Scalable Locally Injective Mappings

Michael Rabinovich[1] Roi Poranne[1] Daniele Panozzo[1,2] Olga Sorkine-Hornung[1]
[1]ETH Zurich [2]New York University

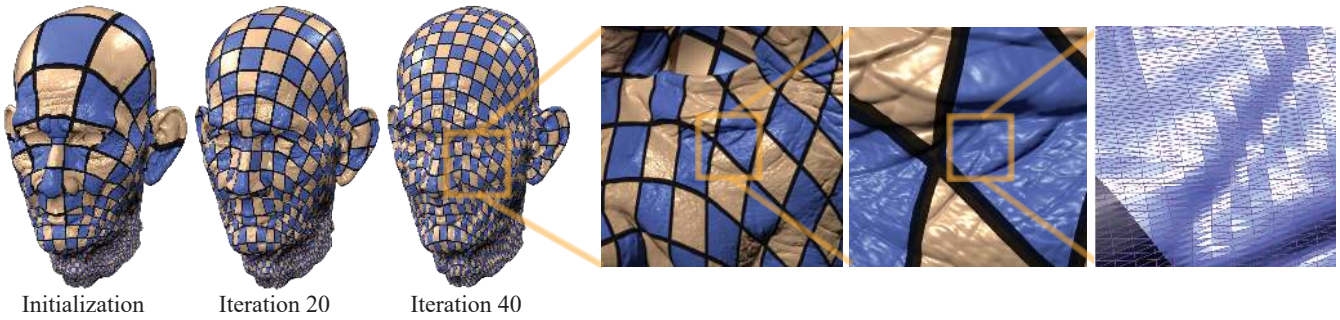


Figure 1: *A locally injective, energy minimizing parameterization of a mesh with over 25 million triangles computed with our algorithm in 80 minutes. The algorithm starts from a highly distorted locally injective initialization and in only 40 iterations, each requiring to solve a sparse linear system, it converges to a highly isometric map that is guaranteed to be free of inverted elements.*

Abstract

We present a scalable approach for the optimization of flip-preventing energies in the general context of simplicial mappings, and specifically for mesh parameterization. Our iterative minimization is based on the observation that many distortion energies can be optimized indirectly by minimizing a simpler proxy energy and compensating for the difference with a reweighting scheme. Our algorithm is simple to implement and scales to datasets with millions of faces. We demonstrate our approach for the computation of maps that minimize a conformal or isometric distortion energy, both in two and three dimensions. In addition to mesh parameterization, we show that our algorithm can be applied to mesh deformation and mesh quality improvement.

1 Introduction

Mappings are an essential tool in computer graphics and geometry processing. One of the most basic uses, and the main focus of this paper, is *mesh parameterization*. Many practical applications such as texture mapping, remeshing, shape correspondence and attribute transfer rely on the computation of a low-distortion parameterization. The problem has been extensively studied, and a plethora of algorithms have been devised. Linear methods were proposed first, providing efficient ways to compute parameterizations, but only able to ensure injectivity of the map when the mesh boundary is fixed a priori, which induces a high distortion. As more powerful processors became available, nonlinear optimization became tractable, allowing to compute free boundary, injective or bijective maps of a very high quality. Still, current nonlinear approaches typically require long computation times and do not scale well to large datasets, such as detailed scanned surfaces.

In this paper, we propose a simple algorithm that combines the benefits of the two approaches: it minimizes state-of-the-art nonlinear energies and processes meshes with millions of triangles within minutes. In particular, we focus on minimizing *flip preventing* energies that infinitely penalize element inversion. On medium-sized meshes, our algorithm is two orders of magnitude faster than competing methods, while being able to minimize many different energies with minimal code changes.

The key idea of our method is to replace the nonlinear energy with a much simpler proxy energy that is easy to minimize with a local/global approach. The difference between the proxy and the original energy is then accounted for in a reweighting scheme that converges to a stationary point of the original energy. Our algorithm scales well to large datasets even using a single core, and it can take advantage of advancements in parallel solution of linear systems, scaling almost linearly as more cores are used. While we are unable to provide a strict bound on the convergence rate, we experimentally found that the number of iterations required by our method is related to the geometric surface complexity and is not affected by the tessellation density; since each iteration only requires solving a sparse linear system, massive datasets can be parameterized quickly (Figure 1).

2 Previous work

Mappings are one the most researched subjects in computer graphics, and specifically the problem of generating locally injective 2D and 3D mappings has garnered a lot of attention in the past decades. In this section, we mention only the most closely related work on the topic of large scale mesh parameterization and we refer to [Floater and Hormann 2005; Sheffer et al. 2006] for an in-depth survey on mesh parameterization techniques.

Linear methods. Linear methods compute a mesh parameterization by solving a linear system, where each mesh vertex is represented as a weighted average of its neighbors. They have been proposed to parameterize topological disk patches [Tutte 1963] or topological spheres [Aigerman and Lipman 2015]. For topological disks, linear methods can be guaranteed to produce bijective parameterizations if the patch boundary is fixed to a convex shape and the weights are positive [Floater 2003]. Free-boundary methods exist that minimize a measure of conformal distortion [Desbrun et al. 2002; Lévy et al. 2002; Zayer et al. 2007; Ben-Chen et al. 2008; Mullen et al. 2008], but they are not guaranteed to produce a bijective map.

Nonlinear methods. Many nonlinear deformation energies have been proposed in the literature for both conformal and isometric

distortion; they are typically minimized using standard optimization methods, such as Newton [Sheffer and de Sturler 2001; Chao et al. 2010], quasi-Newton [Smith and Schaefer 2015] and second-order cone programming [Aigerman et al. 2014]. To simplify implementation and reduce memory usage, several works [Labsik et al. 2000; Hormann and Greiner 2000; Schreiner et al. 2004] opt for a block descent optimization, where in each iteration only a single vertex is free to move. Similarly, [Levi and Zorin 2014; Fu et al. 2015] optimize an independent subset of vertices in parallel. These methods do not scale to large datasets since the number of iterations they need grows quickly with the size of the mesh. In contrast, we observe that the number of iterations required by our method is related to the geometric complexity and not the dataset size: even datasets with millions of elements can be parameterized within a few iterations.

The local/global minimization of isometric distortion [Sorkine and Alexa 2007; Liu et al. 2008] iteratively alternates between a local step and a global step. In the local step, each element is individually perfectly mapped (without any distortion), and in the global step, a linear system is solved to stitch all elements back together. This process recovers from bad initialization very quickly, but it is slow to converge to a local minimum when it is close to it. The decoupling of a local condition from a global "stitching" has been successfully used in other parameterization algorithms [Weber et al. 2012] and to enforce complex constraints [Bouaziz et al. 2012; Poranne et al. 2013]. Our method uses the local/global paradigm and enriches it with a reweighting scheme to efficiently minimize flip preventing energies.

Non-flipping invariant. A recent series of works [Schüller et al. 2013; Fu et al. 2015; Smith and Schaefer 2015] have proposed parameterization energies with a term that goes to infinity when an element inverts. These *flip preventing* energies are minimized starting from a flipless initialization (e.g., [Tutte 1963]) using line search to ensure that they never leave the feasible region. This approach is guaranteed to create a locally injective map given a feasible starting point, but the energies are numerically difficult to optimize, leading to high running times. Our algorithm is specifically designed to optimize these energies and it quickly recovers from the highly distorted starting point.

Bounding, projections and stiffening. Another approach to the creation of locally injective maps is based on directly *bounding* the distortion. Similar to the above, first, a parameterization algorithm is used to generate an initial locally injective map. The energy of the map is then optimized while adhering to a specified distortion bound [Lipman 2012; Kovalsky et al. 2014; Chen and Weber 2015]. A similar, more recent approach projects any map, possibly with flips, to the closest map that has no flips [Aigerman and Lipman 2013; Kovalsky et al. 2015]. The major problem with these approaches is that the elements in the solution they generate have suboptimal distortion. In fact, their distortion tends to be the highest possible without violating the bound (Figure 2). Additionally, a valid solution is not guaranteed to be found using the projection approach, and it often takes thousands of iterations to look for one [Myles et al. 2014].

Another strategy is the so-called *stiffening*, where the idea is to try and coerce inverted elements to reorient correctly. Examples include [Irving et al. 2004], where an added force acts on inverted elements in a simulation, and [Martin et al. 2011], where a volume term is added to the energy of each triangle. Another method related to ours is found in [Bommes et al. 2009]: there, the inverted elements are reweighted in the original parameterization energy, as opposed to minimizing the original energy in the space of locally injective maps.

