Iterative Methods for Improving Mesh Parameterizations

Shen Dong
University of Illinois, Urbana-Champaign
201 N Goodwin Ave
Urbana, IL 61801, USA
shendong@uiuc.edu

Michael Garland
NVIDIA
2701 San Tomas Expressway
Santa Clara, CA 95050, USA
mgarland@nvidia.com

Abstract

We present two complementary methods for automatically improving mesh parameterizations and demonstrate that they provide a very desirable combination of efficiency and quality. First, we describe a new iterative method for constructing quasi-conformal parameterizations with free boundaries. We formulate the problem as fitting the coordinate gradients to two guidance vector fields of equal magnitude that are everywhere orthogonal. In only one linear step, our method efficiently generates parameterizations with natural boundaries from those with convex boundaries. If repeated until convergence, it produces the unique global minimizer of the Dirichlet energy. Next, we introduce a new non-linear optimization framework that can rapidly reduce interior distortion under a variety of metrics. By iteratively solving linear systems, our algorithm converges to a high quality, low distortion parameterization in very few iterations. The two components of our system are effective both in combination or when used independently.

1 Introduction

Methods for automatically constructing surface parameterizations are essential tools for computer graphics. Many techniques in common use—including texture mapping, parametric remeshing, and finite element simulation—rely on the availability of high quality surface parameterizations. Several effective parameterization methods have been proposed in recent years. In this paper, we introduce a pair of new methods that can efficiently produce high quality parameterizations by iterative improvement.

We begin by developing a method to compute a free boundary quasi-conformal parameterization from one with prescribed boundary. The system is derived from a variational problem of fitting the gradients of the parametric coordinate functions to two orthogonal guidance vector fields. This produces an iterative method which, in a single step, can significantly reduce parametric distortion by finding a more "natural" boundary shape. Our method generates output that has considerably lower distortion, both in areas and angles, than other linear methods that directly solve for a free boundary conformal parameterization [15, 3]. It is also more efficient than recent linear methods [27], which require two improvement steps rather than one to achieve a satisfactory boundary shape.

Next, we present a method for rapidly optimizing nonlinear energy functions to generate parameterizations low in both area and angle distortion. By using the mean value weights derived from the local minimizer of the energy functions and simultaneously optimizing the parametric values of all vertices, we obtain a very high convergence rate. Compared with recent methods [26] that also efficiently produce parameterizations with low distortion, we are able to reduce the area distortion much further in a similar time frame while keeping the angle distortion low.

2 Background

We are interested in constructing a parameterization of a triangulated simplicial 2-manifold patch M represented by its set of vertices V, edges E, and faces F. We assume that the manifold has one or more boundary loops and that it is embedded in a 3-dimensional Euclidean space by a coordinate function $\mathbf{x} : V \to \mathbb{R}^3$. The parameterization we seek is represented by a mapping $\phi : D \subset \mathbb{R}^2 \to V$ that assigns coordinates (u_i, v_i) to each vertex $i \in V$. From the piecewise linear coordinate functions $u, v : M \to D$ we can construct two gradient vector fields $\nabla u, \nabla v : F \to \mathbb{R}^3$ that are constant over each triangle of M.

A number of methods have been developed to automatically construct such parameterizations. For a comprehensive overview, we recommend the survey of Floater and Hormann [7].

Linear Parameterization Methods. Floater [5] demonstrated that convex combination maps can be used to auto-

matically produce parameterizations by solving a single linear system. The resulting parameterization is provably bijective provided that the boundary is mapped onto a convex polygon. He subsequently developed a much simpler set of coefficients using the mean value property of harmonic functions [6].

Several authors have investigated harmonic maps [18, 4, 9] as a way to obtain conformal parameterizations because of the equivalence in minimizing the Dirichlet energy and the conformal energy. Desbrun *et al.* [3] extended the analysis to allow for unconstrained boundaries. Lévy *et al.* [15] introduced a somewhat different approach to conformal parameterization based on a discretization of the Cauchy-Riemann equations. Both of these conformal techniques can be used to produce "natural" boundary shapes.

Zayer *et al.* [26] extended the usual linear system construction by solving a quasi-harmonic problem. Their approach reduces the distortion of an initial convex parameterization by computing a mapping of the plane onto itself that reproduces, as nearly as possible, the Jacobian of the initial parameterization. In effect, it tries to find the most isometric (least squares sense) mapping. Their approach also accommodates unconstrained boundaries [27].

Non-Linear Methods. Non-linear optimization methods can generally be used to produce parameterizations with lower distortion than the linear methods just mentioned, but at the cost of longer running times and greater implementation complexity.

The MIPS method by Hormann *et al.* [10] minimizes the Dirichlet energy per parameter area. The angle-based flattening (ABF) scheme by Sheffer and de Sturler [21] minimizes a parameterization's weighted sum of angle deviation from the surface to achieve results with very low angular (i.e., conformal) distortion. Sheffer *et al.* [22] subsequently devised a more efficient numerical method (ABF++) for solving the same underlying problem. Kharevych *et al.* [12] proposed solving global optimization problems on the intersection angles and the radii of the circumcircles of the triangles to produce a conformal parameterization.

It is often convenient to express parametric distortion measures in terms of the singluar values Γ, γ of the 3×2 Jacobian matrix $J\phi$. This approach can accommodate a number of different metrics [20, 23, 2]. Most previous techniques for minimizing such distortion metrics have focused on iteratively repositioning single vertices—by random line search [20], Newton's method [11], or conjugate gradient [2], for example—until convergence. Yoshizawa *et al.* [24] take a somewhat different approach, iteratively reweighting a convex combination linear system in order to diffuse distortion error over the mesh.

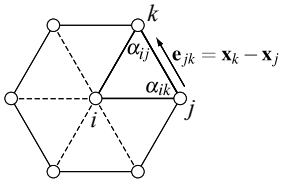


Figure 1. Edges & angles in *link* of vertex i.

