otherwise}, \end{cases} \tag{1}$$

where Δ is the Laplacian operator and $g(v_i)$ represents the mean spoke length of v_i:

$$g(v_i) = \frac{1}{|N_i|} \sum_{j\in N_i} \|v_j - v_i\|.$$

Given a scalar function f defined on the mesh, its discrete Laplacian operator at a vertex v_i has the form

$$\Delta f(v_i) = \sum_{j\in N_i} \omega_{ij} \left(f(v_j) - f(v_i) \right), \tag{2}$$

where ω_{ij} are edge weights such that $\sum_{j\in N_i} \omega_{ij} = 1$. The solution of the equation assigns each vertex of the mesh a Poisson distance from the constrained vertices. As stated in Dong *et al*. (2005), assuming that the right-hand side is not always zero, the system has a unique solution even when only one constraint is specified (see a short proof in the Appendix).

We approximate the geodesic distance with the Poisson distance defined above for that they share two common and vital properties: the global minimum points must be the constrained vertices (i.e., source vertices in the literature about geodesic distance), and the distance is strictly positive for other vertices. Dong *et al*. (2005) showed that if all ω_{ij} are strictly positive, global and local minimum points are always the constrained vertices (ignoring numerical issues). As a consequence, the field generated is strictly positive everywhere on the mesh when the constraint values are zero. Thus, we can take it as a distance field defined on the mesh, and apply it to some applications where the geodesic distance is originally employed.

3.2 Poisson distance map and boundary Poisson distance map

3.2.1 Poisson distance map

If C contains only one vertex, the solution of the system Eq. (1), with the spring weights as the edge weights, is the Poisson distance from the vertex to other vertices, which defines the PDM on the mesh. The distance can be seen as an approximation of the geodesic distance from the vertex in some sense. The distance generated by the straightest geodesic map, if it is successful, is the best approximation of the geodesic distance. Therefore, we compare the PDM with it in Fig. 5 to illustrate the similarity and difference between the geodesic distance and the Poisson distance. As is evident in the last column of Fig. 5, the straightest geodesic map may generate the wrong distance far from the right case for failing to cross a hole, while the PDM may lead to a better result. Although applications dependent on the geodesic distance strictly may not benefit from our PDM, we are sure that the PDM will find many applications where the magnitude of the distance is not vital, such as diffusing a value with the weights defined by the geodesic distance.

3.2.2 Boundary Poisson distance map

If C is the boundary vertices set, the solution of Eq. (1), with the cotangent weights as the edge weights, assigns each vertex a distance from the boundary of the mesh (BPDM), which can be seen as an approximation of the BGDM. It can be used to find the center vertex of a mesh patch automatically by choosing a global maximum point. If there are multiple global maximum points, any of them can be used. We illustrate two good properties of the BPDM in Fig. 6. First, the first and third columns of Fig. 6 show that both the BGDM and the BPDM consider the symmetry of the model. However, the center vertex determined by the BPDM shows the symmetry more than the BGDM. The model in the first column of

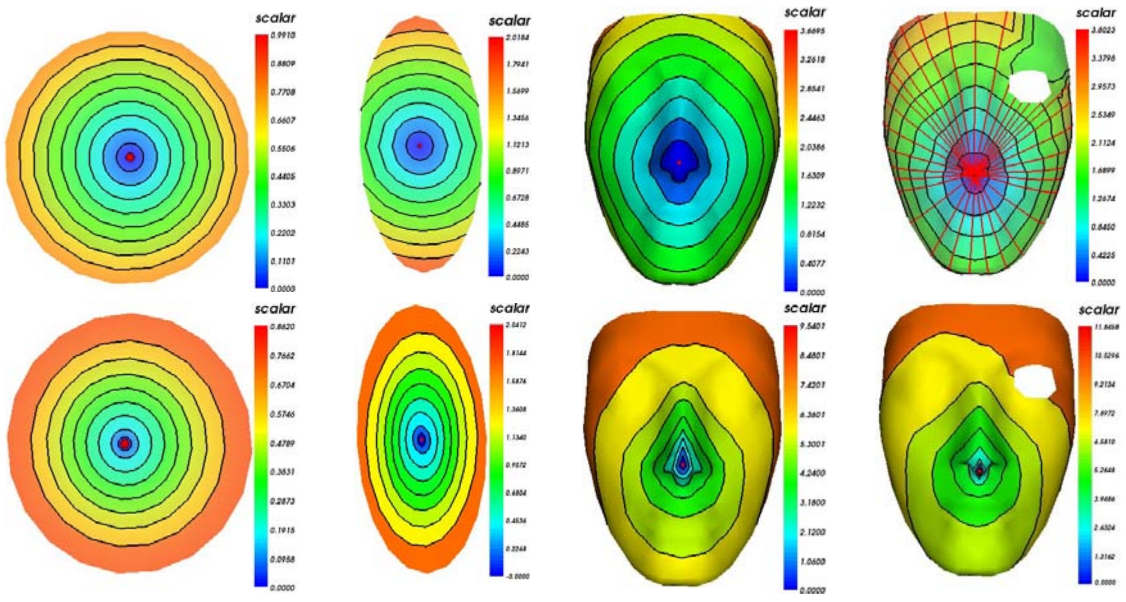
Fig. 5 Distance from a vertex
The top row is computed by the straightest geodesic distance map; the bottom row is computed by PDM