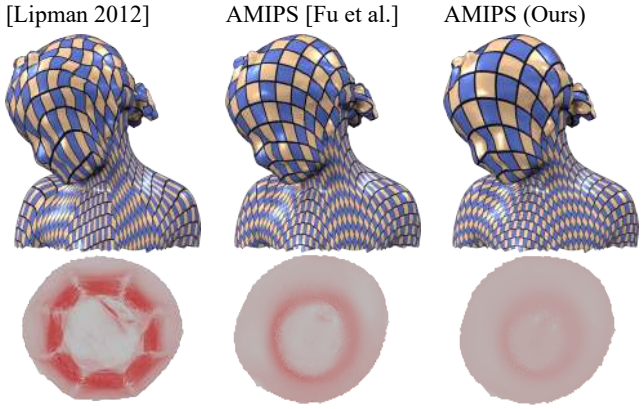


Figure 2: *The bounded ARAP energy (left, result taken from [Fu et al. 2015]) pushes the triangles to the distortion bound, while a direct minimization of the AMIPS energy [Fu et al. 2015] evenly distributes the distortion over the surface (middle). Our approach can also optimize the AMIPS energy (right), further reducing the energy to a local minimum. This model has 43K faces and required less than 20 iterations to converge. Bottom row: the distortion magnitude is visualized by the saturation of the red color.*

Large scale. Computing locally injective maps with a large number of elements is a challenging numerical problem that has been tackled in surprisingly few research papers in graphics. Apart from linear methods that scale well to large problems since they only involve the solution of a linear system, the only other works we are aware of are ABF++ [Sheffer et al. 2005] and [Kovalsky et al. 2015]. The former uses a multiresolution hierarchy to make the problem tractable at the expense of a higher distortion compared to the original ABF; it can optimize only for angle preservation. Kovalsky et al. [2015] proposes a projection method, and is thus not minimizing map distortion (Figure 2), making it impractical for some graphics applications. Our method is the first algorithm that can robustly and efficiently compute locally injective maps that contain millions of elements and minimize a conformal or isometric distortion measure.

Other applications. In addition to parameterization, our algorithm can be used to deform 3D objects and scale to massive 3D datasets. We can adapt our method to minimize mesh improvement energies, as suggested in [Lipman 2012; Aigerman and Lipman 2013; Fu et al. 2015]. "Seamless" rigid and conformal parameterizations (i.e., parameterizations whose gradients match across seams, see [Myles and Zorin 2012; Myles et al. 2014; Diamanti et al. 2015]) are also supported, although we cannot enforce the integer translations that are necessary for remeshing applications [Bommes et al. 2012].

3 Algorithm

Denote the input triangle or tetrahedral mesh by $M = (V, F)$, where V is the set of vertices and F is the set of elements. A common way to define a distortion energy of a mapping is via a function of the Jacobians of the parameterization, measuring a distance of each Jacobian to its closest rotation. We denote the Jacobian of element $f \in F$ by $\mathbf{J}_f$. Then the energy we wish to minimize is

$$\min_{\mathbf{x}} \quad \sum_{f \in F} A_f \, \mathcal{D}(\mathbf{J}_f(\mathbf{x})), \tag{1}$$

where $\mathcal{D}(\cdot)$ is the distortion measure, $\mathbf{x} \in \mathbb{R}^{|V| \times d}$ contains the mapping coordinates of the vertices ($d = 2, 3$) and A_f is the area

(or volume) of element f. A popular choice is the As-Rigid-As-Possible (ARAP) measure [Liu et al. 2008], which is defined by

$$\mathcal{D}(\mathbf{J}_f(\mathbf{x})) = \|\mathbf{J}_f(\mathbf{x}) - \mathbf{R}(\mathbf{J}_f(\mathbf{x}))\|_F^2 \tag{2}$$

where $\mathbf{R}(\mathbf{J}_f(\mathbf{x}))$ is the closest rotation to $\mathbf{J}_f(x)$, and $\|\cdot\|_F$ denotes the Frobenius norm. Note that the Jacobians are linear functions of $\mathbf{x}$.

Local/Global optimization. Liu *et al.* [2008] proposed to minimize the ARAP energy using a *local/global* algorithm, an iterative process that alternates between two steps. The local step finds at each iteration k the closest rotation to each Jacobian, that is $\mathbf{R}_f^k := \mathbf{R}(\mathbf{J}_f^{k-1}(\mathbf{x})) = \mathbf{U}\mathbf{V}^\top$, where $\mathbf{J}_f^{k-1}(\mathbf{x}) = \mathbf{U}\mathbf{S}\mathbf{V}^\top$ is the *signed* SVD of $\mathbf{J}_f^{k-1}(\mathbf{x})$. The global step solves (1) with the distortion $\mathcal{D}$ defined as in (2), assuming the rotations are fixed to $\mathbf{R}_f^k$, namely,

$$\mathbf{x}^{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \quad \sum_{f\in F} A_f \|\mathbf{J}_f(\mathbf{x}) - \mathbf{R}_f^k\|_F^2 \,. \tag{3}$$

This is a simple quadratic energy, which can be minimized by solving a linear system (see [Liu et al. 2008]).

Flip preventing energies. The ARAP energy above does not sufficiently penalize inverted elements, hence flips can occur frequently [Civit-Flores and Susin 2014; Martinez Esturo et al. 2014]. However, the local/global algorithm, which has been designed to minimize that energy, has the interesting property of making very large steps when it is far from the solution (e.g., when the parametrization is initialized with Tutte's parameterization, see Figure 4). A natural question would be: can we adapt this algorithm to other distortion energies? Given the shortcomings of ARAP, one would be interested in quickly minimizing flip preventing energies. One such example is the *symmetric* Dirichlet energy [Schreiner et al. 2004; Smith and Schaefer 2015], defined by $\|\mathbf{J}_f(\mathbf{x})\|_F^2 + \|\mathbf{J}_f^{-1}(\mathbf{x})\|_F^2$, and other examples we experimented with are summarized in Table. 1. In the following we show how to adapt the local/global algorithm to these energy.

Stiffening and reweighting. A simple approach is to enhance the global step with weights that make the quadratic energy resemble the one we wish to minimize. This idea is closely related to the stiffening often used in global parametrization [Bommes et al. 2009], where (3) is modified to include scalar per-face weights w_f, i.e.,

$$\mathbf{x}^{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \quad \sum_{f\in F} A_f \, w_f^k \|\mathbf{J}_f(\mathbf{x}) - \mathbf{R}_f^k\|_F^2. \tag{4}$$

Intuitively, the idea is to put more weight in the energy on elements with higher distortion, in the hope that they will take precedence over other triangles, and in the next iteration their distortion will diminish. The challenge is to iteratively update the weights to guarantee the the algorithm minimizes the desired energy. One possible approach would be to use the update rule of Iteratively Reweighted Least Squares [Pighin and Lewis 2007; Yoshizawa et al. 2004], where the weights are updated to locally match the proxy and target energy:

$$w_f^{k+1} = \frac{\mathcal{D}(\mathbf{J}_f^k(\mathbf{x}))}{\|\mathbf{J}_f^k(\mathbf{x}) - \mathbf{R}_f^k\|_F^2}. \tag{5}$$

However, our experiments show that this update rule is not sufficient to guarantee convergence, and in fact often fails to find a descent direction .

Figure 3: *The ARAP energy (left) introduces 6K inverted triangles in the parametrization (highlighted in magenta) which result in highly distorted regions around the neck and the wing of the Gargoyle. Our algorithm avoids the problem by minimizing a flip preventing symmetric Dirichlet energy (right). This model has 99K faces and took 20 iterations and 3 seconds to optimize.*

Matrix weights. We propose a slightly different algorithm: rather than reweighting the energy to match the value of the distortion measure, we reweight it so that its gradients match. This is a simple modification, which is guaranteed to find a descent direction, as we show in the next section. However, matching gradients is not possible with just a scalar weight, due to the lack of degrees of freedom, but we can achieve it using a $d \times d$ matrix of weights $\mathbf{W}_f^k$, and by transforming (4) into

$$\min_{\mathbf{x}} \quad \sum_{f\in F} A_f \left\| \mathbf{W}_f^k (\mathbf{J}_f(\mathbf{x}) - \mathbf{R}_f^k) \right\|_F^2 . \tag{6}$$

This equation can be rewritten in matrix form and linearly solved, as detailed in Appendix A.