3 Quasi-Conformal Parameterization with Free Boundary

Our goal in this stage is to compute a free boundary quasi-conformal parameterization of a given surface patch. Specifically, this means that we wish to find coordinate functions $u, v : V \to \mathbb{R}$ with the property that

$$\nabla v = \mathcal{R} \nabla u \tag{1}$$

where the operator $\mathcal{R}$ denotes a counter-clockwise rotation of 90° about the surface normal. This is simply one representation of the well-known Cauchy-Riemann equations. Note that for discrete meshes, no such parameterization exists unless M is developable. Thus in general, we wish to find a parameterization that is as conformal as possible.

3.1 Fitting to Guidance Gradient Fields

While we cannot expect to find coordinate functions that globally satisfy the Cauchy-Riemann equations (1), it is obviously trivial to find two vector fields $\mathbf{g}_1, \mathbf{g}_2 : F \to \mathbb{R}^3$ such that $\mathbf{g}_2 = \mathcal{R}\mathbf{g}_1$. Assuming for the moment that we can find a *good* pair of vector fields, we can frame the problem of computing the coordinate functions (u, v) as a variational fitting problem. We want to find (u, v) whose gradient fields most closely approximate the guidance fields $\mathbf{g}_1, \mathbf{g}_2$:

$$\min_{(u,v)} \int_M \|\nabla u - \mathbf{g}_1\|^2 + \|\nabla v - \mathbf{g}_2\|^2 \tag{2}$$

Ray *et al.* [19] propose a similar formulation, with the goal of fitting a global parameterization to the principal direction fields of the manifold.

It is well known [17, 25] that the Euler-Lagrange equations for this variational problem are simply a system of Poisson equations:

$$\Delta u = \operatorname{div} \mathbf{g}_1 \qquad \Delta v = \operatorname{div} \mathbf{g}_2 \tag{3}$$

To solve these systems, we use the usual discrete definitions of the divergence and Laplacian operators. The divergence of the tangent vector field $\mathbf{g}$ at a vertex $i \in V$ is given by:

$$\operatorname{div}_i \mathbf{g} = \sum_{(j,k) \in \operatorname{Lk} i} \mathbf{g} \cdot \mathcal{R}\mathbf{e}_{jk} \tag{4}$$

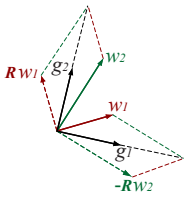


Figure 2. Going from arbitrary vector fields $\mathbf{w}_1, \mathbf{w}_2$ to orthogonal vector fields with equal magnitude $\mathbf{g}_1, \mathbf{g}_2$.

where $\mathrm{Lk}\, i$ is the *link* of the vertex—the set of all edges connecting vertices adjacent to i. By convention, we assume that the edges $(j,k) \in \mathrm{Lk}\, i$ are always oriented counterclockwise about the outward-facing surface normal.

Then the Laplacian $\Delta_i f$ of a function f at vertex $i \in V$ is defined as the divergence $\mathrm{div}_i(\nabla f)$:

$$\Delta_i f = \sum_{(j,k)\in \mathrm{Lk}\, i} \cot \alpha_{ij}(f_i - f_j) + \cot \alpha_{ik}(f_i - f_k) \quad (5)$$

Here α_{ij}, α_{ik} are angles opposite the edges (i,j) and (i,k), respectively, in the triangle (i,j,k), as shown in Figure 1.

3.2 Estimating Guidance Fields

Now the key question is *how* to pick the guidance vector fields $\mathbf{g}_1, \mathbf{g}_2$ to drive the computation of the coordinate functions (u,v). In general, constructing "good" guidance fields can be a complex non-linear optimization problem. In our case, we want to find a conformal mapping of the surface onto the plane. Therefore, we would like the pair of guidance fields to have the same properties as the gradient fields of a perfectly conformal mapping— everywhere orthogonal with equal magnitude. Given two *arbitrary* vector fields $\mathbf{w}_1, \mathbf{w}_2$, the following construction we propose is *guaranteed* to produce a pair of vector fields $\mathbf{g}_1, \mathbf{g}_2$ with these properties (Figure 2):

$$\mathbf{g}_1 = \frac{1}{2}(\mathbf{w}_1 - \mathcal{R}\mathbf{w}_2) \qquad \mathbf{g}_2 = \frac{1}{2}(\mathbf{w}_2 + \mathcal{R}\mathbf{w}_1) \quad (6)$$

Then we pick $\mathbf{w}_1, \mathbf{w}_2$ to be the gradient fields of a *constrained boundary* conformal parameterization because of their resemblance to the guidance fields of an ideal conformal parameterization. In other words, we use the DCP system of Desbrun *et al.* [3] to solve for coordinate functions (s,t) satisfying:

$$\begin{bmatrix} \Delta s \\ \Delta t \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (7)$$

with the boundary mapped to a convex shape—we use the unit circle in all our results. Then we let:

$$\mathbf{w}_1 = \nabla s \qquad \mathbf{w}_2 = \nabla t \quad (8)$$

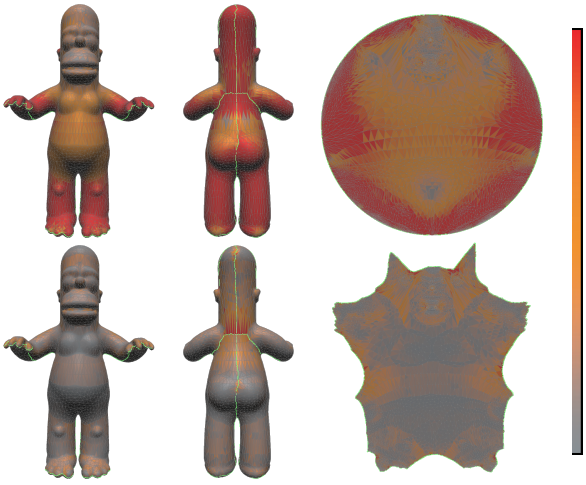


Figure 3. From an initial parameterization (top), we produce a result (bottom) with a natural boundary shape and much lower distortion in one linear step. The color ramp on the right maps conformal energy from $\Gamma/\gamma = 1$ (gray) to $\Gamma/\gamma = 2$ (red).

