Surface Parameterization via Aligning Optimal Local Flattening

Zhonggui Chen Ligang Liu[*] Zhengyue Zhang Guojin Wang

Department of Mathematics
Zhejiang University, China

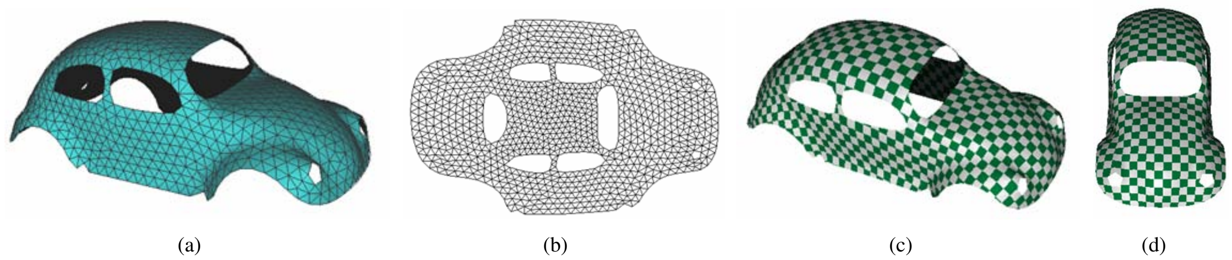


Figure 1: Parameterizing the beetle model with several holes using our approach. (a) The beetle model; (b) the parametrization result; (c),(d) different views of texture mapping results.

Abstract

This paper presents a novel parameterization method for a non-closed triangular mesh. For every flattened 1-ring neighbors, we choose a local coordinate frame, and the local geometry structure is represented as local parametric coordinates. Then the global optimal parametric coordinates are attained by aligning all the local parametric planes while preserving the local structure as much as possible. The boundary conditions are not necessary in our method, thus no high distortion appears around the boundary, and distortion is uniformly distributed over parametric domain. In addition, our method can operate directly on mesh surface which has holes without any preprocessing of surface partition. Furthermore, linear constraints are allowed in the parameterization in a least squares sense.

CR Categories: J.6 [Computer-Aided Engineering]: Computer-aided design—Surface parameterization;

Keywords: Surface parameterization, local geometry structure, alignment, texture mapping

1 Introduction

Surface parameterization can be viewed as a one-to-one mapping from a mesh surface onto a suitable domain. Generally, a complex mesh is decomposed into a set of simple patches that are topologically equivalent to disks. When unfolding a patch onto a plane, the reduction of angle and area distortion are the two principal aims. The existing non-linear methods addressing this issue usually lead to intricate and computationally expensive numerical schemes. On the other hand, most of the linear methods require to fix the boundary of the patch, which typically generate significantly more distortion than free-boundary techniques. In particular, if a non-closed patch has more than one boundary (i.e., having holes inside it), a general approach is to partition the patch into a set of simple subpatches. As we have known, partition often produces high distortion and discontinuity along cutting boundaries.

In this paper, we present techniques for surface parameterization by solving eigen system of a matrix or solving linear systems which can deal with meshes with multiple boundaries and does not require fixed boundaries. This is a new way of creating non-convex parameterization patches that may even have holes.

We are much inspired by the recent work of local tangent space alignment (LTSA) algorithm for manifold learning and nonlinear dimensionality reduction[Zhang and Zha 2005]. In the convex combination method, an interior vertex is represented as a convex combination of its 1-ring neighbors [Floater 1997], while we represent the local geometry structure of the vertex and its 1-ring neighbors as local parametric coordinates. In our approach, the local patch is optimally flattened into a small planar patches, keeping the local geometry structure well. Different local patches might have overlapped region with each other.

Then the global optimal parametric coordinates are attained by aligning all the local parametric patches [Zhang and Zha 2005]. Since most of the vertices correspond to more than one local patch, a direct placement of all patches could not in general be conforming with all the individual local patches. All the local patches are aligned in the plane by different local affine transformations to obtain a global coordinate system. Thus, the global alignment is performed by minimizing all the transformations of the local patches, which preserves the local geometry structure as much as possible. This is attained by solving the eigenvalues and eigenvectors of a matrix.

As the alignment optimization is performed over all vertices, it is not necessary to set boundary conditions. Thus no high distortion appears around the boundary and distortion is uniformly distributed over parametric domain. In addition, our method can operate directly on mesh surface which has holes without any preprocessing of surface partition, while partition-based methods often produce high distortion and discontinuity around the partition boundaries. Linear constraints on the vertices on the parametric domain are easily be integrated into our parameterization approach, which is obtained by solving a sparse linear system.