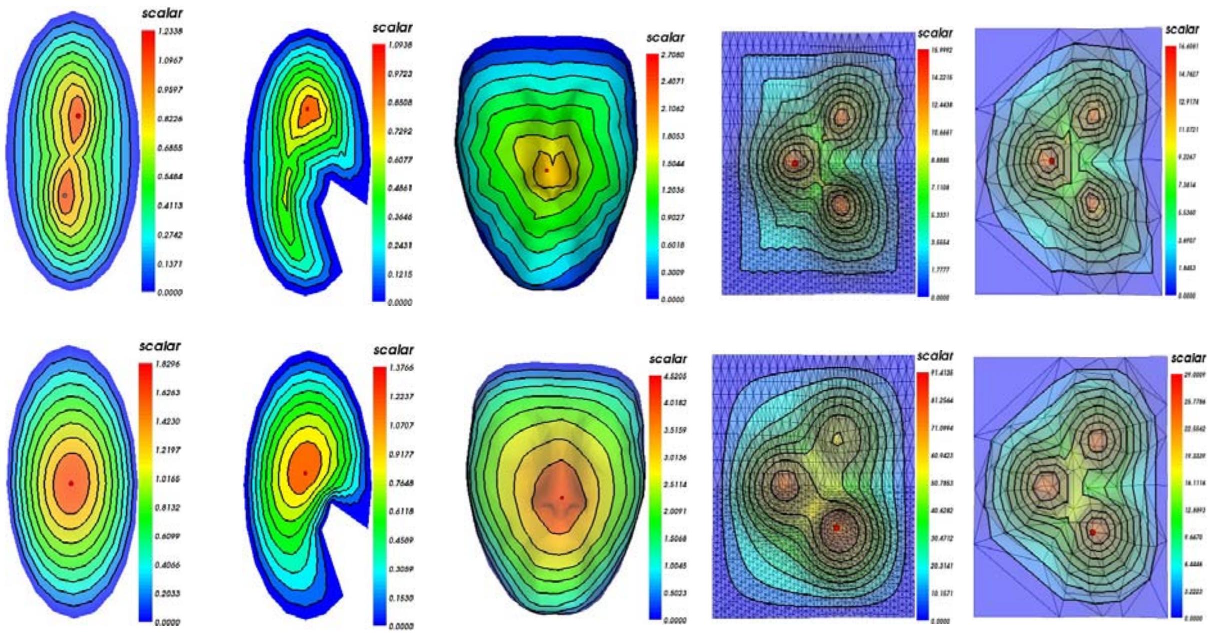
Fig. 6 Boundary distance map and the center vertex computed using the map
The top row is BGDM and its center vertex and the bottom row is BPDM and its center vertex

Fig. 6 is an ellipse scaled from a disk in one direction and the center vertex from the boundary should obviously be the polar vertex. The BPDM captures one global maximum point and it is the polar vertex, while the BGDM computes two global maximum points (for this simple and planar model, the closed vertex of the mean of the two vertices is the center vertex, but it may be not easy to locate a right closed vertex for complicate cases). In the third column, the center vertex determined by the BGDM also has some difference from the BPDM, which leads to higher distortion in the parameterization followed, as shown in the first column of Table 2 (see p.196). The second column shows results for a non-symmetric model.

Lastly, the last two columns show that both the BGDM and the BPDM are robust under different mesh resolutions and sampling rates.