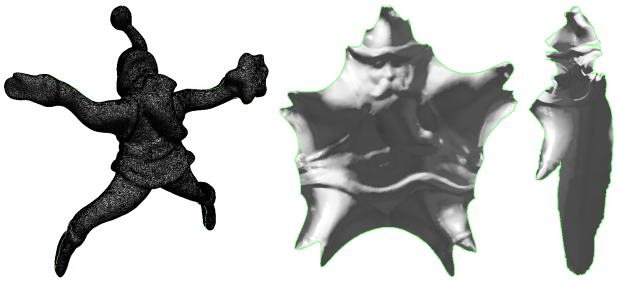
Note that even in areas where the initial parameterization has gradient fields $\nabla s, \nabla t$ that are far from conformal—implying a large angle distortion—our guidance fields $\mathbf{g}_1, \mathbf{g}_2$ are always orthogonal and equal in length. The fact that it is linear in $\nabla s, \nabla t$ is also quite convenient since it simplifies the final system considerably.

At this point, we can expand the Poisson system given in Eq. 3 using our specific guidance fields and the discrete divergence and Laplacian operators described above. This produces the following system of equations:

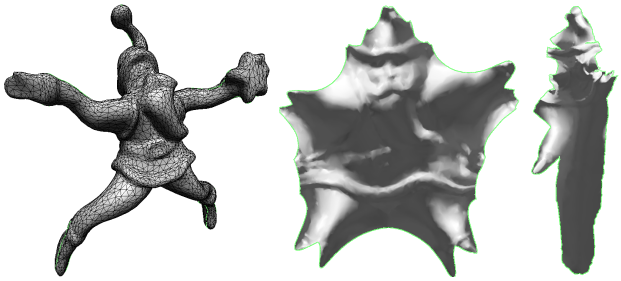
$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \frac{1}{2}\left(\begin{bmatrix} \Delta s \\ \Delta t \end{bmatrix} + \sum_{(j,k)\in \mathrm{Lk}\, i} \begin{bmatrix} t_j - t_k \\ s_k - s_j \end{bmatrix} \right) \quad (9)$$

For a mesh with n vertices, this gives two $n \times n$ sparse linear systems, one for u and one for v. The solution of the system is unique up to an additive constant. Thus we constrain one boundary vertex to have coordinates $(u,v) = (0,0)$.

Figure 3 shows an example of this process. The high level of distortion in the initial constrained boundary parameterization in the top row is effectively reduced after solving the linear system of Eq. 9 to get a parameterization in the bottom row with considerably better boundary layout. Also note that the left hand side of Eq. 9 is identical to the usual DCP system, and the right hand side has non-zero entries only in rows corresponding to boundary vertices, a fact that can be attributed to our choice of $\mathbf{w}_1, \mathbf{w}_2$. So augment-



(a) Model sampled with 114K triangles



(b) Model sampled with 11K triangles

Figure 4. Our method (center) produces very similar boundaries for both tessellations, whereas LSCM (right) has more noticeable differences and much higher area distortion.

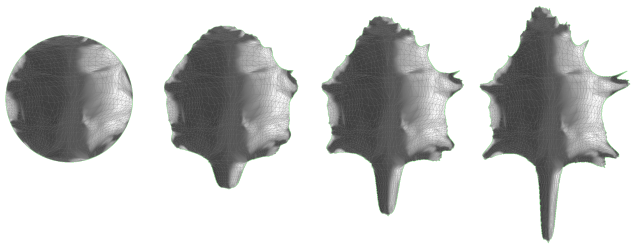
ing existing parameterization code with our free boundary method is a trivial task.

Our free boundary construction also has the added benefit of being fairly insensitive to different tessellations of the same surface shape. Figure 4 shows a surface that has been tessellated at very different resolutions, the original model with 114k faces and a QSlim [8] simplified model with 11k faces. Most significantly, the boundaries are complex and are sampled quite differently. Nevertheless, our method performs robustly, producing very similar boundary shapes. LSCM [15] produces more noticeable shape differences, in the character's left hand and right leg, for example, and has significantly higher area distortion.

Even though our free boundary scheme does not guarantee bijectivity, experiments show that triangle flips happen rarely, at a frequency very similar to that of the conventional fixed boundary discrete conformal parameterization.

3.3 Controlling Boundary Evolution

Our system can be generalized by introducing a parameter λ to control the degree of orthogonality of the guidance



(a) $\lambda = 0$ (b) $\lambda = \frac{1}{6}$ (c) $\lambda = \frac{1}{3}$ (d) $\lambda = \frac{1}{2}$

Figure 5. Flattening a cow with different values of the evolution speed parameter λ.

fields, and thus the speed of boundary evolution:

$$\mathbf{g}'_1 = (1-\lambda)\Delta s - \lambda \mathcal{R} \Delta t \tag{10}$$

$$\mathbf{g}'_2 = (1-\lambda)\Delta t + \lambda \mathcal{R} \Delta s \tag{11}$$

These generalized guidance fields lead to the generalized linear system:

$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = (1-\lambda) \begin{bmatrix} \Delta s \\ \Delta t \end{bmatrix} + \lambda \sum_{(j,k)\in \mathrm{Lk}\, i} \begin{bmatrix} t_j - t_k \\ s_k - s_j \end{bmatrix} \tag{12}$$

This generalization provides a way for the user to control the "regularity" of the boundary. An illustration of the effect of varying λ is shown in Figure 5. Selecting $\lambda = 0$ preserves the initial convex boundary. Increasing towards $\lambda = 1/2$ allows increasing boundary irregularity in order to minimize distortion. A user can therefore control the tradeoff between boundary irregularity and distortion. This kind of control is valuable when building multi-chart atlases, for instance, where the goal is to balance overall distortion with texture packing efficiency.