We wish to find $\mathbf{W}_f^k$ such that the gradient of each term in (6) is equal to the gradient of $\mathcal{D}(\mathbf{J}_f(\mathbf{x}))$, that is, $\mathbf{W}_f^k$ are the solutions to

$$\nabla_{\mathbf{J}_f} \left\| \mathbf{W}_f^k (\mathbf{J}_f - \mathbf{R}_f^k) \right\|_F^2 = \nabla_{\mathbf{J}_f} \mathcal{D}(\mathbf{J}_f), \text{ when } \mathbf{J}_f = \mathbf{J}_f^k. \tag{7}$$

For brevity, we neglect the indices and simply write (7) as $\nabla_{\mathbf{J}} \|\mathbf{W}(\mathbf{J} - \mathbf{R})\|_F^2 = \nabla_{\mathbf{J}} \mathcal{D}(\mathbf{J})$. Assuming $\mathbf{J} - \mathbf{R}$ is invertible, we can manipulate this equation and find that $\mathbf{W}$ must satisfy,

$$\mathbf{W}^\top \mathbf{W} + \mathbf{W}\mathbf{W}^\top = \nabla_{\mathbf{J}} \mathcal{D}(\mathbf{J})(\mathbf{J} - \mathbf{R})^{-1}. \tag{8}$$

If $(\mathbf{J} - \mathbf{R})^{-1}$ does not exist, we can take the limit of the right-hand side above or a pseudo-inverse instead. The choice does not have an impact on the algorithm, and we defer this discussion to the next section. Since the left-hand side is semi-positive definite, a solution to (8) exists if and only if the right hand-side is semi-positive definite. In that case, we take the solution

$$\mathbf{W} = \left(\frac{1}{2} \nabla_{\mathbf{J}} \mathcal{D}(\mathbf{J})(\mathbf{J} - \mathbf{R})^{-1} \right)^{1/2}, \tag{9}$$

where the square root is a matrix principal root. We discuss the existence of the root in Section 4.

Line search. Flip preventing energies are undefined when the mapping contains an inverted element. A crucial part of our algorithm is to make sure that flips are never introduced in any step. We achieve that by using a simple backtracking line search (see [Nocedal and Wright 2006], Section 3.1). Specifically, assume $\mathbf{x}^k$ is the current iterate, and $\mathbf{d}^k$ is the search direction such that $\mathbf{x}^k + \mathbf{d}^k$ is the solution to (6). Then our next iterate is $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha \mathbf{d}^k$, where

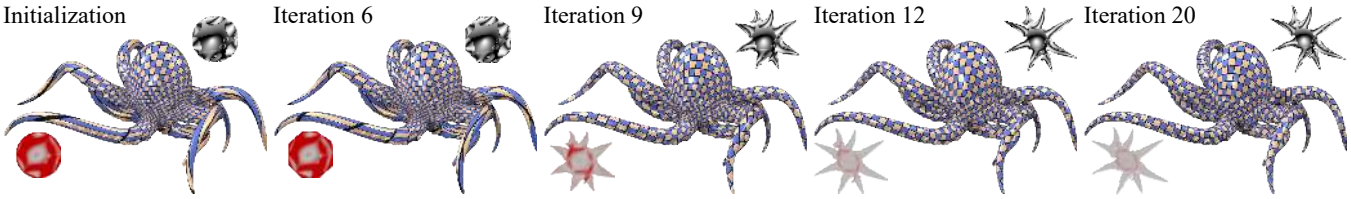


Figure 4: *Minimization of the symmetric Dirichlet energy on the Octopus model. Note how quickly the boundary of the UV map recovers from the distorted starting point. This model has 299K faces and took 20 iteration and 5.6 seconds to optimize.*

we wish to find α such that $\mathbf{x}^{k+1}$ is inversion free. Our strategy is to start with $\alpha = \min\{1, \alpha_{\max}\}$, where $\alpha_{\max}$ is the maximal step size before inversion, computed using the technique from [Smith and Schaefer 2015](Section 3.3); then we divide α by 2 until we find a solution for which the original energy (1) is lower.

The requirement in (7) essentially ensures that each step will reduce the energy. Since we require (7) to hold, the same is true for the gradients w.r.t. $\mathbf{x}$, that is,

$$\nabla_{\mathbf{x}} \left\| \mathbf{W}_f^k (\mathbf{J}_f^k(\mathbf{x}) - \mathbf{R}^k) \right\|_F^2 = \nabla_{\mathbf{x}} \mathcal{D}(\mathbf{J}_f^k(\mathbf{x})). \tag{10}$$

Hence, it also holds that for the sum over the elements,

$$\nabla_{\mathbf{x}} \sum_{f \in F} \left\| \mathbf{W}_f^k (\mathbf{J}_f(\mathbf{x}) - \mathbf{R}^k) \right\|_F^2 = \nabla_{\mathbf{x}} \sum_{f \in F} \mathcal{D}(\mathbf{J}_f^k(\mathbf{x})). \tag{11}$$

The energy in (6) is convex, and so the solution to (6) results in a descent direction (i.e. the dot product between that direction and the gradient is negative). Since the gradients match, this is also a descent direction for the original energy in (1). Therefore, our algorithm is guaranteed to reduce the energy at each iteration, just like the original ARAP method [Sorkine and Alexa 2007; Liu et al. 2008].

We observed that convergence in our case is similar to ARAP, that is, the algorithm progresses very rapidly at the beginning but then becomes much slower close to a minimum (see Fig. 7 for comparison with other methods). In Fig. 13 we show an extreme case where our method can be complemented by Newton's method to speed up convergence in the final iterations.

Reweighted local/global algorithm. Enriching the local/global algorithm with the matrix reweighting scheme and line search leads to an algorithm that is simple to implement, easily parallelizable, scales to datasets with millions of elements, supports many distortion energies and can be used to compute 2D or 3D locally injective maps. Algorithm 1 provides an overview of the method.

Algorithm 1: Reweighted local/global

Input:
A mesh M with a set of vertices V and elements F

Output:
A set of mapping coordinates $\mathbf{x} \in \mathbb{R}^{|V| \times d}$ minimizing (1)

Initialization:
$\mathbf{x}^0$ = Tutte(V, F)

Optimization:
while *has not converged* **do**
- Compute closest rotation $\mathbf{R}_f^k$ for each Jacobian $\mathbf{J}_f^k$.
- Update the weights $\mathbf{W}_f^k$ for each face using (9) or (14).
- Solve (6) to generate a descent direction $\mathbf{d}^k$.
- Find $\alpha_{\max}$, as in [Smith and Schaefer 2015](Section 3.3).
- Perform backtracking line search with step size $\alpha = \min\{1, \alpha_{\max}\}$.
- $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha\, \mathbf{d}^k$

Return:
Mapping coordinates $\mathbf{x} = \mathbf{x}^k$

4 Distortion energies

In the previous section we describe the algorithm for isometric distortions in general terms. In this section, we proceed with the treatment of specific isometric distortion measures, as well as generalizing to other types of distortions. Additionally, we fill the gap that remains from the previous section, that is, ensuring that Eq. (8) is indeed valid. We start with the latter, as it naturally leads to the derivation of the relevant formulas.

SVD viewpoint. A common property of many useful distortion measures is that they are *rotation invariant*:

Definition 4.1. *A distortion measure* $\mathcal{D}(\mathbf{J})$ *is* rotation invariant *if*

$$\mathcal{D}(\mathbf{J}) = \mathcal{D}(\mathbf{U}\mathbf{J}\mathbf{V}^\top)$$

for any rotation matrices $\mathbf{U}$ *and* $\mathbf{V}$.

From now on, we assume that $\mathcal{D}(\mathbf{J})$ is *rotation invariant*. All rotation invariant distortion measures can be written solely in terms of the singular values of $\mathbf{J}$, as shown in the following lemma.

Lemma 4.2. *Let* $\mathbf{J} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$ *be the Singular Value Decomposition of* $\mathbf{J}$. *Then,*

$$\mathcal{D}(\mathbf{J}) = \mathcal{D}(\mathbf{S}), \tag{12}$$

$$\nabla_{\mathbf{J}} \mathcal{D}(\mathbf{J}) = \mathbf{U}\, \nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S})\, \mathbf{V}^\top. \tag{13}$$

Proof. See Appedix B.1.

For example, the ARAP distortion measure, which is rotation invariant, can be written as

$$\|\mathbf{J} - \mathbf{R}\|_F^2 = \|\mathbf{S} - \mathbf{I}\|_F^2 = \sum_{i=1}^{d} (\sigma_i - 1)^2,$$

where σ_i are the singular values of $\mathbf{J}$.

Using Lemma 4.2, we can write (9) in SVD form,

$$\mathbf{W} = \mathbf{U} \left(\frac{1}{2} \nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S}) (\mathbf{S} - \mathbf{I})^{-1} \right)^{1/2} \mathbf{U}^\top = \mathbf{U}\mathbf{S}_{\mathbf{W}}\mathbf{U}^\top. \tag{14}$$

Since the matrix expression inside the square root is diagonal, it is trivial to take the root. From (14) we can clearly see that

$$\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S})(\mathbf{S} - \mathbf{I}) \succeq 0 \tag{15}$$

must hold, and in the following we show that this is always true for isometric distortion measures.