[*] Correspondence: ligangliu@zju.edu.cn

Our method preserves both the local and the global structure of the mesh surface, and it is very suitable for computer graphics applications which require parameterization with low geometric distortion, such as texture mapping.

2 Related Work

There has been a large amount of work on surface parameterizations in the literature. We refer the reader to [Floater and Hormann 2005] for a more general survey. Below we briefly review the major techniques.

Fixed boundary approaches treat the surface flattening problem as minimizing a functional that measures the distortion of the triangles, which typically generate significantly more distortion than free-boundary techniques. Generally, the boundary vertices are fixed on a convex polygon. Then the parameterization for the interior vertices is formulized as a large linear system or a nonlinear optimization problem[Floater 1997; Floater 2003].

Non-linear approaches do not require to fix the boundaries[Sheffer and de Sturler 2001; Lévy et al. 2002; Sheffer et al. 2005]. Zigelman et al. use a multi-dimensional scaling (MDS) method that optimally preserves the geodesic distances between mesh vertices[Zigelman et al. 2002]. In a recent work, Zayer et al. present a boundary-free parameterization method consisting of several simple steps, each solving a linear problem. The overall process is controlled using suitable guidance tensor fields reflecting the intrinsic surface geometry [Zayer et al. 2005]. The non-linear approach in [Zhang et al. 2005] uses Green-Lagrange tensor to measure and to guide the optimization process by adding scaffold triangles. Other approaches use the global conformal structure or discrete one-forms to parameterize surface [Gu and Yau 2003; Gortler et al. 2006; Kharevych et al. 2006; Ray et al. 2006].

A limitation of planar parameterization techniques is that it generally requires that an entire surface be cut into one or more disk-like charts, where each chart is parameterized independently[Gu et al. 2002; Sander et al. 2003]. A closed genus-zero surface can be parameterized into the unit sphere without any cuts [Praun and Hoppe 2003; Gotsman et al. 2003]. Other approaches parameterize the mesh onto a simplicial domain which could be a simplified based mesh [Lee et al. 1998] or an abstract simplicial complex [Praun et al. 2001; Kraevoy and Sheffer 2004].

Surface parameterization is also related to the problems of dimensionality reduction and manifold learning which aim to compress the data in size and to discover compact representations of its variability. Classical techniques include principal components analysis [Jolliffe 1989] or multidimensional scaling (MDS)[Cox and Cox 1994]. More generally there is a wider class of techniques. Local approaches (LLE [Roweis and Saul 2000], Laplacian Eigenmaps [Belkin and Niyogi 2002]) attempt to preserve the local geometry of the data by mapping nearby points on the manifold to nearby points in the low-dimensional representation. Global approaches (Isomap [Tenenbaum et al. 2000]) attempt to preserve geometry at all scales. Recently, the local tangent space alignment (LTSA) algorithm [Zhang and Zha 2005] uses the tangent space in the neighborhood of a data point to represent the local geometry and then align those tangent spaces to construct the global coordinate system.

3 Global Alignment of Optimal Local Flattening

We begin with a brief overview of the proposed framework and then elaborate on its different components.

3.1 Overview

A triangular surface mesh $\mathcal{M}$ is represented as the pair $(\mathcal{T}, X)$, where $\mathcal{T}$ is a simplicial complex representing the connectivity of vertices, edges and faces, and $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, where $\mathbf{x}_i$ refer to the geometric position of the vertices in $\mathbb{R}^3$. The triangular surface mesh $\mathcal{M}$ we deal with in this paper is non-closed, but possibly with holes.

Let $\mathbf{x}_{i_1}, ..., \mathbf{x}_{i_k}$ be the 1-ring neighbors of $\mathbf{x}_i$, and define $X_i = \{\mathbf{x}_i, \mathbf{x}_{i_1}, ..., \mathbf{x}_{i_k}\}$. Then our basic approach consists of the following two steps:

- **Optimal local flattening:** The local patch consisting of a vertex as well as its 1-ring neighbors is optimally flattened into small planar patches. All the local patches may overlap with each other.
- **Global Alignment:** All the local patches are aligned in the plane by different local affine transformations to obtain a global coordinate system. The local geometric structures of the patches are preserved as much as possible during the alignment.

3.2 Optimal local flattening

There are different ways of mapping X_i into the plane, i.e., mapping $\mathbf{x}_i$ into $\mathbf{q}_i$ and $\mathbf{x}_{i_1}, ..., \mathbf{x}_{i_k}$ into suitable $\mathbf{q}_{i_1}, ..., \mathbf{q}_{i_k}$ respectively.