3.4 Iteration and Convergence

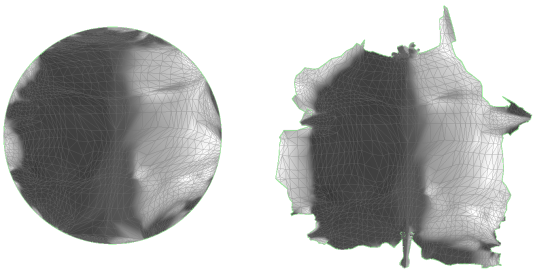
Our update procedure (Eq. 9) can obviously be applied iteratively. It will converge to a fixpoint parameterization (u^*, v^*) satisfying

$$\begin{bmatrix} \Delta u^* \\ \Delta v^* \end{bmatrix} = \sum_{(j,k)\in \mathrm{Lk}\, i} \begin{bmatrix} v_j^* - v_k^* \\ u_k^* - u_j^* \end{bmatrix} \tag{13}$$

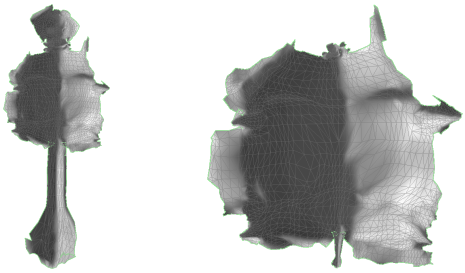
Desbrun *et al.* [3] have shown that such a parameterization attains the *unique global minimum* of the quadratic Dirichlet energy:

$$E_D = \frac{1}{2} \int_M \|\nabla u\|^2 + \|\nabla v\|^2 \tag{14}$$

Indeed, Eq. 13 is the system they derive for their natural boundary conformal parameterization (NDCP). However, it



(a) Iterative method starting from circular boundary



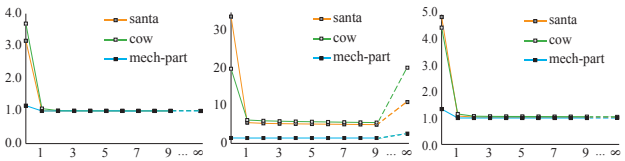
(b) Iterative method starting from NDCP boundary

Figure 6. Solving Eq. 9 iteratively converges to the same global minimum of the Dirichlet energy from very different initial states.

is critical to note that our fixpoint parameterization satisfies these equations at *every vertex*, whereas as the NDCP parameterization does *not* satisfy the equations at the two vertices whose locations are constrained to make the linear system non-singular.

A trivial difference as this may seem, it can produce a drastic difference in the results. As an example, Figure 6(b) compares the convergent state of our approach and the output of NDCP. Our result has a lower Dirichlet energy of $E_D = 1.00578$ as opposed to $E_D = 1.00649$ for the NDCP result. Figure 6 also demonstrates that from two very different starting points—a circular boundary one (top) and the NDCP boundary one (bottom)—our iterative process converges to the same global minimum. Note that when measuring Dirichlet energy, the parameterization is scaled to have the same area as the surface to avoid the Dirichlet energy approaching zero due to shrinkage.

At times it may be important to find the minimizer of the Dirichlet energy when angle distortion is the only consideration. However, this often come at a high price in terms of increased area distortion. Figure 7 shows the progression of these measures during our iterative process. We can see that the vast majority of measure reduction takes place during the first iteration, and that area distortion can increase significantly in later iterations. In practice, we recommend using a *single* improvement step. This generally yields more



(a) Dirichlet energy (b) Area distortion (c) Angle distortion

Figure 7. Distortion measured during iterative improvement. The first iteration clearly dominates in distortion reduction.

than 90% of the achievable improvement in distortion, and the cost in running time is usually not worth the incremental improvement in later iterations.

4 Fast Non-linear Optimization

The quasi-conformal method we have just described produces parameterizations with very low angle distortion, but possibly high area distortion. In many applications it is desirable to find a balance between area and angle distortions. To accomplish this, we propose an optimization framework that can quickly minimize a non-linear energy function by iteratively solving a very small number (usually 3∼5) of linear systems.

Our method is inspired by the observation that any parameterization ϕ can be obtained by solving *one single* linear system if given the appropriate weights. Specifically, if we use the mean value weights [6] of ϕ to fill a sparse convex combination matrix L, by the linear reproduction property, solving L should give us ϕ in a single step. Thus to find our target parameterization ϕ^* that minimizes an energy function E, we can instead try to find the corresponding matrix L^* that will produce ϕ^*.

Obviously we don't know L^* yet, but we know that each interior vertice of ϕ^* must be located at the minimum of E within its 1-ring. So starting from an initial parameterization ϕ_0, we approximate L^* by another matrix L_0 constructed in the following way. For each interior vertex i of ϕ_0, we compute the target location within its 1-ring that minimizes E (the 1-ring vertices doesn't change), from which we derive a set of mean value weights. Repeating this for every interior vertex, we assemble a convex combination matrix L_0 row by row.

By solving L_0, we get a new parameterization ϕ_1 that is much closer to ϕ^* than ϕ_0. Since all vertices are updated simultaneously in each step, iterating this process produces a series of parameterizations that converges quickly, and we know that the fixpoint must by construction be a (local) minimum of E. Bundary vertices are held fixed across iterations.