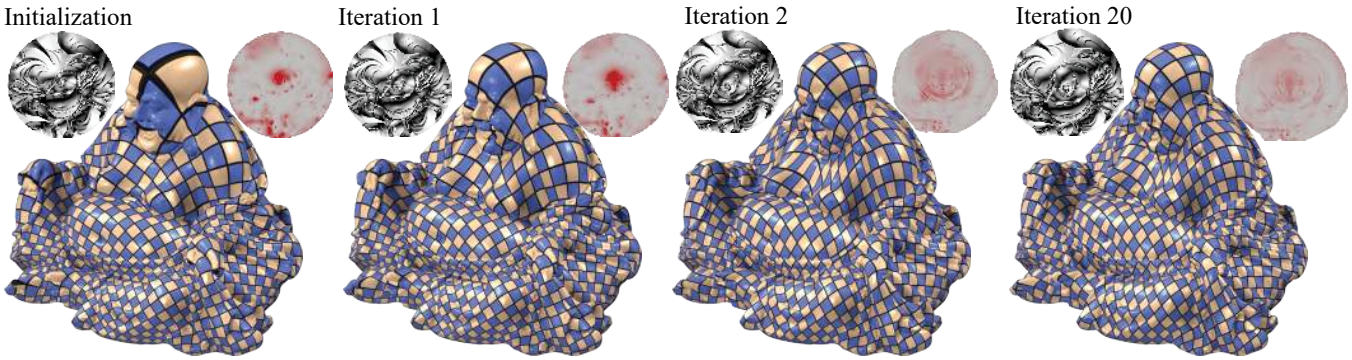


Figure 5: *Minimization of the symmetric Dirichlet energy on the Buddha model. Despite the massive size of the dataset 470K faces), our algorithm produces an optimized locally injective parametrization in 14 seconds.*

Isometric energies. We applied our algorithm to a variety of isometric distortion measures. In our context, a *true* isometric distortion measure is one that is rotationally invariant, minimal *only* for rotations, and is separable in terms of the singular values. The last condition means that

$$\mathcal{D}(\mathbf{S}) = \mathcal{D}(\sigma_1, \ldots, \sigma_d) = \sum_i f_i(\sigma_i). \tag{16}$$

This condition makes it is easy to show that Eq. (15) holds. We show it below for a specific case, and prove it in more general terms later.

In this work we experimented with the Hencky (true) strain $\left\|\log \mathbf{J}^\top \mathbf{J}\right\|_F^2$ [Paillé et al. 2013] , a symmetric version of the Dirichlet energy $\|\mathbf{J}\|_F^2 + \|\mathbf{J}^{-1}\|_F^2$ [Schreiner et al. 2004; Smith and Schaefer 2015], and an exponential symmetric Dirichlet energy $\exp(s(\|\mathbf{J}\|_F^2 + \|\mathbf{J}^{-1}\|_F^2))$, the latter inspired by [Fu et al. 2015]. All of these satisfy the condition for a true isometric distortion.

We demonstrate the solution to (14) using the symmetric Dirichlet energy. The same steps can be used for any of the above isometric distortion measures, which we report in Table 1. The Symmetric Dirichlet energy in terms of the singular values is

$$\mathcal{D}(\mathbf{J}) = \|\mathbf{J}\|_F^2 + \|\mathbf{J}^{-1}\|_F^2 = \sum_{i=1}^{d} (\sigma_i^2 + \sigma_i^{-2}).$$

Hence, $(\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S}))_i = 2(\sigma_i - \sigma_i^{-3})$, where we introduce the notation $(\mathbf{S})_i$ to refer to the i'th diagonal entry of $\mathbf{S}$. Thus,

$$(\mathbf{S_W})_i = \sqrt{\frac{\sigma_i - \sigma_i^{-3}}{\sigma_i - 1}} \tag{17}$$

when $\sigma_i \neq 1$, and $(\mathbf{S_W})_i = 4$, which is the limit, otherwise. The expression under the root is always nonnegative, so we can always find weights to match the gradients.

General distortion measures. Up to this point, we only considered the case of isometric distortion measures. Essentially, the only difference between the isometric case and the general case is in the local step in (6), that is, the choice of $\mathbf{R}_f^k$. While in the isometric case the choice is clear, namely the closest rotation, the general case is more ambiguous, since it depends on the energy. This issue has been treated for a certain range of distortion measures in [Liu et al. 2008]. In this section we describe an approach for fixing the local step for conformal and more general distortion measures.

Because we no longer take the local step to be the closest rotation, it is appropriate to give it a new letter, $\mathbf{L}$. Since we still assume that the distortion measure is rotation invariant, we should always choose $\mathbf{L}$ to have the same singular vectors as $\mathbf{J}$.

The modification of the local step changes the formula for $\mathbf{W}$ in Eq. (14), replacing the identity with the matrix $\mathbf{S_L}$, and the requirement (15) becomes

$$\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S})(\mathbf{S} - \mathbf{S_L}) \succeq 0. \tag{18}$$

The condition on the local step is now clear: The sign of each element of $\mathbf{S} - \mathbf{S_L}$ must match the sign of each element of $\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S})$. The computation in the local step can be easily adapted on a case-by-case basis.

General construction. We discuss a general construction for a wide range of distortion measures. Recall the definition of a *separably convex* function:

Definition 4.3. *A function $f(x_1, ..., x_n)$ is* separably (strictly) convex*, if for each i the single variable function $f_{(x_1,..,x_{i-1},x_{i+1},..,x_n)}(x_i)$, which is constructed by freezing all of the other variables, is* (strictly) convex.

In the case where $\mathcal{D}(\mathbf{S})$ is separably strictly convex, we propose to set $\mathbf{S_L}$ such that each entry of $\mathbf{S_L}$ minimizes the corresponding entry of $\mathcal{D}(\mathbf{S})$, assuming the other singular values are fixed. In other words, we set each $(\mathbf{S_L})_i$ to be the solution of

$$\frac{\partial}{\partial \sigma_i} \mathcal{D}_{(\sigma_1,\ldots,\sigma_{i-1},\sigma_{i+1},\ldots,\sigma_d)}(\sigma_i) = 0 \tag{19}$$

We show in Appendix B.2 that this choice always satisfies Eq. (18). The reasoning behind this choice is to always push each singular value towards its closest minimum. It is also a direct generalization

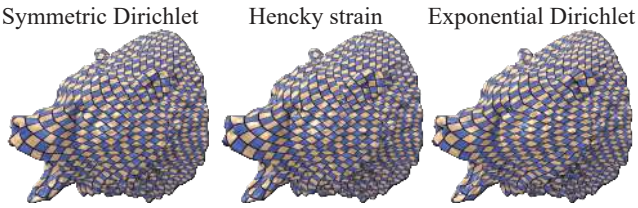


Figure 6: *Minimizing isometric distortions for the Bear model. Our approach is general and supports many distortion energies such as Symmetric Dirichlet (left), Hencky strain (middle) and Exponential Dirichlet (right). This model has 296K faces and required an average of 9 seconds to optimize.*

of the isometric case shown above, as the choice of the closest rotation exactly satisfies (19).

As an example, we derive the expressions for the AMIPS energy $E^{\star}_{iso}$ from [Fu et al. 2015] in Table 1. This energy is defined using the distortion measure

$$\mathcal{D}^{\star}_{iso}(\mathbf{J}) = \exp\left(s \cdot \mathcal{D}_{iso}(\mathbf{J})\right)$$

where

$$\mathcal{D}_{iso}(\mathbf{J}) = \frac{1}{2}\left[\left(\frac{\mathrm{tr}(\mathbf{J}^{\top}\mathbf{J})}{\det(\mathbf{J})}\right) + \frac{1}{2}(\det(\mathbf{J}) + \det(\mathbf{J}^{-1}))\right].$$

A parameterization result with this energy can be found in Fig. 2.

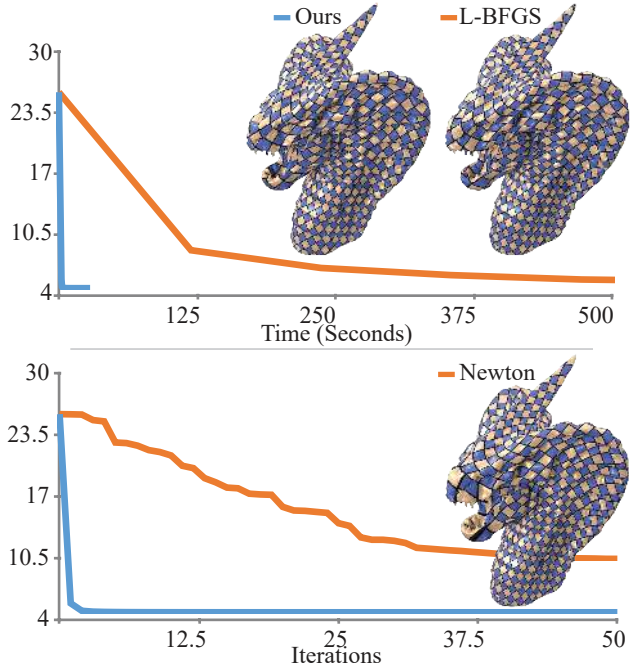


Figure 7: *Our iterations are slower than those of L-BFGS (since we have to solve a sparse linear system), but they overall progress much faster (top). Our iterations are also making considerably more progress than Newton iterations when we are far from the minimum (bottom). This model has 386K faces, and in both cases we minimize the symmetric Dirichlet energy. See Figure 13 for another comparison with Newton's method.*

Conformal distortions. The approach above is not necessarily optimal in terms of convergence speed, and in some cases may even be counterintuitive. Indeed, for conformal distortions, it may produce a local step that is not a similarity. As an example for a different approach targeted at conformal distortion, we consider a specific conformal distortion measure proposed in [Fu et al. 2015]. This measure is defined for dimension d by:

$$\mathcal{D}(\mathbf{J}) = \frac{\mathrm{tr}(\mathbf{J}^{\top}\mathbf{J})}{\det(\mathbf{J})^{2/d}}. \tag{20}$$

Again, in this case we replace the closest rotation in the local step by a similarity matrix, $L = \bar{\sigma}UV^{\top}$, where $\bar{\sigma}$ is a scalar.