We adopt a local geodesic polar map first introduced by Welch and Witkin [Welch and Witkin 1994], which has been known in differential geometry which preserves arc length in each radial direction. The vertex $\mathbf{x}_i$ and its neighbors are flattened in the plane such that the edge length are exactly preserved, and the angles between two consecutive edges are preserved up to a common factor. This can easily be done incrementally, by selecting a first edge and constructing the flattened 1-ring by pivoting around the middle vertex while ensuring the desired properties. That is,

$$\| \mathbf{q}_{i_j} - \mathbf{q}_i \| = \| \mathbf{x}_{i_j} - \mathbf{x}_i \|,$$

$$\angle(\mathbf{q}_{i_j}, \mathbf{q}_i, \mathbf{q}_{i_{j+1}}) = 2\pi\angle(\mathbf{x}_{i_j}, \mathbf{x}_i, \mathbf{x}_{i_{j+1}}) / \sum_{j=1}^{k} \angle(\mathbf{x}_{i_j}, \mathbf{x}_i, \mathbf{x}_{i_{j+1}}),$$

$$j = 1, ..., k,$$

where $\mathbf{x}_{i_{k+1}} = \mathbf{x}_{i_1}, \mathbf{q}_{i_{k+1}} = \mathbf{q}_{i_1}$ and $\angle(\mathbf{a}, \mathbf{b}, \mathbf{c})$ denotes the angle between the vectors $\mathbf{a} - \mathbf{b}$ and $\mathbf{c} - \mathbf{b}$, see Fig. 2(a),(b).

For boundary point $\mathbf{x}_i$ and its 1-ring neighbors, we compute $\mathbf{q}_{i_1}, ..., \mathbf{q}_{i_k}$ as

$$\| \mathbf{q}_{i_j} - \mathbf{q}_i \| = \| \mathbf{x}_{i_j} - \mathbf{x}_i \|,$$

$$\angle(\mathbf{q}_{i_j}, \mathbf{q}_i, \mathbf{q}_{i_{j+1}}) = \angle(\mathbf{x}_{i_j}, \mathbf{x}_i, \mathbf{x}_{i_{j+1}}), \;\; j = 1, ..., k.$$

It can be seen that $\mathbf{q}_i, \mathbf{q}_{i_1}, ..., \mathbf{q}_{i_k}$ are unique up to translations and rotations in $\mathbb{R}^2$ [Floater 1997]. We set $\mathbf{q}_i = 0$ and $\mathbf{q}_{i_1} = (\| \mathbf{x}_{i_1} - \mathbf{x}_i \|, 0)$ and then compute $\mathbf{q}_{i_2}, ..., \mathbf{q}_{i_k}$ in sequence.

This local flattening may be regarded as an optimal mapping because it is approximately conformal as it preserves the angle ratios, and some local lengths as well.

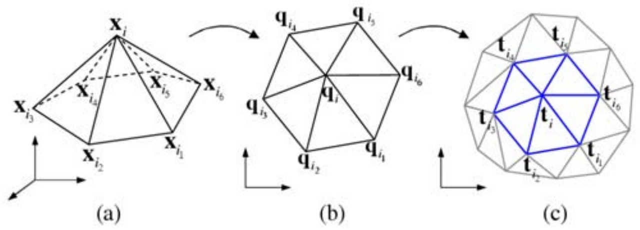


Figure 2: Local and global parametric coordinates of a vertex and its 1-ring neighbors: (a) vertex and its 1-ring neighbors; (b) optimal local flattening; (c) global parametric coordinates.

3.3 Global alignment

Now for each vertex $\mathbf{x}_i$ and its 1-ring neighbors $\mathbf{x}_{i_1}, ..., \mathbf{x}_{i_k}$, we have a flattened patch. The local geometry structures of all patches are represented as local parametric coordinates $\{\mathbf{q}_i, \mathbf{q}_{i_1}, ..., \mathbf{q}_{i_k}\}$.

We now consider to construct the global parametric coordinates $\mathbf{t_i} \in \mathbb{R}^2$, for all the vertices $\mathbf{x}_i, i = 1, ..., N$, see Fig. 2(c). We want the global coordinates to preserve local geometry structures determined by the local coordinates $\mathbf{q}_{i_j}$. The basic idea is inspired from the work of [Zhang and Zha 2005]. We briefly introduce the global alignment approach in this section.