4.1 Finding Target Vertex Positions

To compute the minimum-energy position of a vertex within its 1-ring, we use the Nelder-Mead simplex method [16, 14]. This method will begin with a 2-simplex (i.e., a triangle) contained within the 1-ring of the vertex. The three corners of this simplex reflect potential target locations for the vertex. It iteratively applies one of five search operations—illustrated in Figure 8—to the simplex, making its way downhill and narrowing the region in which the optimal location may lie. The choice of move is guided by the relative energy function value at the corners of the simplex. The procedure terminates on convergence or after a maximum number of function evaluations have been performed.

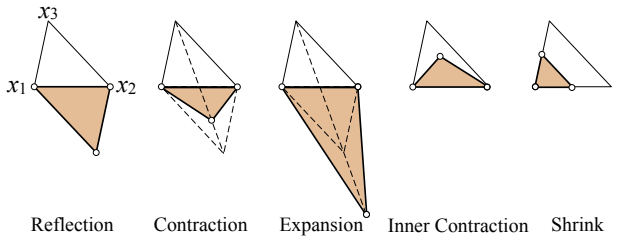


Figure 8. Possible search moves performed on simplex $x_1x_2x_3$.

The initial simplex is an equilateral triangle around the vertex's current position, with an edge length of $1/100$ the diameter of the 1-ring. We terminate the iteration when the total change in the location of simplex corners is less than $1/400$ the diameter, or the number of function evaluations exceeds 100. The corner of the simplex with the lowest energy value is then chosen as the target location.

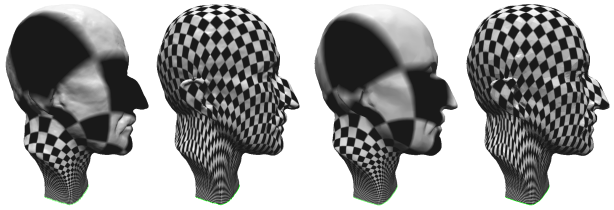
The Nelder-Mead method is a widely used optimization technique that has a number of attractive properties. It is very efficient, requiring on average only 10–20 objective function evaluations per vertex. The total time spent in local minimum search is almost always less than the time spent on solving the linear systems (see Table 2). As a direct method, it requires only function evaluations and no derivatives information, whose precise estimation can be very difficult to get. For convex energy functions, the Nelder-Mead method will provably converge [14] (in pathological cases to a non-minimizer). Finally, it has the added benefit of being trivial to implement.

4.2 Choice of Distortion Metric

Many distortion metrics are possible choices for the energy function E. As stated earlier, these are conveniently represented in terms of the singular values of the Jacobian matrix $J\phi$. We would like to minimize both area distortion ($\Gamma\gamma$) and angular distortion (Γ/γ). We have found that the combined metric proposed by Degener *et al.* [2] provides a good trade-off between these two factors. We use a variant of their metric suggested by Floater and Hormann [7]:

$$E = \sum_{i \in F} \left(\Gamma_i \gamma_i + \frac{1}{\Gamma_i \gamma_i} \right) \left(\frac{\Gamma_i}{\gamma_i} + \frac{\gamma_i}{\Gamma_i} \right) \text{Area}_i \qquad (15)$$

This metric balances area and angle distortion, avoids parameter cracks, and penalizes both shrinking and stretching. Being a convex function, our Nelder-Mead optimizer is also guaranteed to converge with this choice of E.



(a) 200K model (b) 10K model

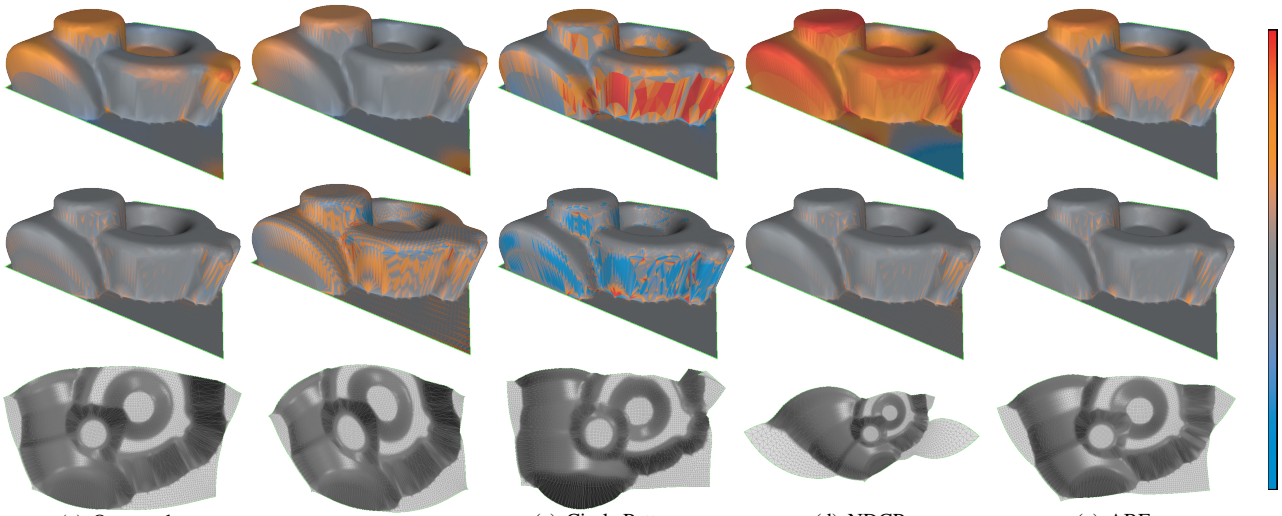
Figure 9. Our non-linear optimization substantially reduces distortion in only 4 iterations, and does so equally well at different resolutions.

Our method reduces the value of the energy metric very rapidly (see Table 1). A very small number of iterations are generally needed even for highly non-planar models or models with large size (see Figure 9).

Our optimization process is also very robust to poor inputs. Figure 10 demonstrates starting the iteration with convex combination maps computed from (1) mean value weights and (2) randomly generated weights. Our optimization eliminates the noises of the parameterization very quickly, and the two cases converge to very similar results.