3.2.3 Edge weights used in the Poisson distance map and boundary Poisson distance map

We have tested the mean value weights, cotangent weights, and spring weights for the PDM and the BPDM. The Poisson distance produced using the mean values weights or cotangent weights does not vary enough near the boundary to reflect the shape of the mesh patch, so we use the spring weights in the PDM. We use the cotangent weights in the BPDM as using the mean value weights fails to produce the same center vertex under different mesh resolutions, and using the spring weights is too sensitive to the boundary of the mesh patch which may produce a different center vertex even if the mesh's boundary is disturbed only a little.

4 Measured boundary parameterization

At a high level, measured boundary parameterization methods run in four successive steps as shown in Fig. 7. Sections 4.1–4.4 correspond to the steps in Fig. 7 and Section 4.5 discusses variation and validation of measured boundary methods.

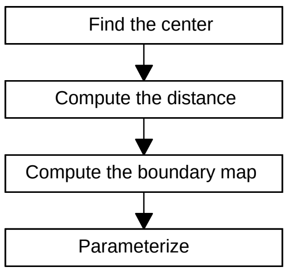

Fig. 7 Algorithm pipeline for the measured boundary paramererization

4.1 Find the center vertex

The first step is to choose a center vertex for a disk-like mesh, which has an effect upon the final parameterization distortion. An improper center vertex may lead to higher distortion. As far as we know, earlier measured boundary parameterization methods (Lee *et al.*, 2006; Lee and Lee, 2007; Shapira and Shamir, 2008) did not mention how it is computed. Both the BGDM and our newly defined BPDM can be used to find a good center vertex automatically. In our tests, the BPDM leads to less conformal distortion than the BGDM in most cases. We first compute the BPDM by setting all boundary vertices as constrained vertices and solving Eq. (1) using the cotangent weights. Then we choose a global maximum point of the map as the center vertex. If there are multiple global maximum points, any of them can be used.

4.2 Compute the 'distance' from the center vertex to all boundary vertices

After choosing the center vertex, the second step is to compute the 'distance' from it to all boundary vertices. Using the geodesic distance usually leads to less distortion, whilst the distance defined by the PDM leads to a more regular domain boundary. Though no previous work is concerned with the regularity of the domain boundary in a free boundary parameterization method, we think this property may play a role in some applications in the near future. The PDM is computed by setting the center vertex as a constrained vertex and solving Eq. (1) using the spring weights.

4.3 Compute the boundary map according to the 'distance' computed

In the third step, we build a polar map using the center vertex as the origin and map the boundary vertices to a unit circle first. Each central angle is determined by the ratio of angles formed by two successive boundary vertices and the center vertex to 2π. Then we change the radius according to the 'distance' assigned on each boundary vertex to obtain the boundary map. It is similar to the method in Lee and Lee (2007).

4.4 Parameterize the remaining vertices

Finishing the above three steps, we have parameterized the model's boundary into a polygon in a measured way. Now any fixed boundary parameterization method can be employed to parameterize the remaining parts. We use the discrete conformal parameterization here by solving Eq. (3) twice. Noting that the equations share the same left hand side with the equations in the first step, we can reuse the matrix factorization of the first step. Thus, the parameterization is computed simply by two back-substitutions, which can facilitate the whole process.

$$\begin{cases} \Delta f(v_i) = 0, & v_i \in V \setminus C, \\ f(v_i) = f_i, & \text{otherwise}, \end{cases} \tag{3}$$

where Δ is the Laplacian operator.

4.5 Variation and validation of measured boundary methods

The above four steps defined a framework of measured boundary parameterization. Composing different center vertex choosing strategies (BGDM or BPDM), and distance generated algorithms (the geodesic distance by Dijkstra's algorithm or by the fast marching algorithm or by the straightest geodesic map, or the PDM), and parameterization methods, we obtain different kinds of measured boundary parameterization methods. We test several kinds in Section 5.

None of the measured boundary methods guarantee the validation of the parameterization. If the parameterization is not a valid planar embedding, in spite of this seldom happening in our experiments, two post-processing methods can be performed, as introduced in Karni *et al*. (2005).

5 Results and discussion

We have shown the sequence of our approach and distortion compared with the DCP in Fig. 1. To present a full analysis of our approach, we have compared several kinds of measured boundary methods, ABF++ (Sheffer *et al*., 2005) and the DCP, in distortion measures, parameter domain boundaries, robustness, effects of choosing different center vertices, and computation time. Note that the results of ABF++ were generated by running Graphite 2.0-alpha developed by INRIA in France.