Each of the local patches $\mathbf{q}_i, \mathbf{q}_{i_1}, ..., \mathbf{q}_{i_k}$ corresponds to a patch $\mathbf{t}_i, \mathbf{t}_{i_1}, ..., \mathbf{t}_{i_k}$ in the global parametrization, see Fig. 2(c). Since most of the vertices correspond to more than one local patch, a direct placement of all patches could not in general be conforming with all the individual local patches.

For each pair of local patches $\{\mathbf{q}_i, \mathbf{q}_{i_1}, ..., \mathbf{q}_{i_k}\}$ and $\{\mathbf{t}_i, \mathbf{t}_{i_1}, ..., \mathbf{t}_{i_k}\}$, we compute an affine transformation $U_i \in \mathbb{R}^{2\times 2}$ (without translation term) between them which minimizes the following errors:

$$\sum_{j=0}^{k} \|\epsilon_{i_j}\|^2, \;\; i = 1, 2, ..., N,$$

where

$$\epsilon_{i_j} = \mathbf{t}_{i_j} - (\mathbf{t}_i + U_i \mathbf{q}_{i_j}), j = 0, 1, ..., k, i = 1, ..., N, \quad (1)$$

is the local reconstruction error and $i_0 = i$. Let

$$T_i = [\mathbf{t}_i, \mathbf{t}_{i_1}, ..., \mathbf{t}_{i_k}], Q_i = [\mathbf{q}_i, \mathbf{q}_{i_1}, ..., \mathbf{q}_{i_k}].$$

Denoting $E_i = [\epsilon_i, \epsilon_{i_1}, ..., \epsilon_{i_k}]$, we can rewrite Eq. 1 in matrix form as

$$E_i = T_i(I - \mathbf{e}\mathbf{1}^T) - U_i Q_i, i = 1, ..., N. \quad (2)$$

Then the optimal affine transformation matrix U_i is obtained in matrix form by

$$U_i = T_i(I - \mathbf{e}\mathbf{1}^T)P_i, \quad (3)$$

where I is the identity matrix, $\mathbf{e} = [1, 0, ..., 0]^T$ is a vector containing 1 in first position and 0 elsewhere, $\mathbf{1}$ is a $k+1$-dimensional column vector of all ones, and $P_i = Q_i^T(Q_i Q_i^T)^{-1}$.

We substitute the form of U_i in Eq. 3 into Eq. 2 and rewrite E_i in terms of the matrix T_i as

$$E_i = T_i(I - \mathbf{e}\mathbf{1}^T)(I - P_i Q_i), i = 1, ..., N.$$

To preserve as much of the local geometry in the global parametric coordinates, we minimize the overall reconstruction errors, i.e.,

$$\sum_{i=1}^{N} \omega_i \|E_i\|_F^2 = \sum_{i=1}^{N} \omega_i \|T_i(I - \mathbf{e}\mathbf{1}^T)(I - P_i Q_i)\|_F^2,$$

where the weights ω_i are areas of local flattened patches and $\|.\|_F$ stands for the Frobenius norm of a matrix.

Let $T = [\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_N]$, $W_i = (I - \mathbf{e}\mathbf{1}^T)(I - P_i Q_i)$, and S_i be the $0-1$ selection matrix such that $TS_i = T_i$. We then need to find the T to minimize the overall reconstruction error

$$\sum_{i=1}^{N} \omega_i \|E_i\|_F^2 = \|TSW\|_F^2, \quad (4)$$

where $S = [S_1, ..., S_N]$ and $W = diag(\sqrt{\omega_1}W_1, ..., \sqrt{\omega_N}W_N)$. The minimization of Eq. 4 is performed subject to some constraints that make the problem well posed. Otherwise, one can just choose T to be zero. We rewrite T in the from, $T = [\mathbf{u}, \mathbf{v}]^T$, where $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{N\times 1}$ are N-dimensional column vectors. To avoid T degenerating into zero, we will impose the constraints

$$\|\mathbf{u}\|^2 = 1, \|\mathbf{v}\|^2 = 1. \quad (5)$$

Note that we can translate the output T by a constant displacement without affecting the minimization objective function. We remove the translational degree of freedom by requiring the outputs to be centered on the origin:

$$\sum_{i=1}^{N} \mathbf{t}_i = 0. \quad (6)$$

If $\mathbf{u}$ and $\mathbf{v}$ are linearly dependent, all the output points will lie on one line. Based on this insight, we require that vectors $\mathbf{u}$ and $\mathbf{v}$ are orthogonal:

$$\mathbf{u}^T \cdot \mathbf{v} = 0. \quad (7)$$

This constraint makes the surface be flattened as much as possible.