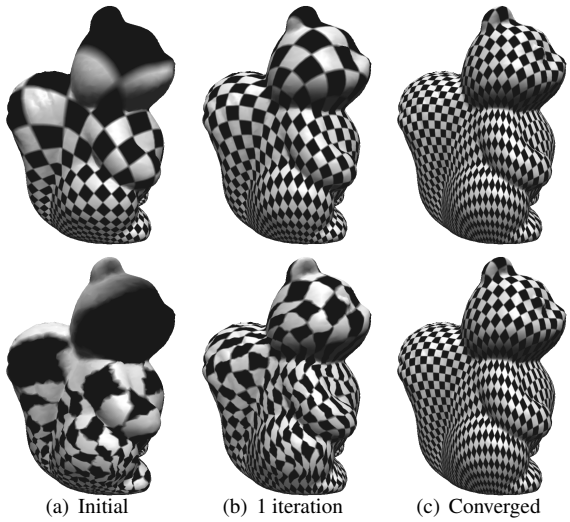
Since the mean value weights are always positive, the parameterization we get after any number of iterations is guaranteed to be bijective for a non-concave boundary. For complex concave boundaries generated by our free boundary parameterization method, our non-linear optimization scheme also performs very robustly.

Table 2 summarizes the performance of our non-linear method. It reports total number of iterations, average function evaluations per vertex, and running time for finding the target vertex location and total time. Note that very few iterations are required for convergence, and the running time is quite modest. We use the UMFPACK [1] solver for our linear systems. Since all matrices used during iteration have the same non-zero pattern, the symbolic factorization can be reused, further improving efficiency.



(a) Our result (b) DTQH (c) Circle Patterns (d) NDCP (e) ABF

Figure 11. Comparison of free boundary parameterization results in terms of area distortion (first row) and angle distortion (second row).



(a) Initial (b) 1 iteration (c) Converged

Figure 10. Whether starting with mean value (top) or random (bottom) convex combination maps, our system quickly converges to the same low distortion result.

		Iteration				
Model	Init.	1	2	3	···	∞
Squirrel (a)	18.57	3.97	3.00	2.77		2.65
Squirrel (b)	92.23	4.85	3.27	2.86		2.65
Planck	23.38	6.09	4.09	3.43		3.03
Planck	24.31	5.62	3.86	3.31	···	3.11
Cow head	254.84	6.47	3.63	3.15		2.93

Table 1. Nelder-Mead convergence.

				Time (s)	
Model	# Faces	Iter.	Avg. Func. Eval.	Opt.	Total
Squirrel (a)	18K	1	18.05	0.25	0.60
Squirrel (b)	18K	1	50.83	0.65	1.01
Planck	10K	4	16.35	0.50	1.02
Planck	200K	4	11.43	7.33	22.84
Cow head	8.5K	4	18.32	0.47	0.95

Table 2. Efficiency of non-linear optimization.

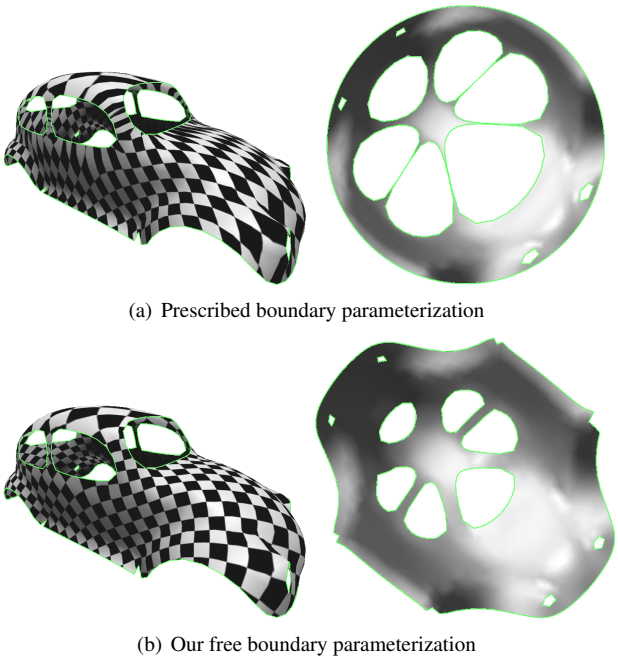
5 Results

We begin our performance analysis with the example shown in Figure 11. This figure compares our linear method with other conformal methods for parameterizing surfaces with unconstrained boundaries. We measure area (angle) distortion by the ratio of the area of the triangles (magnitude of the angles, 3 per face) in the parameter domain to their counterparts on the surface, then visualize the per face (per angle) distortion with a red-to-blue color scale shown on the right. For area (angle) distortion, red corresponds to an undersampling ratio of 4 (1.5) and blue corresponds to an oversampling ratio of 4 (1.5). As we can see, the linear NDCP [3] method may produce result with boundary shapes that are far from optimal, thus leading to high area distortions. For regions that are far from Delaunay, the circle patterns method [12] may produce high distortion and triangle flips. The discrete tensorial quasi-harmonic (DTQH) method of Zayer *et al.* [27] gives a much nicer boundary shape, but suffers from angle distortion caused by shearing. Our method produces result that is low in both area and angle distortion with an ideal boundary shape. The more expensive non-linear ABF method [22] generally produce results with the lowest angle distortion, yet it does not take into account area distortion.

In Figure 12, we demonstrate the capability of our method in handling models with multiple boundary loops. Notice that the initial parameterization is severely distorted near the interior boundaries. Our parameterization method is able to correct them and produce natural boundary shapes in a single step.