5.1 Distortion measures and parameter domain boundaries

We show a statistical comparison of the conformal distortion (Sheffer and de Sturler, 2000) and the L_2 distortion (Sander *et al*., 2001) of ABF++, the DCP and three kinds of measured boundary parameterizations in Table 1. Fig. 8 gives a more intuitive illustration of Table 1, in which the vertical axes are conformal and L_2 distortions, respectively (distortion generated by MBPSGP are not included since it failed in most of the tests). As we can see from the figure and the table, distortions of measured boundary methods, such as MBPGP, MBPPP and MBPSGP, were less than those of DCP, and higher than those of ABF++. Another observation is that the MBPGP generated both less conformal distortion and L_2 distortion than MBPPP for all models except the Mannequin model. The two observations are obvious since we know that, the freer the boundary is, the less the distortion generated by the parameterization is. Measured boundary methods generally produce a freer boundary than DCP and a less free boundary than ABF++, and MBPPP leads to a more regular boundary than MBPGP and ABF++, as shown in Fig. 9.

5.2 Robustness

Table 1 also shows that the MBPGP was the most robust of the three measured boundary methods, while the MBPSGP was the least robust. The MBPSGP often failed as it was so sensitive to the location of the center vertex that it cannot reach all the boundary vertices in most cases.

5.3 Effects of choosing different center vertices

The location of the center vertex has an effect on the subsequent parameterization distortion and it has

Table 1 Conformal and L_2 distortions of different methods

Method	Conformal distortion								L_2 distortion							
	Ellipse	Cutface	Ear	Fandisk	Mann.	Nef.	Stamp	Venus	Ellipse	Cutface	Ear	Fandisk	Mann.	Nef.	Stamp	Venus
ABF++	0	0.0100	0.0109	0.0030	0.0084	0.0063	0.0119	0.2252	1	1.1351	1.2521	1.0313	5.5042	1.0574	1.0499	1.0205
DCP	0.7108	0.7778	0.0835	2.5422	0.0300	0.0651	0.5519	0.1399	1.1076	1.6753	1.1536	3.0295	7.1079	1.0336	1.2495	1.0603
MBPSGP	0	–	–	–	–	0.0023	–	–	1	–	–	–	–	1.0208	–	–
MBPGP	0	0.0513	0.0247	1.1608	0.0218	0.0094	0.0527	0.0382	1	1.0618	1.1580	1.6234	7.1061	1.0208	1.0654	1.0335
MBPPP	0.2540	0.4493	0.0574	2.0626	0.0174	0.0375	0.2811	–	1.1656	1.3896	1.1705	3.0858	6.9891	1.0278	1.2027	–

Ellipse (Fig. 5); Cutface (Fig. 1); Ear (Fig. 2); Fandisk (Fig. 9); Mann.=Mannequin (Fig. 9); Nef.=Nefertiti (Fig. 5); Stamp (Fig. 4); Venus (Fig. 2). –: the respective methods failed to generate a valid parameterization. MBPSGP, MBPGP, and MBPPP represent the three methods, using the BPDM to locate the center vertex, the straightest GDM, GDM by fast marching, and the PDM to accomplish the second step, respectively

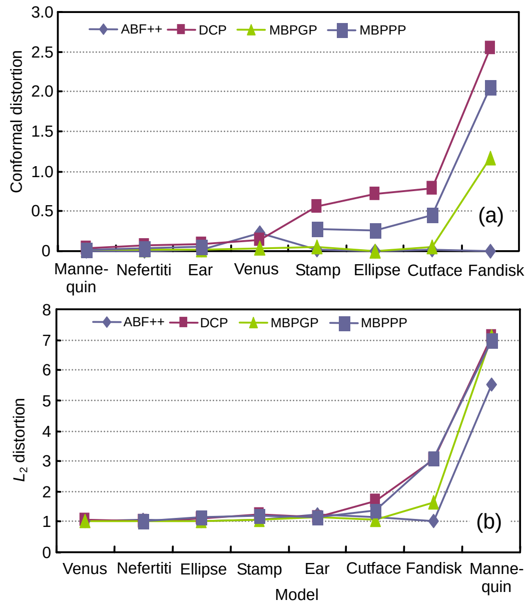

Fig. 8 (a) Conformal distortion and (b) L_2 distortion of different methods

been shown in Fig. 4. Table 2 gives a detailed comparison of determining the center vertex by BGDM and BPDM. We can see that finding the center vertex by BPDM led to less conformal distortion than by BGDM, both for using GDM by fast marching or PDM in the second step of a measured boundary method, in most cases. From the comparison, we deduce that BPDM is better than BGDM for measured boundary methods.