Under the constraints Eq. 5- 7, the optimal global parametric coordinates T is uniquely determined up to a rotation. The optimization of Eq. 4 can be performed by introducing Lagrange multiplier to enforce the constraints in Eq. 5- 7.

Using some matrix computing techniques[Horn and Johnson 1990; Zhang and Zha 2005], the optimal global coordinates T is given by the two eigenvectors of the matrix B corresponding to the 2nd and 3rd smallest eigenvalues, where the matrix B is as follows

$$B = SWW^T S^T. \quad (8)$$

Note that the smallest eigenvalue of matrix B is zero and the corresponding eigenvector is the vector $\mathbf{1}$ of all ones. Discarding this eigenvector enforces the constraint in Eq. 6 that the parametrization region is centered at origin.

As we have seen from the above derivation, all the local flattened coordinates are optimally aligned in the plane by different local affine transformation to obtain a global coordinate system. And an elegant computation approach in matrix form is derived.

4 Texture Mapping with Feature Constraints

In texture mapping an image is wrapped on a mesh surface in order to give it a realistic look, which benefits from the planar parameterization. To achieve better results, it is often necessary to force a

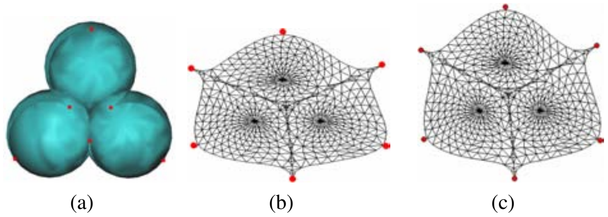


Figure 3: User interactive parameterization in realtime: (a) the original mesh; (b),(c) parameterization results with different point constraints.

correspondence between some of the details of the texture and the features of the model. For example, in a 3D face mesh, it is important to register the eyes of the mesh and the eyes in the image, and the same for the mouth, nose and any other prominent features, see Fig. 6.

The constrained texture mapping problem can be thought as parameterizing the 3D mesh while constraining some of the vertices to lie in a specified feature position in the texture coordinate plane. That is, some position constraints should be added into the minimization problem in Eq. 4.

The user can constrain some points to given parameter values as

$$\mathbf{t}_{i_k} = \mathbf{c}_{i_k}, k = 1, ..., s. \tag{9}$$

We add the additional constraints Eq. 9 as soft constraints in the minimization of Eq. 4, which results in the following linear system

$$TA = T[SW, L] = [0, C], \tag{10}$$

where L is $N \times s$ matrix in which each column contains only one non-zero element used to constrain the position of the points with the element:

$$l_{jk} = \begin{cases} \mu, & j = i_k; \\ 0, & \text{else}; \end{cases} \quad 1 \leq k \leq s,\ 1 \leq j \leq N;$$

where μ is the weight of the point constraints, $A = [SW, L]$, and $C = \mu[c_{i_1}, ..., c_{i_s}]$ is a $2 \times s$ matrix.

The least squares solution of the linear system can be found by solving the normal equation of Eq. 10, which is equivalent to minimizing the following error function:

$$\|TSW\|_F^2 + \mu^2 \sum_{k=1}^{s} \|\mathbf{c}_{i_k} - \mathbf{t}_{i_k}\|^2.$$

The positions of the points can be found by solving the sparse linear system in Eq. 10 in a least squares sense as:

$$T = [0, C]A^T(AA^T)^{-1}.$$

Thanks to the linear nature of solutions of the linear system, we can allow the user to interactively modify the positions of constrained points while updating the parameterization in realtime. First we compute the least squares solutions of the following linear equations:

$$\mathbf{g}_k^T A = [0, \mu \mathbf{e}_k^T], k = 1, ..., s,$$

where $\mathbf{e}_k$ is an $s \times 1$ column vector containing 1 in k-th position and 0 elsewhere, and $\mathbf{g}_k$ is the unknown $N \times 1$ column vector. By solving these linear systems, we obtain a set of basic solutions $\mathbf{g}_k (k = 1, ..., s)$ of Eq. 10. Thus the solution of Eq. 10 can be explicitly represented as

$$T = \sum_{k=1}^{s} \mathbf{c}_{i_k} \mathbf{g}_k^T.$$

Therefore, the parameterization for a given set of constrained points can easily be reconstructed in realtime. And the user can interactively adjust the positions of constrained points to get a satisfying parameterization result, see Fig. 3.