For the 2D case, we show in Appendix B.3 that by setting $\bar{\sigma}$ to be any value such that $\sigma_1 > \bar{\sigma} > \sigma_2$, condition (18) is satisfied. Specifically, we choose the geometric average $\bar{\sigma} = \sqrt{\sigma_1\sigma_2}$. An example can be seen in Fig. 8. The 3D case is a bit more involved; We show in Appendix B.3 that we can choose $\bar{\sigma} = ((\sigma_1^2 + \sigma_3^3)/2)^{1/2}$. We also minimize the exponent of this conformal energy, which shares the same local step for 3D in Fig. 11. We summarize all derivations for the local step in Appendix B.3.

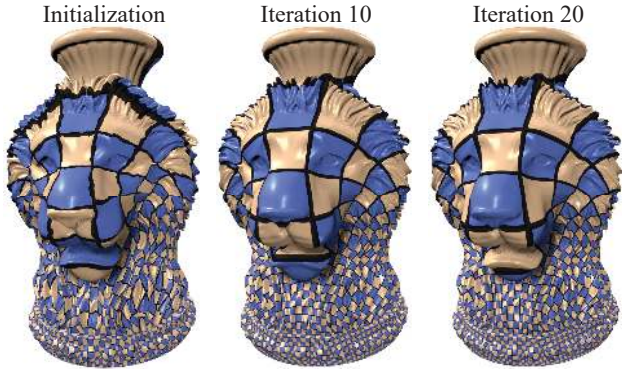


Figure 8: *Minimization of a conformal energy using our method. Note that visually the difference between 10 and 20 iterations is already quite small, and after 20 becomes negligible.*

5 Results

We ran all our experiments on a 12-core Xeon clocked at 2.7 GHz, using the PARDISO solver [Schenk et al. 2007] for the linear system solve. The sparsity pattern of the linear system in every iteration never changes, allowing us to reuse the symbolic factorization between iterations. We provide a reference implementation for parametrization using the symmetric Dirichlet energy in supplemental material (see also GitHub repository in [Rabinovich 2016]).

Our method requires a feasible, i.e., inversion free starting point: for 2D, in all examples, unless stated otherwise, we use Tutte's parametrization with cotangent weights; if they produce a flipped element, we resort to uniform weights. For the mesh improvement and deformation examples in 3D, the rest pose is used as the starting point. We report the running times in all figures. To make them comparable between parametrization and deformation applications, we exclude the time required to construct the starting point, which is required only for the former.

To avoid prescribing boundary conditions (necessary since the energies are rotation invariant) we regularize the linear system from (6) by adding a proximal term $\lambda \left\|\mathbf{x} - \mathbf{x}^k\right\|^2$ with $\lambda = 10^{-4}$.

5.1 Single-patch 2D parametrization

Single-patch 2D parametrization is ubiquitously used in modeling software to generate UV maps given a predefined set of cuts. Our approach improves over existing algorithms providing superior quality and higher efficiency, in addition to supporting extremely detailed models with millions of elements. We show the quality of our results on several meshes with different energies throughout the paper. We highlight that our method is not tied to a specific parametrization energy, and can minimize all rotation invariant energies, provided that the local step satisfies (18).

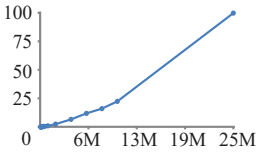
Scalability. The efficiency of our method stems from the simplicity of its implementation (it relies on solving a linear system at every iteration) and due to the experimental observation that the number of iterations is related with the geometric complexity of the model, instead of depending on the tessellation density. We show experimental evidence supporting this claim in Figure 9, where we plot the

Name	$\mathcal{D}(\mathbf{J})$	$\mathcal{D}(\sigma)$	$(\nabla_{\mathbf{S}}\mathcal{D}(\mathbf{S}))_i$	$(S_{\mathbf{L}})_i$
Symmetric Dirichlet	$\|\mathbf{J}\|_F^2 + \|\mathbf{J}^{-1}\|_F^2$	$\sum_{i=1}^n (\sigma_i^2 + \sigma_i^{-2})$	$2(\sigma_i - \sigma_i^{-3})$	1
Exponential Symmetric Dirichlet	$\exp(s \cdot (\|\mathbf{J}\|_F^2 + \|\mathbf{J}^{-1}\|_F^2))$	$\exp(s \cdot \sum_{i=1}^n (\sigma_i^2 + \sigma_i^{-2}))$	$s \cdot \exp(s \cdot 2(\sigma_i - \sigma_i^{-3}))$	1
Hencky strain	$\left\|\log \mathbf{J}^\top \mathbf{J}\right\|_F^2$	$\sum_{i=1}^n (log^2 \sigma_i)$	$2(\frac{\log \sigma_i}{\sigma_i})$	1
AMIPS	$\exp(s \cdot \frac{1}{2}(\frac{\mathrm{tr}(\mathbf{J}^\top\mathbf{J})}{\det(\mathbf{J})} + \frac{1}{2}(\det(\mathbf{J}) + \det(\mathbf{J}^{-1})))$	$\exp(s(\frac{1}{2}(\frac{\sigma_1}{\sigma_2} + \frac{\sigma_2}{\sigma_1}) + \frac{1}{4}(\sigma_1\sigma_2 + \frac{1}{\sigma_1\sigma_2}))$	$s \cdot \exp(s \cdot (\frac{1}{4}(\sigma_{i+1} - \frac{1}{\sigma_{i+1}\sigma_i^2}) + \frac{1}{2}(\frac{1}{\sigma_{i+1}} - \frac{\sigma_{i+1}}{\sigma_i^2}))$	$\sqrt{\frac{2\sigma_{i+1}^2+1}{\sigma_{i+1}^2+2}}$
Conformal AMIPS 2D	$\frac{\mathrm{tr}(\mathbf{J}^\top\mathbf{J})}{\det(\mathbf{J})}$	$\frac{\sigma_1^2+\sigma_2^2}{\sigma_1\sigma_2}$	$\frac{1}{\sigma_{i+1}} - \frac{\sigma_{i+1}}{\sigma_i^2}$	$\sqrt{\sigma_1\sigma_2}$
Conformal AMIPS 3D	$\frac{\mathrm{tr}(\mathbf{J}^\top\mathbf{J})}{\det(\mathbf{J})^{\frac{2}{3}}}$	$\frac{\sigma_1^2+\sigma_2^2+\sigma_3^2}{(\sigma_1\sigma_2\sigma_3)^{\frac{2}{3}}}$	$\frac{-2\sigma_{i+1}\sigma_{i+2}(\sigma_{i+1}^2+\sigma_{i+2}^2-2\sigma_i^2)}{(3\sigma_i\sigma_{i+1}\sigma_{i+2})^{\frac{5}{3}}}$	$\sqrt{\frac{\sigma_1^2+\sigma_3^2}{2}}$

Table 1: *Energies we used in this paper, expressed also in terms of the singular values, with their derivatives and our choice for the local step.*

energy on a progressive input mesh (Lucy) sampled between 0.6 and 6 million faces. Note that the majority of the competing methods become impractically slow for models larger than 100K vertices. An exception is [Kovalsky et al. 2015], which is targeted at a specific definition of conformal distortion. However, this method can only find the closest feasible mesh to a given initial guess (without minimizing a distortion energy) and it often fails to find a solution in our experiments.

The running time of our algorithm is dominated by the linear system solve. The PARDISO solver has good scaling properties and can parallelize the computation on multiple cores, although the complexity is still super-linear, as shown in the inset, where we plot a data point for each result shown in the paper.



"Seamless" constraints. In Figure 10, we show an example of a "seamless" parametrization computed with our algorithm. The seams of the parametrization are hidden by adding soft constraints that force the gradients of the parametrization to match on the seams, up to a fixed permutation. We experimentally observed that the constraints are nevertheless satisfied (up to numerical precision) and we leave a more detailed investigation as future work.