5.4 Computation time

We conclude with a runtime comparison. The MBPPP needed to solve four sparse linear systems, one for choosing a center vertex (step 1), one for computing a distance field (step 2), and the other two for parameterization (step 2). As the first and last two systems share the same Laplacian matrix, we can reuse the matrix factorization to facilitate our algorithm. Thus, the numerical procedure required only two matrix factorizations (steps 1 and 2) and four back-substitutions (steps 1, 2 and 4). As the Laplacian matrix is sparse, the factorization is very efficient. Similarly, the numerical procedure of the MBPGP was composed of one matrix factorization, three back-substitutions and one fast marching, as the only difference between them was that the MBPGP used the fast marching algorithm to accomplish the second step, instead of solving a linear system.

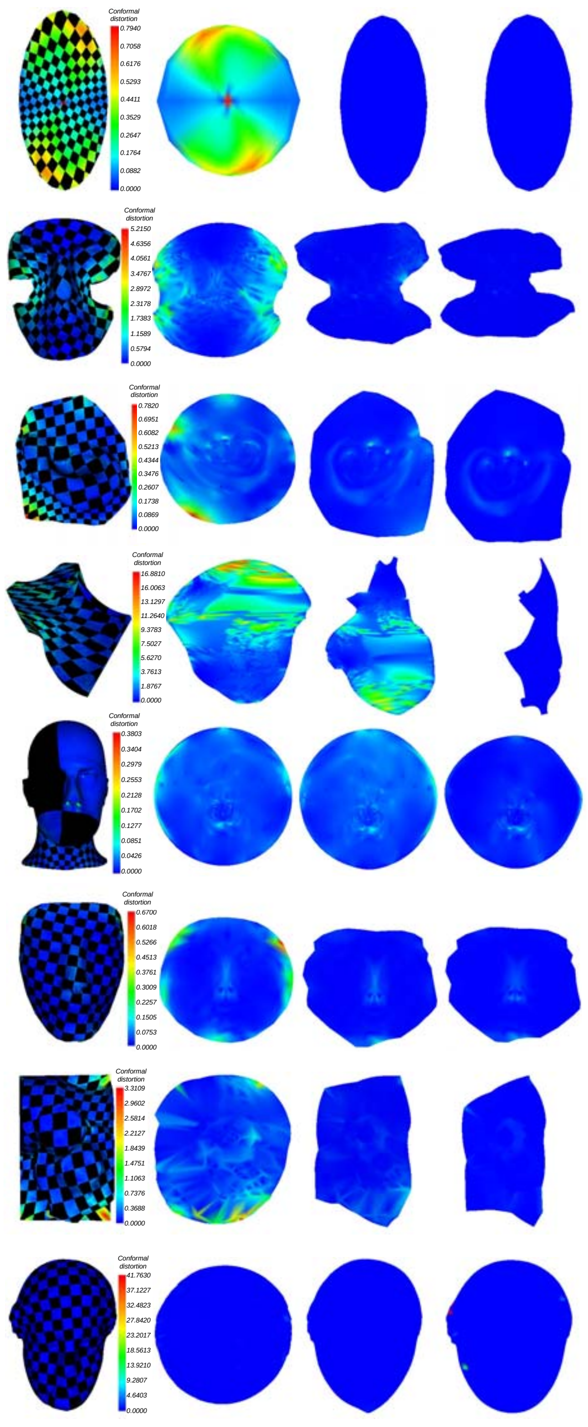

Fig. 9 Parameter domains of MBPPP, MBPGP, and ABF++

From top to bottom, the models used are Ellipse, Cutface, Ear, Fandisk, Mannequin, Nefertiti, Stamp, and Venus, respectively. The first column is models with texture mapping under MBPPP; the remaining columns, from left to right, are parameter domains of MBPPP, MBPGP, and ABF++, respectively. Conformal distortion is coded by color

Table 2 Conformal distortion obtained by choosing center vertex using different methods

Method	Conformal distortion							
	Nefertiti	Mannequin	Ear	Venus	Cutface	Stamp	Ellipse	Fandisk
BGDM+GDM	0.0104	0.0264	0.0262	0.0385	0.0422	0.0543	0.0350	1.0701
BPDM+GDM	0.0094	0.0218	0.0247	0.0382	0.0513	0.0527	0	1.1608
BGDM+PDM	0.0406	0.0238	0.0592	–	0.4644	0.3124	0.3608	2.0270
BPDM+PDM	0.0375	0.0174	0.0573	–	0.4493	0.2811	0.2540	2.0626

Table 3 lists some computation time of the ABF++, the DCP, the MBPGP, and the MBPPP. All time was measured on a 2.0 GHz Pentium notebook with 1 GB RAM. We used the supernodal sparse Cholesky factorization (CHOLMOD) (Chen *et al*., 2008) to factor matrices and solve the linear systems. For computing the geodesic distance, we employed the fast marching algorithm implemented by Gabriel Peyré. As is evident from Table 3, the MBPGP and the MBPPP were faster than ABF++. The fast marching algorithm was very efficient so that MBPPP was only a little faster than the MBPGP.