5 Experimental Results

We have applied our parameterization approach to a variety of non-closed 3D meshes.

Fig. 4(a) shows an irregularly sampled face model. Fig. 4(b) shows the parameterization result using the proposed approach. Note that the symmetry of the face features is preserved despite the drastic change in sampling rate. Fig. 4(c) shows the model with a checkerboard texture map in order to visualize the quality of the parameterization. We can see that the checkerboard pattern is not visibly distorted by the mapping.

We compare our method with the method proposed in [Zigelman et al. 2002] which finds an embedding of an open mesh in the plane by a multi-dimensional scaling (MDS) method that optimally preserves the geodesic distances between mesh vertices. This method also does not require predefined planar boundary. However, the method does not work well for meshes with holes, see Fig. 5. Any hole will likely be a circle in the parameterization domain in the method as it preserves the geodesic distance between vertices but introduces high distortions around interior boundaries, as shown in Fig. 5(b). Moreover, this method is computationally expensive, since it requires computing the geodesic distance between every two vertices on the surface. The mesh model with 803 vertices and 1524 triangles shown in Fig. 5(a) is flattened in 0.457 and 91 seconds by our method and method of [Zigelman et al. 2002] respectively. We obtain satisfying parameterization results of meshes with holes using our method, see Fig. 5(c) and Fig. 1.

Fig. 6 shows the result of texture mapping by adding feature point constraints in the planar parameterization. The prominent features on the 3D face model shown in Fig. 6(a), including eyes, nose, and mouth, are specified to be mapped to the corresponding points in the image show in Fig. 6(b). We obtain a proper texture mapping without any pre-operations on the texture image, such as warping and deformations, see Fig. 6(c) and (d).

As described in Section 4, the user can interactively adjust the positions of constrained points to get a satisfying parameterization result in realtime. Fig. 3(b) and Fig. 3(c) show two parametrization

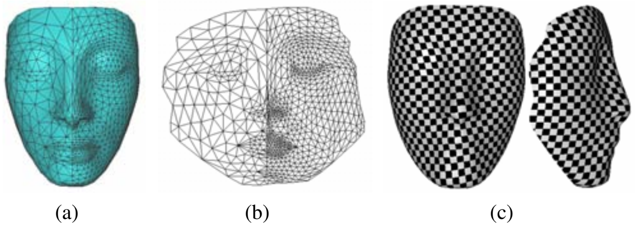


Figure 4: Example of parameterizing irregularly sampled model: (a) original face model; (b) parameterization result; (c) different views of the model with checkerboard texture mapping.

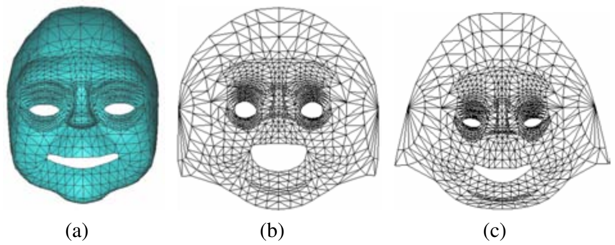

(a) (b) (c)

Figure 5: Comparison with method of Zigelman et al.'s method: (a) original face model with holes; (b) parameterization result of Zigelman et al.'s method; (c) parameterization result of our method.

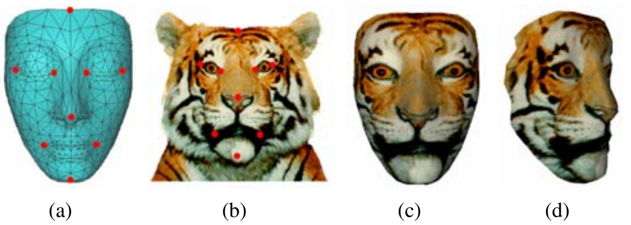

(a) (b) (c) (d)

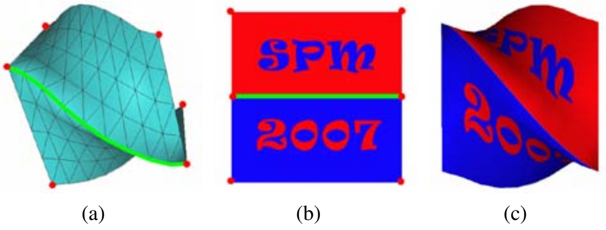
Figure 6: Constrained texture mapping: (a) The original face model with constrained vertices marked as red dots; (b) texture image with positions corresponding to constrained vertices ; (c), (d) textured model with different views.

results of the mesh shown in Fig. 3(a) under two different point constraints specified by the users.