Figure 13 compares the performance of our non-linear optimization technique with DTQH [26], which like our method, iteratively solves linear systems to reduce distortion. Both results are produced using the same number of iterations (i.e., linear system solutions). For area (angle) distortion, red corresponds to an undersampling ratio of 5 (2) and blue corresponds to an oversampling ratio of 5 (2). Our method achieves a significantly lower area distortion—an average of 1.62 vs. 11.06—while having comparable angle distortion— an average of 1.33 vs. 1.13.

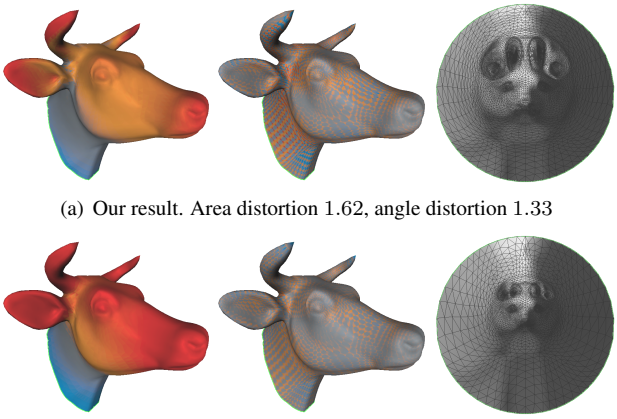
Figure 14 compares our non-linear optimization method with the traditional per-vertex relaxation optimization method used by Sander *et al.* [20]. Here both methods are used to minimize their L_2 distortion metric. The left column shows an intermediate state of each minimization process with similar L_2 error. Notice that in our result distortion is distributed much more evenly, leading to much more smoothly varying checkerboard pattern, especially around the ear. The right column shows the convergent states of the two methods, and our simultaneous optimization scheme is able to achieve a noticeably lower L_2 distortion than the local relaxation method: 1.428 vs. 2.682. Note that the pa-



(a) Prescribed boundary parameterization

(b) Our free boundary parameterization

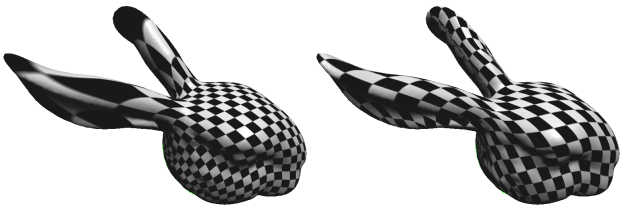
Figure 12. Our method handles models with multiple interior boundaries well.



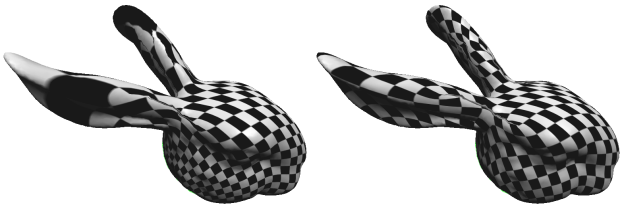
(a) Our result. Area distortion 1.62, angle distortion 1.33

(b) DTQH result. Area distortion 11.06, angle distortion 1.13

Figure 13. Comparison of area distortion (left) and angle distortion (center) of the cow head results.



(a) Our method for minimizing the L_2 metric



(b) Local relaxation method in Sander *et al.* [20]

Figure 14. In an intermediate state with equal L_2 metric (left), we achieve more evenly distributed distortion. Our convergent state (right) also has a lower L_2 metric of 1.428 than the 2.682 of [20].

rameter crack in this case is a well-known artifact of the L_2 energy itself.

Figure 15 illustrates the quality and flexibility of iteratively performing our linear free boundary parameterization scheme. After one iteration (at left), the area/angle distortions are reduced from $10.00/2.46$ for the initial circular boundary to $4.07/1.07$. If we iterate until either area or angle distortion no longer decreases—65 iterations in this example—the area/angle distortions are further reduced to $2.93/1.05$. The application has the flexibility to decide whether the additional improvement is worth the additional running time, and may terminate the iteration at any time.

Figure 16 demonstrates the coupling of the two methods we propose. Starting from a DCP parameterization with circular boundary, we first apply our boundary improvement method. Then the result is used as input to our non-linear optimization framework, which improves the interior while keeping the boundary fixed. The area distortion is reduced from an initially very high 239.52 to a mere 2.88. In this paper we use cut models provided to us as is. For the few models that are closed, we cut out a a simple and natural boundary curve.

6 Conclusion

We have presented a new system for parameterizing surfaces consisting of two components: a linear quasi-conformal method with unconstrained boundaries and a lin-

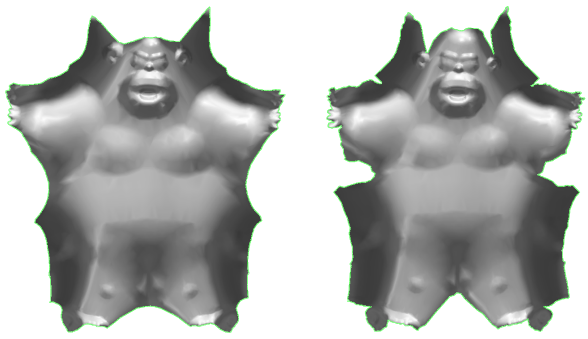
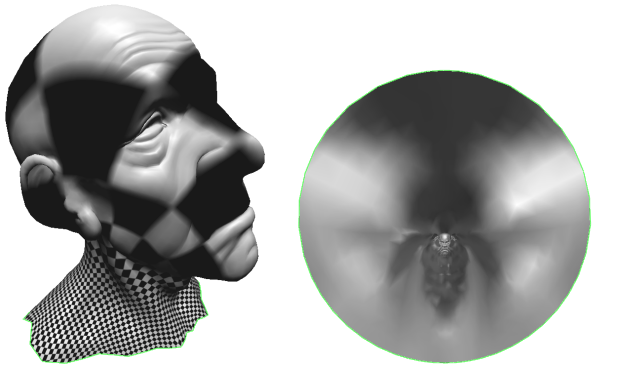
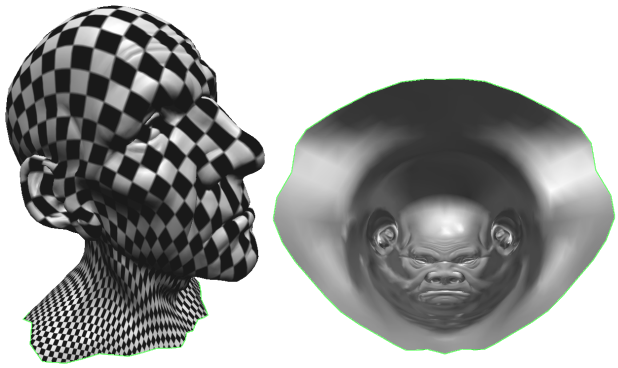