Table 3 Time comparison among ABF++, DCP, and our methods

Model	V	F	Time (s)			
			ABF++	DCP	MBPGP	MBPPP
Mannequin	6743	13424	4.172	0.249	0.499	0.498
Fandisk	6699	12946	1.406	0.188	0.391	0.376
Venus	1900	3526	0.390	0.030	0.092	0.060
Cutface	1203	2295	0.343	0.031	0.078	0.062
Ear	859	1676	0.203	0.015	0.046	0.030
Stamp	476	864	0.078	0.009	0.031	0.032

V: number of vertices; F: number of faces

6 Conclusion and future work

In this paper, we analyzed the effect of choosing a different center vertex for measured boundary methods. Based on this observation, we introduced two techniques, the BGDM and the BPDM, to determine the center vertex as necessary supplements to the existing measured boundary methods. Moreover, the center vertex by the BPDM respects the symmetry of a mesh path and leads to less distortion in our tests.

Using the above two techniques, we described a complete procedure for measured boundary methods. In addition, we presented a new measured boundary parameterization method by PDM, which leads to a more regular domain boundary with some distortion sacrificed. Our method is simple to implement because it relies only on solving three sparse linear systems. It is also efficient because of matrix factorization being reused. Hence, our method is much faster than the state-of-the-art ABF++.

All measured boundary methods, including our method, have a common drawback, that is the bijectivity of the parameterization, which cannot be guaranteed. Some post-processing stages are needed when the bijectivity is lost.

Some questions still remain unanswered. First and foremost, we would like to find a better boundary parameterization method according to conformal distortion and other distortions. Applying the PDM and the BPDM to other applications where the geodesic distance has played a central role, such as remeshing (Gabriel and Laurent, 2005), is also interesting work. Finally, we will try to apply hierarchical techniques to speed up our algorithm for large models.

Acknowledgements

We wish to express our thanks to the anonymous reviewers for giving advice, and to Rhaleb Zayer for providing the fan disk model with seam.

References

Aujay, G., Hétroy, F., Lazarus, F., Depraz, C., 2007. Harmonic Skeleton for Realistic Character Animation. Proc. ACM SIGGRAPH/Eurographics Symp. on Computer Animation, p.151-160.

Ben-Chen, M., Gotsman, C., Bunin, G., 2008. Conformal flattening by curvature prescription and metric scaling. *Comput. Graph. Forum*, **27**(2):449-458. [doi:10.1111/j.1467-8659.2008.01142.x]

Chen, Y., Davis, T.A., Hager, W.W., Rajamanickam, S., 2008. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Trans. Math. Softw.*, **35**(3):1-14. [doi:10.1145/1391989.1391995]

Desbrun, M., Meyer, M., Alliez, P., 2002. Intrinsic parameterizations of surface meshes. *Comput. Graph. Forum*, **21**(3):209-218. [doi:10.1111/1467-8659.00580]

Dong, S., Garland, M., 2007. Iterative Methods for Improving Mesh Parameterizations. IEEE Int. Conf. on Shape Modeling and Application, p.185-194. [doi:10.1109/SMI.2007.23]

Dong, S., Kircher, S., Garland, M., 2005. Harmonic functions for quadrilateral remeshing of arbitrary manifolds. *Comput. Aided Geom. Des.*, **22**(5):392-423. [doi:10.1016/j.cagd.2005.04.004]

Fattal, R., Lischinski, D., Werman, M., 2002. Gradient Domain High Dynamic Range Compression. Proc. 29th Annual Conf. on Computer Graphics and Interactive Techniques, p.249-256. [doi:10.1145/566654.566573]

Floater, M.S., 2003. Mean value coordinates. *Comput. Aided Geom. Des.*, **20**(1):19-27. [doi:10.1016/S0167-8396(03)00002-5]

Floater, M.S., Hormann, K., 2005. Surface Parameterization: A Tutorial and Survey. Advances in Multiresolution for Geometric Modelling. Springer Berlin Heidelberg, p.157-186. [doi:10.1007/3-540-26808-1_9]

Gabriel, P., Laurent, C., 2005. Geodesic Computations for Fast and Accurate Surface Remeshing and Parameterization. *In*: Brezis, H. (Ed.), Elliptic and Parabolic Problems. Springer-Verlag, p.157-171. [doi:10.1007/3-7643-7384-9_18]

Hormann, K., Greiner, G., 1999. MIPS: An Efficient Global Parameterization Method. Curve and Surface Design. Vanderbilt University Press, Saint-Malo, p.153-162.

