Pacific Graphics 2018
H. Fu, A. Ghosh, and J. Kopf
(Guest Editors)

Volume 37 (2018), Number 7

Piecewise linear mapping optimization based on the complex view

Björn Golla Hans-Peter Seidel Renjie Chen†

Max-Planck Institute for Informatics, Saarbrücken, Germany

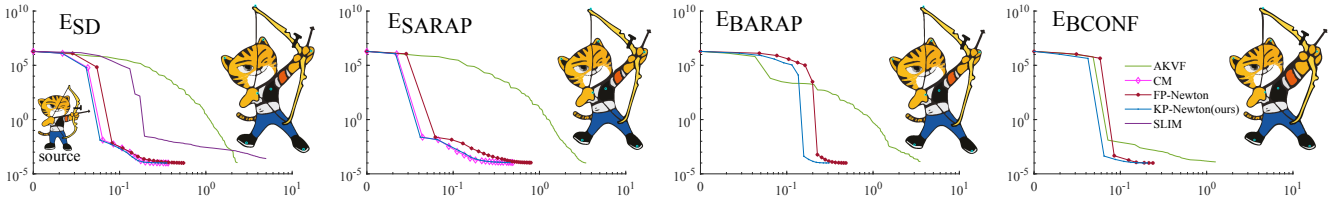


Figure 1: *Comparison of different optimization methods for shape deformation. The archery shape is deformed with user specified positional constraints (blue dots). The depicted deformation is achieved by minimizing four different energies: E_{SD}, E_{SARAP}, $E_{\mathrm{BCONF}}(k = 0.1)$ and $E_{\mathrm{BARAP}}(k = 0.1)$, all initialized using the rest pose. Notice that the results of optimizing the different energies are visually similar, however Newton's method is more efficient minimizing E_{BCONF} and E_{BARAP}.*

Abstract

We present an efficient modified Newton iteration for the optimization of nonlinear energies on triangle meshes. Noting that the linear mapping between any pair of triangles is a special case of harmonic mapping, we build upon the results of Chen and Weber [CW17]. Based on the complex view of the linear mapping, we show that the Hessian of the isometric energies has a simple and compact analytic expression. This allows us to analytically project the per-element Hessians to positive semidefinite matrices for efficient Newton iteration. We show that our method outperforms state-of-the-art methods on 2D deformation and parameterization. Further, we inspect the spectra of the per triangle energy Hessians and show that given an initial mapping, simple global scaling can shift the energy towards a more convex state. This allows Newton iteration to converge faster than starting from the given initial state. Additionally, our formulations support adding an energy smoothness term to the optimization with little additional effort, which improves the mapping results such that concentrated distortions are reduced.

CCS Concepts

• ***Theory of computation*** → ***Nonconvex optimization;*** • ***Computing methodologies*** → *Computer graphics;*

1. Introduction

For variational shape modeling, various geometric distortion measures have been proposed to capture different physical material properties in geometric processing. The most popular models are elasticity and rigidity. In general, a mapping between arbitrary domains cannot be locally rigid everywhere, i.e. preserve both the local area and angle. Therefore, optimization procedures have to be used so that the overall distortion of the sought mapping is minimized. While some geometric distortion measures, such as the LSCM [LPRM02] conformal energy, are convex and therefore easy to optimize, most distortion measures are nonlinear and nonconvex and require more sophisticated methods for optimization.

While being very effective for finding local minima, Newton's method has generally much higher per iteration cost than first order methods, as the system matrix changes in each iteration. However, recent development has shown that much of the cost can be reduced and amortized in the mapping computation tasks. For example, Rabinovich et al. [RPPSH17], Claici et al. [CBSS17] and Shtengel et al. [SPSH*17] use the fact that even though the sparse system matrix changes between iterations, it possesses the same sparsity

† Corresponding author.
This research was supported by the Max Planck Center for Visual Computing and Communication.

pattern. This allows to reuse the symbolic factorization of the direct solvers.

Recently, Shtengel et al. [SPSH*17] construct the composite majorizer of the objective energy as a convex proxy and apply Newton iteration to minimize it. We present a more straightforward approach by directly computing the Hessian of the energy function, and projecting it to a close-by positive semi-definite (PSD) matrix for Newton iteration. Our main observation is that the (linear) mapping of each triangle in a triangular mesh is a special case of harmonic mappings, where both the holomorphic and anti-holomorphic components are linear functions. This allows us to adapt the analytic Hessian projection developed for general harmonic mappings that are formulated using boundary elements by Chen and Weber [CW17]. While our modification is not optimal in the sense that it does not project to the closest positive definite matrix as in [CW17], we show through many examples that in practice the projection is close to the optimal and it leads to similar convergence behavior.

We also show that the same strategy can be used for other geometric energies. Optimizing element-wise defined distortion energies may lead to concentrated distortions. We show that under the complex view, a smoothness energy for neighboring triangles can be easily differentiated and incorporated into the element-wise distortion energy and optimized using the same pipeline.

We show that our method provides advantages over the current state-of-the-art in deformation and parameterization. Figure 1 shows an example for shape deformation, where we compare the runtime towards convergence between our method and several competing methods. We outperform SLIM and AKVF, two recent first order methods. We also compare in depth with CM, the second order method of Shtengel et al. [SPSH*17] in Section 5 and show that the two methods share significant similarities. Both are essentially projecting the Hessian to PSD matrices, yet our method is straightforward to adapt to new energies and shows faster convergence in many test cases. We also show that our analytical characterization of the PSDness of the per-element Hessians gives rise to possible analysis about the energy convexity profile. This allows our method to outperform competing second order methods in parameterization by applying global scaling to the initialization, such that the initial state of the optimization is locally more convex.