5.2 3D deformation

Volumetric deformation energies can be minimized with our method, benefiting in a similar way as 2D parametrization. In Figure 11, we demonstrate an example of a cube deformation with two different discretization resolutions (48K and 250K tetrahedra) and two different distortion energies (exponential Dirichlet and exponential of the conformal AMIP). The running time of our algorithm is 0.5 and 8 seconds per iteration, respectively, and we used 10 iterations.

5.3 Mesh improvement

Recently, several authors started to use mappings in order to improve the quality of meshes. We compare out algorithm for mesh improvement in 3D against the methods of [Aigerman and Lipman 2013; Fu et al. 2015]. In Figure 12, we used the exponential Dirichlet energy, which is inspired by the exponential AMIPS energy proposed in [Fu et al. 2015]. As can be seen in Table 2, our approach outperforms the competing methods in all but one case. In all cases, we performed 10 iterations, which on average took 3 seconds.

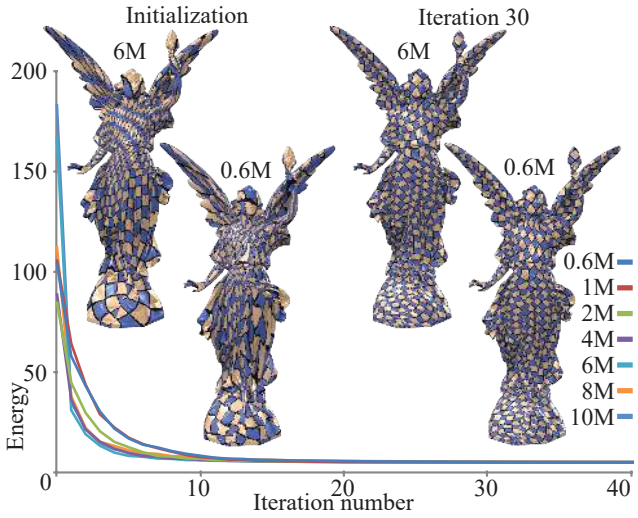


Figure 9: *We compare the behavior of our algorithm on progressively simplified versions of the Lucy model, at different mesh complexities. Note that all scales exhibit similar behavior, and although the initializations are very different (seen in the image for 0.6M and 6M triangle meshes), the end result is visually similar. Additionally, we observe that the convergence speed does not depend on the complexity of the mesh, being similar for all resolutions. For example, in the 6M version, the energy reduction is faster than both the more detailed and coarser versions. This is likely due to a less distorted initialization, which is greatly affected by the specific connectivity (and not geometry) of the mesh.*

6 Limitations and future work

We presented a general approach to quickly minimize many practical types of distortion energies. The most obvious limitation of our approach is inherited from the local/global method originally used to minimize the ARAP energy: our algorithm converges slowly near a local minimum. This problem stems from the slow propagation of the rotations in the local step, making small rotation over a large part of the parameterization hard to recover from. We exemplify this problem in a stress test similar to [Smith and Schaefer 2015] (see Figure 13). The challenge in this test is to recover from Tutte's embedding of Hilbert curve shaped into a developable surface. The embedding is bijective, but highly distorted. We tested

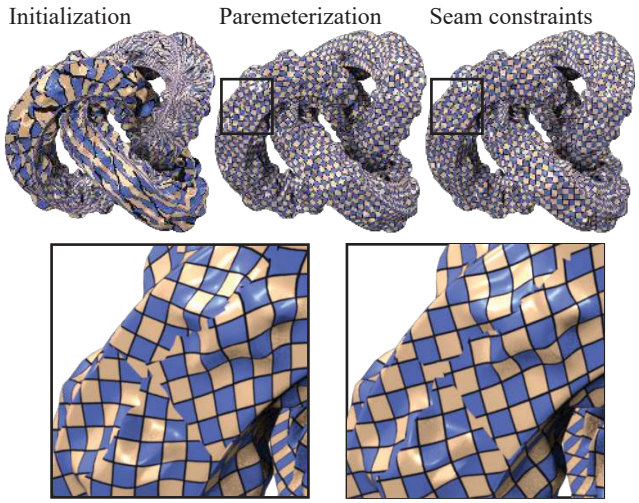


Figure 10: *An example of seamless global parameterization computed with our method. Starting from Tutte's embedding (left), we minimize the symmetric Dirichlet energy first, and then activate seamless soft constraints to make the parametrization's derivatives match on the seams, up to permutation.*

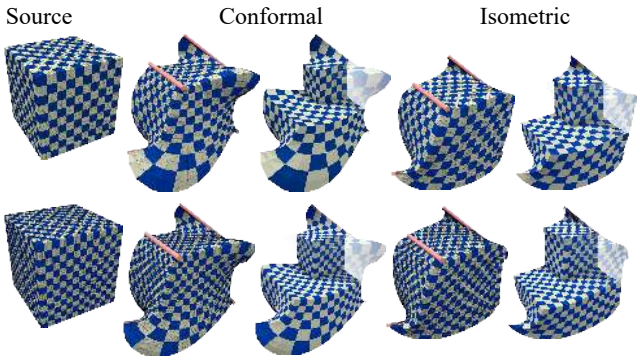


Figure 11: *Our method can be used to deform tetrahedral meshes, minimizing a conformal energy (middle) or the exponential Symmetric Dirichlet isometric energy (right). The cubes have 48K (top) and 350K (bottom) tetrahedra, and our algorithm took 5 and 80 seconds, respectively. We picked 4 edges of the cube (shown as cylinders) and manipulated them. The right image in each pair shows the interior of the deformed cube.*

both our algorithm and Newton's method on this example and found complementary behavior: both algorithms first untangle the disc initialization into a long "strip", and then proceed with curling it into the final position. Both algorithms take thousands of iterations to complete both stages. However, ours quickly handles the recovery from the distorted initialization but is slow to converge, while Newton's method exhibits the opposite behavior. By combining the two algorithms, we reconstruct the Hilbert curve in fewer than 200 iterations.

The stress test leads to another limitation, which is the requirement for an inversion free initialization. While Tutte's embedding is guaranteed to work in 2D, we are not aware of a method to compute an inversion free tetrahedral mesh in 3D (even with a fixed convex boundary), restricting our method's applicability in 3D.

A final limitation is that our algorithm only supports rotation invariant distortion energies.

Name	Init. Dihed.	BD	AMIPS	Our Method
Duck	(10,163)	(16,148)	(19.6,161.5)	(19, 138.16)
Elephant	(8,167)	(16,148)	(13.7,161.3)	(20.2,141.2)
Elephant2	(15,157)	(18,147)	(19,150)	(23.1,142.3)
Hand	(9,162)	(16,148)	(18,156)	(21.1, 143.3)
Max	(21,151)	(14,153)	(27.2,137.3)	(29,141.5)
Rocker	(10,163)	(16,148)	(21.6,148.7)	(22.8,139.4)
Skull	(0.8,178)	(14,153)	(21.1,147.6)	(17.4,157.8)
Dragon	(31,140)	(28,139)	(27.8,139.37)	(31.8, 137.6)

Table 2: *Comparison of mesh improvement achieved by running [Aigerman and Lipman 2013] (BD) and [Fu et al. 2015] (AMIPS). In each entry in the table we show the minimal and maximal dihedral angle, where the second column show the initial values. As can be seen, in all cases except for the Skull dataset, our method outperforms the competing methods.*

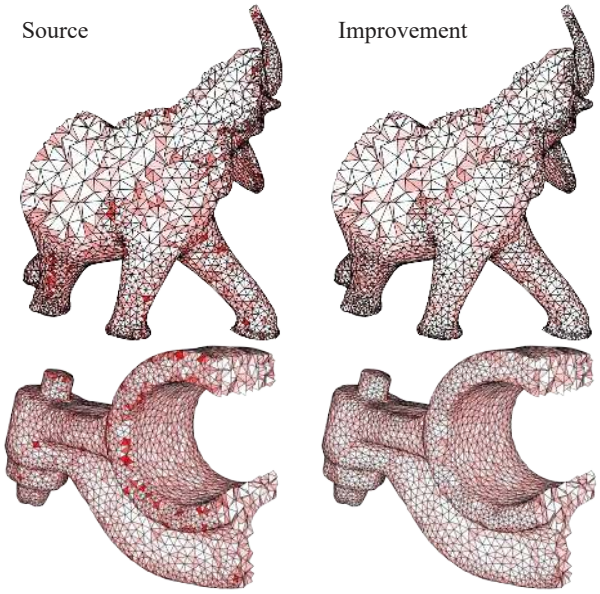


Figure 12: *An example of 3D mesh improvement computed with our method minimizing the Exponential Dirichlet energy. The elephant has 33.5K tetrahedra and the rocker arm 35.5K: our entire optimization took 3 and 3.2 seconds, respectively.*

We believe that the key idea of our algorithm is general, and other finite element problems could benefit from it, and we leave this as an exciting future work.