Hormann, K., Lévy, B., Sheffer, A., 2007. Mesh Parameterization: Theory and Practice. ACM SIGGRAPH Course Notes, p.1-122.

Karni, Z., Gotsman, C., Gortler, S.J., 2005. Free-Boundary Linear Parameterization of 3D Meshes in the Presence of Constraints. Proc. Int. Conf. on Shape Modeling and Applications, p.268-277. [doi:10.1109/SMI.2005.22]

Kazhdan, M., Bolitho, M., Hoppe, H., 2006. Poisson Surface Reconstruction. Proc. 4th Eurographics Symp. on Geometry Processing, p.61-70.

Kharevych, L., Springborn, B., Schröder, P., 2006. Discrete conformal mappings via circle patterns. *ACM Trans. Graph.*, **25**(2):412-438. [doi:10.1145/1198555.1198665]

Kimmel, R., Sethian, J.A., 1998. Computing geodesic paths on manifolds. *PNAS*, **95**(15):8431-8435. [doi:10.1073/pnas.95.15.8431]

Lee, H., Tong, Y., Desbrun, M., 2005. Geodesics-based one-to-one parameterization of 3D triangle meshes. *IEEE Multim.*, **12**(1):27-33. [doi:10.1109/MMUL.2005.5]

Lee, S., Lee, H., 2007. Parameterization of 3D Surface Patches by Straightest Distances. Proc. 7th Int. Conf. on Computational Science: Part II, **4488**:73-80. [doi:10.1007/978-3-540-72586-2_10]

Lee, S., Han, J., Lee, H., 2006. Straightest paths on meshes by cutting planes. *LNCS*, **4077**:609-615. [doi:10.1007/11802914_47]

Lee, Y., Kim, H.S., Lee, S., 2002. Mesh parameterization with a virtual boundary. *Comput. Graph.*, **26**(5):677-686. [doi:10.1016/S0097-8493(02)00123-1]

Lévy, B., Petitjean, S., Ray, N., Maillot, J., 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, **21**(3):362-371. [doi:10.1145/ 566654.566590]

Liu, L., Zhang, L., Xu, Y., Gotsman, C., Gortler, S.J., 2008. A local/global approach to mesh parameterization. *Comput. Graph. Forum*, **27**(5):1495-1504. [doi:10.1111/j.1467-8659.2008.01290.x]

Mitchell, J.S.B., 2000. Geometric Shortest Paths and Network Optimization. Handbook of Computational Geometry. Elsevier Science, Amsterdam, the Netherlands, p.633-701.

Mullen, P., Tong, Y., Alliez, P., Desbrun, M., 2008. Spectral conformal parameterization. *Comput. Graph. Forum*, **27**(5):1487-1494. [doi:10.1111/j.1467-8659.2008.01289.x]

Nealen, A., Igarashi, T., Sorkine, O., Alexa, M., 2006. Laplacian Mesh Optimization. Proc. 4th Int. Conf. on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia, p.381-389. [doi:10.1145/1174429.1174494]

Pérez, P., Gangnet, M., Blake, A., 2003. Poisson image editing. *ACM Trans. Graph.*, **22**(3):313-318. [doi:10.1145/882262.882269]

Polthier, K., Schmies, M., 2006. Straightest Geodesics on Polyhedral Surfaces. ACM SIGGRAPH Course, p.30-38. [doi:10.1145/1185657.1185664]

Sander, P.V., Snyder, J., Gortler, S.J., Hoppe, H., 2001. Texture Mapping Progressive Meshes. Proc. 28th Annual Conf. on Computer Graphics and Interactive Techniques, p.409-416. [doi:10.1145/383259.383307]

Shapira, L., Shamir, A., 2008. Local Geodesic Parameterization: An Ant's Perspective. Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration, Mathematics and Visualization. Springer Berlin Heidelberg, p.127-137. [doi:10.1007/b106657_7]

Sheffer, A., de Sturler, E., 2000. Surface Parameterization for Meshing by Triangulation Flattening. Proc. 9th Int. Meshing Roundtable, p.161-172.