It is worthwhile to point out that we can also add other linear constraints such as line constraints into our method. Fig. 7(a) shows a twisted surface with a sharp edge in green color. The vertices on the sharp edge are constrained to be mapped onto the points located at one line on texture domain shown in green color in Fig. 7(b). Hence the boundary in texture image coincides with the sharp edge on the twisted surface after texture mapping (see Fig. 7(c)). Furthermore, users can introduce scores of constraints while maintaining a low-distortion parameterization.

5.1 Implementations

The most time-consuming parts of our algorithm are computing the eigenvectors of the matrix B in Eq. 8 and solving the linear system Eq. 10. Note that both the matrix B and AA^T are sparse and symmetric. We use the ARPACK sparse eigensystem solver (http://www.caam.rice.edu/software/ARPACK) to compute the eigenvectors of B. It implements an iterative Arnoldi method that allows us to efficiently compute only the bottom 2 eigenvectors of B. And we use the direct solver TAUCS [Toledo 2003] in our implementation to solve the sparse linear system.



(a) (b) (c)

Figure 7: Texture mapping with line constraints: (a) a twisted surface with a sharp edge; (b) a texture image; (c) the texture mapping result.

A summary of the results of our method is shown in Table 1. All the examples presented in this paper were made on a 2.8GHz Pentium IV computer with 1.0GB memory. For comparison, we use the geometric stretch metric L^2 and L^∞ defined in [Sander et al. 2001] as the measures. The L^2 norm measures the mean stretch over all directions, while the worst-case norm L^∞ measures the greatest stretch. Our method turns out to solve a linear system of equations if any user constraints are given. All running times and stretch measures of our method with user constraints are measured before user adjusting the positions of constrained points. The difference between the running times of both of our methods is very small, thanks to using the efficient sparse eigensystem solver in our implementation. We can see that parameterization results generated by our method without user constraints always have lower distortions.

5.2 Discussion

The optimization of Eq. 4 can be performed subject to either nonlinear constraints 5- 7 or linear constraints 9. The user can constrain serval points to given positions to get a least squares solution, other than to optimize the Eq. 4 with the nonlinear constraints $5-7$ by computing the eigenvectors of matrix. However, sometimes it is hard to specify points to appropriate positions, and we find that the solutions with nonlinear constraints always have lower distortions (see Table 1).

In this paper, we adopt a simple way which only considers flattening each 1-ring neighborhood to plane. It is proved to be sufficient in all our experiments. However, any other low-distortion local flattening methods (with larger region of neighborhood) can be integrated into our framework.

6 Conclusion

We proposed a novel approach for parameterizing a non-closed triangular mesh that might have interior holes. Our approach is inspired by the recent work of LTSA for manifold learning. First each patch consisting each vertex as well as its 1-ring neighbors is optimally flattened in a local coordinate frame. Then all the local flattening patches are aligned by different local affine transformations to obtain a global coordinate system. The boundary conditions are not necessary in our method, thus no high distortion appears around the boundary, and distortion is uniformly distributed over parametric domain. The user can also specify linear constraints and the parameterization is obtained in a least squares sense. Our approach can preserve both the local and global structure of the mesh surfaces, thus it is very suitable for many computer graphics applications including texture mapping.

The major drawback in our current implementation is that the proposed approach may contain local overlapping triangles in the planar embedding. It is much worthwhile to combine some iterative method such as [Sander et al. 2001] with our approach to produce unfolding mapping.

References

BELKIN, M., AND NIYOGI, P. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds. MIT Press, 157–186.

Model	**Size**		**Without user constraints**			**With user constraints**		
	Vertex	Triangle	Time (s)	L^2	L^∞	Time (s)	L^2	L^∞
Beetle (Fig. 1)	984	1759	0.6310	1.0224	1.3560	0.4148	1.0279	2.4263
Balls (Fig. 3)	547	1032	0.2636	1.0540	3.6926	0.1875	1.1175	3.9296
Face (Fig. 4)	729	1404	0.4031	1.0123	1.5016	0.3021	1.0403	1.9915
Face (Fig. 5)	803	1524	0.4565	1.0542	2.5641	0.3123	1.1486	2.6947
Twisted Surface (Fig. 7)	90	140	0.0282	1.1438	1.5213	0.0116	1.1752	1.6251

Table 1: A summary of the running time results of our method with and without user constraints.

Cox, T., and Cox, M. 1994. *Multidimensional Scaling*. Chapman & Hall, London.