Figure 15. Performing 1 step of boundary improvement (left) vs. iterating until distortion no longer decreases (right).



(a) Initial DCP parameterization



(b) Output parameterization

Figure 16. Combining our boundary improvement and non-linear optimization methods.

ear framework for minimizing more general non-linear distortion metric. Both phases are efficient, easy to implement, and produce high quality results. Compared to similar methods, our scheme is generally more efficient and produces results with lower distortion.

We believe that this work could be extended in a number of useful ways. Our linear phase could be extended to produce global parameterizations by the use of appropriate cross-boundary transition functions [13, 19]. It would be of large benefit to allow the non-linear optimization framework to optimize the boundary as well as the interior of a parameterization. Further analysis should also produce insight into the class of distortion measures that can be successfully and stably optimized by this approach.

References

[1] T. A. Davis and I. S. Duff. An unsymmetric-pattern multifrontal method for sparse LU factorization. *SIAM Journal on Matrix Analysis and Applications*, 18(1):140–158, 1997.

[2] P. Degener, J. Meseth, and R. Klein. An adaptable surface parameterization method. In *Proc. 12th Int'l Meshing Roundtable*, pages 201–213, Sept. 2003.

[3] M. Desbrun, M. Meyer, and P. Alliez. Intrinsic parameterizations of surface meshes. In *Eurographics 2002 Conference Proceedings*, pages 209–218, 2002.

[4] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. *Proceedings of SIGGRAPH'95*, pages 173–182, 1995.

[5] M. S. Floater. Parameterization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231–250, 1997.

[6] M. S. Floater. Mean value coordinates. *Computer Aided Geometric Design*, 20(1):19–27, Mar. 2003.

[7] M. S. Floater and K. Hormann. Surface parameterization: a tutorial and survey. In N. A. Dodgson, M. S. Floater, and M. A. Sabin, editors, *Multiresolution in Geometric Modelling*. Springer, 2004.

[8] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *Proc. SIGGRAPH '97*, pages 209–216, 1997.

[9] S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro, and M. Halle. Conformal surface parameterization for texture mapping. In H. Hagen, editor, *IEEE Transactions on Visualization and Computer Graphics*, volume 6 (2), pages 181–189. IEEE Computer Society, 2000.

[10] K. Hormann and G. Greiner. MIPS: An efficient global parametrization method. In P.-J. Laurent, P. Sablonnière, and L. L. Schumaker, editors, *Curve and Surface Design: Saint-Malo 1999*, Innovations in Applied Mathematics, pages 153–162. Vanderbilt University Press, Nashville, TN, 2000.

[11] K. Hormann, U. Labsik, and G. Greiner. Remeshing triangulated surfaces with optimal parameterizations. *Computer-Aided Design*, 33(11):779–788, Sept. 2001.

[12] L. Kharevych, B. Springborn, and P. Schroder. Discrete conformal mappings via circle patterns. *ACM Trans. Graph.*, 25(2):412–438, 2006.

[13] A. Khodakovsky, N. Litke, and P. Schröder. Globally smooth parameterizations with low distortion. *ACM Transactions on Graphics*, 22(3):350–357, July 2003. Proc. SIGGRAPH 2003.

[14] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM J. Optimization*, 9(1):112–147, 1996.

[15] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics*, 21(3):362–371, July 2002. (Proc. SIGGRAPH 2002).

[16] J. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[17] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Transactions on Graphics*, 22(3):313–318, 2003. (Proc. SIGGRAPH 2003).

[18] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

[19] N. Ray, W. C. Li, B. Levy, A. Sheffer, and P. Alliez. Periodic global parameterization. *ACM Transactions on Graphics*, 2005. Submitted.

[20] P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe. Texture mapping progressive meshes. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 409–416, 2001.

[21] A. Sheffer and E. de Sturler. Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers*, 17(3):326–337, Oct. 2001.

[22] A. Sheffer, B. Lévy, M. Mogilnitsky, and A. Bogomyakov. ABF++: Fast and robust angle based flattening. *ACM Trans. Graph.*, 24(2):311–330, 2005.

[23] O. Sorkine, D. Cohen-Or, R. Goldenthal, and D. Lischinski. Bounded-distortion piecewise mesh parameterization. In *VIS '02: Proceedings of the conference on Visualization '02*, pages 355–362, 2002.

[24] S. Yoshizawa, A. Belyaev, and H.-P. Seidel. A fast and simple stretch-minimizing mesh parameterization. In *Proc. Shape Modeling Applications 2004*, pages 200–208, June 2004.

[25] Y. Yu, K. Zhou, D. Xu, X. Shi, H. Bao, B. Guo, and H.-Y. Shum. Mesh editing with Poisson-based gradient field manipulation. *ACM Transactions on Graphics*, 23(3):644–651, 2004. Proc. SIGGRAPH 2004.

[26] R. Zayer, C. Rossl, and H.-P. Seidel. Discrete tensorial quasi-harmonic maps. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005 (SMI' 05)*, pages 278–287, 2005.

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