Sheffer, A., Hart, J.C., 2002. Seamster: Inconspicuous Low-Distortion Texture Seam Layout. Proc. Conf. on Visualization, p.291-298.

Sheffer, A., Lévy, B., Mogilnitsky, M., Bogomyakov, A., 2005. ABF++: fast and robust angle based flattening. *ACM Trans. Graph.*, **24**(2):311-330. [doi:10.1145/1061347.1061354]

Sheffer, A., Praun, E., Rose, K., 2006. Mesh parameterization methods and their applications. *Found. Trends Comput. Graph. Vis.*, **2**(2):105-171. [doi:10.1561/0600000011]

Sifri, O., Sheffer, A., Gotsman, C., 2003. Geodesic-Based Surface Remeshing. Proc. 12th Int. Meshing Roundtable, p.189-199.

Springborn, B., Schröder, P., Pinkall, U., 2008. Conformal equivalence of triangle meshes. *ACM Trans. Graph.*, **27**(3):1-11. [doi:10.1145/1360612.1360676]

Tierny, J., Vandeborre, J.P., Daoudi, M., 2006. 3D Mesh Skeleton Extraction Using Topological and Geometrical Analyses. Proc. 14th Pacific Conf. on Computer Graphics and Applications, p.85-94.

Wang, C.C.L., Wang, Y., Tang, K., Yuen, M.M.F., 2004. Reduce the stretch in surface flattening by finding cutting paths to the surface boundary. *Comput. Aided Des.*, **36**(8):665-677. [doi:10.1016/S0010-4485(03)00024-1]

Yang, Y.L., Kim, J.H., Luo, F., Hu, S.M., Gu, X.F., 2008. Optimal surface parameterization using inverse curvature map. *IEEE Trans. Vis. Comput. Graph.*, **14**(5): 1054-1066. [doi:10.1109/TVCG.2008.54]

Yu, Y.Z., Zhou, K., Xu, D., Shi, X.H., Bao, H.J., Guo, B.N., Shum, H.Y., 2004. Mesh editing with Poisson-based gradient field manipulation. *ACM Trans. Graph.*, **23**(3): 644-651. [doi:10.1145/1015706.1015774]

Zayer, R., Rössl, C., Seidel, H.P., 2005. Setting the Boundary Free: A Composite Approach to Surface Parameterization. Proc. 3rd Eurographics Symp. on Geometry Processing, p.91-100.

Zhou, K., Huang, J., Snyder, J., Liu, X.G., Bao, H.J., Guo, B.N., Shum, H.Y., 2005. Large mesh deformation using the volumetric graph Laplacian. *ACM Trans. Graph.*, **24**(3):496-503. [doi:10.1145/1073204.1073219]

Zhu, X.P., Hu, S.M., Ralph, M., 2003. Skeleton-based seam computation for triangulated surface parameterization. *LNCS*, **2768**:1-13. [doi:10.1007/b11966]

Appendix

Theorem A1 The linear Eq. (1) with one inner constrained vertex has one unique solution, for a mesh patch without holes, if all $\omega_{ij}>0$.

Proof We assume that v_l is the constrained vertex. That the system has a unique solution is equivalent to that the only solution of the equation

$$\begin{cases}\sum_{j\in N_i}\omega_{ij}(f(v_j)-f(v_i))=0 \quad \forall i\neq l,\\ f(v_l)=0_i,\end{cases} \tag{A1}$$

is $f(v_i)=0$, $i=1, 2, ..., n$.

Let $f_{\max}$ be the maximum of the $f(v_i)=0$, $i=1, 2, ..., n$ and suppose that $f(v_m)=f_{\max}$. Consider any neighbor vertex v_j of the vertex v_m. Since

$$f(v_m)=\sum_{j\in N_m}\omega_{mj}f(v_j), \tag{A2}$$

and because $\forall\omega_{ij}>0$, the only way that Eq. (A2) can be satisfied is $f(v_j)=f_{\max}$; thus, every neighbor vertice of v_k must satisfy $f(v_j)=f_{\max}$, $j\in N_m$. Eventually, since the mesh patch is connected and without holes, a boundary vertex v_b must be reached with the result $f(v_b)=f_{\max}$. This implies that $f_{\max}=0$ and therefore $f=0$.