References

AIGERMAN, N., AND LIPMAN, Y. 2013. Injective and bounded distortion mappings in 3D. *ACM Trans. Graph. 32*, 4.

AIGERMAN, N., AND LIPMAN, Y. 2015. Orbifold Tutte embeddings. *ACM Trans. Graph. 34*, 6.

AIGERMAN, N., PORANNE, R., AND LIPMAN, Y. 2014. Lifted bijections for low distortion surface mappings. *ACM Trans. Graph. 33*, 4, 69:1–69:12.

BEN-CHEN, M., GOTSMAN, C., AND BUNIN, G. 2008. Conformal flattening by curvature prescription and metric scaling. *Computer Graphics Forum 27*, 2, 449–458.

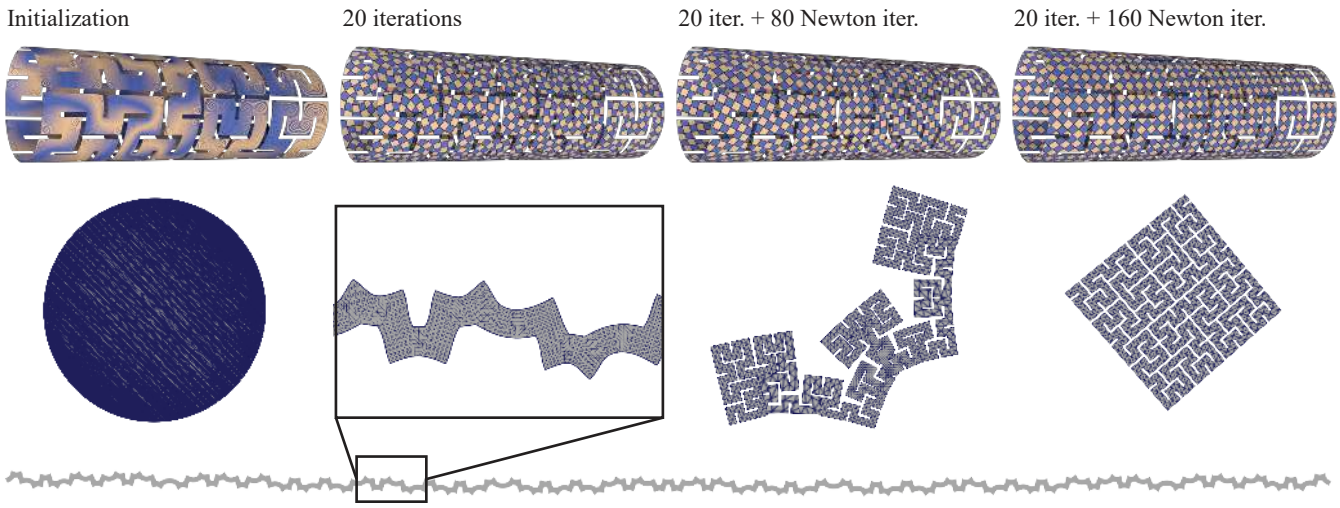


Figure 13: *Example of running our algorithm on a stress test. The highly distorted Tutte's embedding is quickly unrolled by our algorithm in about 20 iterations. As a comparison, it takes Newton's method more than a thousand iterations to reach a similar state. However, from this point the progress of our algorithm slows down considerably. In contrast, Newton's method speeds up from this point on. By combining the strength of both methods, we are able to reach the global minimum in a total of 180 iterations.*

BOMMES, D., ZIMMER, H., AND KOBBELT, L. 2009. Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3, 77:1–77:10.

BOMMES, D., LÉVY, B., PIETRONI, N., PUPPO, E., SILVA, C., TARINI, M., AND ZORIN, D. 2012. State of the art in quad meshing. In *Eurographics STARs*.

BOUAZIZ, S., DEUSS, M., SCHWARTZBURG, Y., WEISE, T., AND PAULY, M. 2012. Shape-up: Shaping discrete geometry with projections. *Comput. Graph. Forum 31*, 5, 1657–1667.

CHAO, I., PINKALL, U., SANAN, P., AND SCHRÖDER, P. 2010. A simple geometric model for elastic deformations. *ACM Trans. Graph. 29*, 4, 38:1–38:6.

CHEN, R., AND WEBER, O. 2015. Bounded distortion harmonic mappings in the plane. *ACM Trans. Graph. 34*, 4.

CIVIT-FLORES, O., AND SUSIN, A. 2014. Robust treatment of degenerate elements in interactive corotational FEM simulations. *Computer Graphics Forum 33*, 6, 298–309.

DESBRUN, M., MEYER, M., AND ALLIEZ, P. 2002. Intrinsic parameterizations of surface meshes. *Comput. Graph. Forum 21*, 3.

DIAMANTI, O., VAXMAN, A., PANOZZO, D., AND SORKINE-HORNUNG, O. 2015. Integrable PolyVector fields. *ACM Trans. Graph. 34*, 4, 38:1–38:12.

FLOATER, M. S., AND HORMANN, K. 2005. Surface parameterization: a tutorial and survey. In *Advances in Multiresolution for Geometric Modelling*, Mathematics and Visualization. 157–186.

FLOATER, M. S. 2003. One-to-one piecewise linear mappings over triangulations. *Mathematics of Computation 72*, 242, 685–696.

FU, X.-M., LIU, Y., AND GUO, B. 2015. Computing locally injective mappings by advanced MIPS. *ACM Trans. Graph. 34*, 4.

GILES, M. 2008. An extended collection of matrix derivative results for forward and reverse mode automatic differentiation.

HORMANN, K., AND GREINER, G. 2000. MIPS: An efficient global parametrization method. In *Curve and Surface Design*, Innovations in Applied Mathematics. 153–162.

IRVING, G., TERAN, J., AND FEDKIW, R. 2004. Invertible finite elements for robust simulation of large deformation. In *Proc. Eurographics Symposium on Computer Animation*, 131–140.

KOVALSKY, S. Z., AIGERMAN, N., BASRI, R., AND LIPMAN, Y. 2014. Controlling singular values with semidefinite programming. *ACM Trans. Graph. 33*, 4.

KOVALSKY, S. Z., AIGERMAN, N., BASRI, R., AND LIPMAN, Y. 2015. Large-scale bounded distortion mappings. *ACM Trans. Graph. 34*, 6, 191:1–191:10.

LABSIK, U., HORMANN, K., AND GREINER, G. 2000. Using most isometric parametrizations for remeshing polygonal surfaces. In *Proc. Geometric Modeling and Processing*, 220–228.

LEVI, Z., AND ZORIN, D. 2014. Strict minimizers for geometric optimization. *ACM Trans. Graph. 33*, 6, 185:1–185:14.

LÉVY, B., PETITJEAN, S., RAY, N., AND MAILLOT, J. 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph. 21*, 3, 362–371.

LIPMAN, Y. 2012. Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph. 31*, 4, 108:1–108:13.

LIU, L., ZHANG, L., XU, Y., GOTSMAN, C., AND GORTLER, S. J. 2008. A local/global approach to mesh parameterization. In *Proc. Symposium on Geometry Processing*, 1495–1504.

MARTIN, S., THOMASZEWSKI, B., GRINSPUN, E., AND GROSS, M. 2011. Example-based elastic materials. *ACM Trans. Graph. 30*, 4, 72:1–72:8.

MARTINEZ ESTURO, J., RÖSSL, C., AND THEISEL, H. 2014. Smoothed quadratic energies on meshes. *ACM Trans. Graph. 34*, 1, 2:1–2:12.

MULLEN, P., TONG, Y., ALLIEZ, P., AND DESBRUN, M. 2008. Spectral conformal parameterization. In *Proc. Symposium on Geometry Processing*, 1487–1494.

MYLES, A., AND ZORIN, D. 2012. Global parametrization by incremental flattening. *ACM Trans. Graph. 31*, 4, 109:1–109:11.

MYLES, A., PIETRONI, N., AND ZORIN, D. 2014. Robust field-aligned global parametrization. *ACM Trans. Graph. 33*, 4.

NOCEDAL, J., AND WRIGHT, S. J. 2006. *Numerical optimization*. Springer Series in Operations Research and Financial Engineering. Springer, Berlin.

PAILLÉ, G.-P., POULIN, P., AND LÉVY, B. 2013. Fitting polynomial volumes to surface meshes with Voronoi squared distance minimization. *Computer Graphics Forum 32*, 5, 103–112.

PIGHIN, F., AND LEWIS, J. 2007. Practical least-squares for computer graphics. In *ACM SIGGRAPH 2007 Courses*.

PORANNE, R., OVREIU, E., AND GOTSMAN, C. 2013. Interactive planarization and optimization of 3D meshes. *Comput. Graph. Forum 32*, 1, 152–163.