Floater, M. S., and Hormann, K. 2005. Surface parameterization: a tutorial and survey. In *Advances in Multiresolution analysis of Geometric Modelling*, N. A. Dodgson, M. S. Floater, and M. A. Sabin, Eds. Springer Verlag, 157–186.

Floater, M. S. 1997. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design 14*, 3, 231–250.

Floater, M. S. 2003. Mean value coordinates. *Computer Aided Geometric Design 20*, 1, 19–27.

Gortler, S., Gotsman, C., and Thurston, D. 2006. Discrete one-forms on meshes and applications to 3d mesh parameterization. *Computer Aided Geometric Design 33*, 2, 83–112.

Gotsman, C., Gu, X., and Sheffer, A. 2003. Fundamentals of spherical parameterization for 3D meshes. In *Proceedings of SIGGRAPH*, 358–364.

Gu, X., and Yau, S. 2003. Global conformal surface parameterization. In *Proceedings of ACM Symposium on Geometry Processing*.

Gu, X., Gortler, S. J., and Hoppe, H. 2002. Geometry images. In *Proceedings of SIGGRAPH*, 355–361.

Horn, R. A., and Johnson, C. R. 1990. *Matrix Analysis*. Cambridge University Press, Cambridge.

Jolliffe, I. 1989. *Principal Component Analysis*. Springer-Verlag, New York.

Kharevych, L., Springborn, B., and Schröder, P. 2006. Discrete confrmal mappings via circle patterns. *ACM Transactions on Graphics 25*, 2, 412–438.

Kraevoy, V., and Sheffer, A. 2004. Cross-parameterization and compatible remeshing of 3D models. In *Proceedings of SIGGRAPH*, 861–869.

Lee, A., Sweldens, W., Schroder, P., Cowsar, L., and Dobkin, D. 1998. MAPS: Multiresolution adaptive parametrization of surfaces. In *Proceedings of SIGGRAPH*, 95–104.

Lévy, B., Petitjean, S., Ray, N., and Maillot, J. 2002. Least squares conformal maps for automatic texture atlas generation. In *Proceedings of SIGGRAPH*, 362–371.

Praun, E., and Hoppe, H. 2003. Spherical parameterization and remeshing. In *Proceedings of SIGGRAPH*, 340–350.

Praun, E., Sweldens, W., and Schroder, P. 2001. Consistent mesh parameterizations. In *Proceedings of SIGGRAPH*, 179–184.

Ray, N., Li, W. C., Levy, B., Sheffer, A., and Alliez, P. 2006. Periodic global parameterization. *ACM Transactions on Graphics 25*, 4, 1460–1485.

Roweis, S., and Saul, L. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science 290*, 2323–2326.

Sander, P. V., Snyder, J., Gortler, S. J., and Hoppe, H. 2001. Texture mapping progressive meshes. In *Proceedings of SIGGRAPH*, 409–416.

Sander, P., Wood, Z., Gortler, S., Snyder, J., and Hoppe, H. 2003. Multi-chart geometry images. In *Proceedings of Symposium on Geometry Processing*, 138–145.

Sheffer, A., and de Sturler, E. 2001. Parametrization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers 17*, 3, 326–337.

Sheffer, A., Lévy, B., Mogilnitsky, M., and Bogomyakov, A. 2005. ABF++: Fast and robust angle based flattening. *ACM Transactions on Graphics 24*, 2, 311–330.

Tenenbaum, J., de Silva, V., and Langford, J. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science 290*, 2319–2323.

Toledo, S. 2003. Taucs: a library of sparse linear solvers. In *http://www.tau.ac.il/ stoledo/taucs*. Tel-Aviv University, version 2.2.

Welch, W., and Witkin, A. 1994. Free-form shape design using triangulated surfaces. In *Proceedings of SIGGRAPH*, 247–256.

Zayer, R., Rössl, C., , and Seidel, H.-P. 2005. Setting the boundary free: A composite approach to surface parameterization. In *Proceedings of Eurographics Symposium on Geometry Processing*, 91–100.

Zhang, Z., and Zha, H. 2005. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM journal on scientific computing 26*, 11, 313–338.

Zhang, E., Mischaikow, K., and Turk, G. 2005. Feature-based surface parameterization and texture mapping. *ACM Transaction on Graphics 24*, 1, 1–27.

Zigelman, G., Kimmel, R., and Kiryati, N. 2002. Texture mapping using surface flattening via multidimensional scaling. *IEEE Transactions on Visualization and Computer Graphics 8*, 2, 198–207.