RABINOVICH, M., 2016. Scalable locally injective mappings. `https://github.com/MichaelRabinovich/Scalable-Locally-Injective-Mappings`.

SCHENK, O., WÖCHTER, A., AND HAGEMANN, M. 2007. Matching-based preprocessing algorithms to the solution of saddle-point problems in large-scale nonconvex interior-point optimization. *Computational Optimization and Applications 36*, 2-3, 321–341.

SCHREINER, J., ASIRVATHAM, A., PRAUN, E., AND HOPPE, H. 2004. Inter-surface mapping. *ACM Trans. Graph. 23*, 3.

SCHÜLLER, C., KAVAN, L., PANOZZO, D., AND SORKINE-HORNUNG, O. 2013. Locally injective mappings. *Computer Graphics Forum 32*, 5, 125–135.

SHEFFER, A., AND DE STURLER, E. 2001. Parameterization 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 Trans. Graph. 24*, 2, 311–330.

SHEFFER, A., PRAUN, E., AND ROSE, K. 2006. Mesh parameterization methods and their applications. *Found. Trends Comput. Graph. Vis. 2*, 2, 105–171.

SMITH, J., AND SCHAEFER, S. 2015. Bijective parameterization with free boundaries. *ACM Trans. Graph. 34*, 4, 70:1–70:9.

SORKINE, O., AND ALEXA, M. 2007. As-rigid-as-possible surface modeling. In *Proc. Symp. Geometry Processing*, 109–116.

TUTTE, W. T. 1963. How to draw a graph. *Proceedings of the London Mathematical Society 13*, 743–767.

WEBER, O., MYLES, A., AND ZORIN, D. 2012. Computing extremal quasiconformal maps. *Comput. Graph. Forum 31*, 5.

YOSHIZAWA, S., BELYAEV, A., AND SEIDEL, H.-P. 2004. A fast and simple stretch-minimizing mesh parameterization. In *Proceedings of the Shape Modeling International 2004*, IEEE Computer Society, Washington, DC, USA, SMI '04, 200–208.

ZAYER, R., LÉVY, B., AND SEIDEL, H.-P. 2007. Linear angle based parameterization. In *Proc. SGP*, 135–141.

A Solving Eq. (6)

In order to solve (6), we write it in matrix form, and in terms of the coordinates $\mathbf{x}$. Then (6) is transformed into

$$\min_{\mathbf{x}} \quad \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \tag{21}$$

where the structure of $\mathbf{A}$ and $\mathbf{b}$ in the 2D case is as follows. Assuming a set of orthogonal frames per element are prescribed, we let D_x, D_y be the FE gradient matrices of the mesh w.r.t. the frames. Additionally, we define four diagonal matrices, $\mathbf{W}_{ij}$ for $i, j = 1, 2$, where the diagonal of $\mathbf{W}_{ij}$ holds the (i, j) entries of all of the weights $\mathbf{W}_f$. In other words, $\mathbf{W}_{ij} = \mathrm{diag}(\{\mathbf{W}_f(i,j)\}_f)$. Similarly, we define $\mathbf{R}_{ij}$ to be the column vector holding the (i, j) entries of all of the $\mathbf{R}_f$

$$\mathbf{A} = \begin{pmatrix} \mathbf{W}_{11} & \mathbf{W}_{12} & 0 & 0 \\ \mathbf{W}_{21} & \mathbf{W}_{22} & 0 & 0 \\ 0 & 0 & \mathbf{W}_{11} & \mathbf{W}_{12} \\ 0 & 0 & \mathbf{W}_{21} & \mathbf{W}_{22} \end{pmatrix} \begin{pmatrix} D_x & 0 \\ 0 & D_y \\ D_x & 0 \\ 0 & D_y \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \mathbf{R}_{11} \\ \mathbf{R}_{21} \\ \mathbf{R}_{12} \\ \mathbf{R}_{22} \end{pmatrix} \tag{22}$$

This can be readily solved by any least square minimization algorithm.

B Section 3 proofs

B.1 Lemma 3.1

Lemma 3.1 *Let* $\mathbf{J} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$ *be the Singular Value Decomposition of* $\mathbf{J}$. *Then,*

$$\mathcal{D}(\mathbf{J}) = \mathcal{D}(\mathbf{S}) \tag{23}$$

$$\nabla_{\mathbf{J}}\mathcal{D}(\mathbf{J}) = \mathbf{U}\nabla_{\mathbf{S}}\mathcal{D}(\mathbf{S})\mathbf{V}^\top \tag{24}$$

Proof. (23) is immediate from definition (4.1). As for (24), we use the formula for the derivative of the singular values (see [Giles 2008])

$$\nabla_{\mathbf{J}}\mathcal{D}(\mathbf{S}) = \mathbf{U}\nabla_{\mathbf{S}}\mathcal{D}(\mathbf{S})\mathbf{V}^\top \tag{25}$$

□

B.2 Local step, general construction: proof of Eq. (19)

For a rotation invariant $\mathcal{D}(\mathbf{J})$ that is separably strictly convex in singular values, Eq. (18) can be satisfied by setting $(\mathbf{S_L})_i$ such that:

$$\frac{\partial}{\partial\sigma_i}\mathcal{D}_{(\sigma_1,\ldots,\sigma_{i-1},\sigma_{i+1},\ldots,\sigma_d)}(\sigma_i) = 0 \tag{26}$$

In particular, in the case of a true isometric distortion measures, (18) is satisfied by setting the local step as the closest rotation $\mathbf{L} = \mathbf{U}\mathbf{V}^\top$.

This can be seen from the fact that every partial function of $\mathcal{D}(\sigma)$ is strictly convex on $\mathbb{R}_{>0}$, and therefore has a single minimum, $(\mathbf{S_L})_i$. Hence, for every $\sigma_i < (\mathbf{S_L})_i$, $\frac{\partial}{\partial\sigma_i}\mathcal{D}_{(\sigma_1,\ldots,\sigma_{i-1},\sigma_{i+1},\ldots,\sigma_d)}(\sigma_i) < 0$, and for every $\sigma_i > (\mathbf{S_L})_i$, $\frac{\partial}{\partial\sigma_i}\mathcal{D}_{(\sigma_1,\ldots,\sigma_{i-1},\sigma_{i+1},\ldots,\sigma_d)}(\sigma_i) > 0$. This is also true for $(\mathbf{S} - \mathbf{S_L})_i$ and so Eq. (19) is satisfied.

B.3 Conformal Energy Local Step Derivation

Let $\mathcal{D}(\mathbf{J}) = \frac{\mathrm{tr}(\mathbf{J}^\top \mathbf{J})}{\det(\mathbf{J})^{2/d}}$. $\mathcal{D}(\mathbf{J})$ is rotation invariant and can be written as $\mathcal{D}(\sigma) = \frac{\sum_{i=1}^{d} \sigma_i^2}{\sigma_1 \dots \sigma_d}$. By differentiating the distortion measure w.r.t. the singular values in the 2D case, we find that

$$(\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S}))_1 = \frac{1}{\sigma_2} - \frac{\sigma_2}{\sigma_1^2},$$

and similarly, $(\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S}))_2 = \frac{1}{\sigma_1} - \frac{\sigma_1}{\sigma_2^2}$. Assuiming $\mathbf{J}$ is *not* a similarity already, then, since $\sigma_1 > \sigma_2 > 0$, the first entry is negative, while the second is positive. By choosing $\sigma_1 > (\mathbf{S_L})_i > (\sigma_2)$, this holds true for $(\mathbf{S} - \mathbf{S_L})_i$ and so Eq. (18) is satisfied.

For the 3D case,

$$(\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S}))_i = \frac{-2\sigma_{i+1}\sigma_{i+2}(\sigma_{i+1}^2 + \sigma_{i+2}^2 - 2\sigma_i^2)}{(3\sigma_i \sigma_{i+1} \sigma_{i+2})^{5/3}},$$

where the index i cycles from 1 to 3 (i.e., $\sigma_4 = \sigma_1$). This is zero only for $\sigma_i = \sqrt{\frac{\sigma_{i+1}^2 + \sigma_{i+2}^2}{2}}$, and so $(\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S}))_1 < 0$, $(\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S}))_3 > 0$. We note that $\bar{\sigma} = \sqrt{\frac{\sigma_1^2 + \sigma_3^2}{2}}$ satisfies $\sigma_3 < \bar{\sigma} < \sigma_1$. Therefore, by choosing $\mathbf{S_L} = \bar{\sigma}\mathbf{U}\mathbf{V}^\top$, we get $(\mathbf{S} - \mathbf{S_L})_1 < 0$, $(\mathbf{S} - \mathbf{S_L})_3 > 0$, and by construction, same as the proof for (19), we get that the sign of $(\mathbf{S} - \mathbf{S_L})_2$ is equal to the sign of $(\nabla_{\mathbf{S}} \mathcal{D}(\mathbf{S}))_2$.