2. Related Work

Nonlinear optimization has been fundamental for geometric optimization. Hormann and Greiner [HG00] introduced the MIPS energy for parameterization and optimize it via block coordinate descent, which is later used by Fu et al. [FLG15] for locally injective parameterization.

The local/global method, first proposed for surface modeling [SA07], and later adapted for parameterization [LZX*08], can be regarded as a coordinate descent method. It alternates in solving for the mapping in a global step, and local rotations in a local step. It optimizes the As-Rigid-As-Possible (ARAP) energy, which becomes quadratic with fixed rotations. The global step has a fixed system matrix, the mesh laplacian, which results in an efficient optimization procedure, since the system matrix does not change and can be pre-factored.

While high quality deformation results can be obtained by optimizing the ARAP energy, it is finite and does not penalize inversion. This leads to unnatural results. Lipman [Lip12] proposed to explicitly bound the distortion of each individual triangle, and further developed an iterative convexification technique to solve the nonconvex optimization problem efficiently. Aigerman and Lipman [AL13] and Kovalsky et al. [KABL14] further extend the approach to higher dimensions. Kovalsky et al. [KABL15] use alternating projection to compute bound distortion maps. Hefetz et al. [HCW17] propose the alternating tangential projection method with convergences guarantees.

Injective mappings are also achievable via unconstrained optimization, by incorporating the injectivity constraint in the energy. Schüller et al. [SKPSH13] introduced a barrier term to augment arbitrary deformation energies to prevent inversions. Smith and Schaefer [SS15] proposed to use the symmetric Dirichlet energy, which has the barrier property built-in, and use L-BFGS to optimize the nonlinear energy. In the following, we review the latest work on solving such nonlinear optimization problems.

L-BFGS directly approximates the inverse of the Hessian, requiring only the position and gradient information of few previous iterations. While L-BFGS iterations are fast, they typically require many iterations to converge. Liu et al. [LBK17] improve the convergence of L-BFGS by initializing from the Laplacian. Zhu et al. [ZBK18] further improve the convergence of L-BFGS by blending the L-BFGS update with Laplacian preconditioned gradient. Combined with their barrier-aware line search filtering, they achieve fast convergence over a range of problems.

Kovalsky et al. [KGL16] proposed the Accelerated Quadratic Proxy, which uses the mesh Laplacian as a fixed Hessian replacement. They further introduced an acceleration step to speedup the convergence. Although their per-iteration cost is low, recent work [RPPSH17, SBCBG11, ZBK18] shows that it is comparatively less efficient at minimizing the objective energy.

Rabinovich et al. [RPPSH17] adapted the local/global approach [SA07, LZX*08] for general isometric energies. Their presented Scalable Locally Injective Mappings (SLIM) reweighs the ARAP energy to match the gradient of the isometric energy. Similar to the local/global approach, they observe effective iterations in the beginning, but slow convergence approaching the minimum.

Claici et al. [CBSS17] proposed an energy agnostic matrix based on approximate Killing vector fields (AKVF) motivated by the observation that the Killing vector fields correspond to rigid motions [SBCBG11]. They modify the gradient field to be As-Killing-As-Possible and show fast convergence for parameterization.

Most recently, Peng et al. [PDZ*18] reformulate several existing geometric optimization methods as fixed point problems and apply Anderson acceleration [And65] to achieve increased convergence rates.

In physical simulation, the eigenanalysis based per element Hessians projection has been used for various materials [TSIF05, SHST12, XSZB15, SGK18]. Compared to first order methods, the

per-iteration cost for second order methods is much higher in general. Thus, until very recently they have been less popular in geometric optimization. Chao et al. [CPSS10] derived the Hessian of the ARAP energy and used a blackbox Newton solver. While it shows much faster convergence than the local/global approach, the overall performance improvement seems not significant. The state-of-the-art and most closely related work to ours is that of Shtengel et al. [SPSH*17]. They construct a convex majorizer (CM) for the optimization energy and derive the Hessian analytically. Their Hessian is PSD by construction. CM can be applied to any energy which is expressed as a composite function $h(g(x))$, where h and g have convex-concave decompositions. We will show in Section 5 that the Hessian of the convex majorizer shares surprising similarity to our projected Hessian.

Liu et al. [LYNF18] achieve acceleration by progressively interpolating the reference mesh triangles used for the distortion measure. They interpolate between the identity and the current map, such that the distortion is bounded, which makes the optimization problem easier to solve.

Our work is motivated by the boundary element method of Chen and Weber [CW17] for shape deformation. They presented a purely analytic approach for the PSD Hessian modification. Their method achieves real-time performance for shape deformation, however it is explicitly bound to harmonic maps in the plane. In this work, we generalize their approach to triangle meshes. This allows us to perform parameterizations and support more general maps, more specifically, piecewise linear maps.

3. Background

Here, we give a brief overview of the concepts relevant for the presented Newton's method for the optimization of piecewise linear mappings. We refer the reader to the numerical optimization book by Nocedal and Wright [NW06] and the complex analysis book by Ahlfors [Ahl79] for further reading.

3.1. Newton's Method and Locally Injective Mappings

Newton's method is an optimization method for iteratively finding the stationary points of the objective function. In each iteration, it updates the current solution based on the gradient and Hessian. The update is guaranteed to decrease the objective if the Hessian is positive definite, which means the function is locally convex. However, this is generally not the case in practice. To address this issue, various strategies exist for projecting the Hessian to PSD matrices. Adding a scaled identity matrix to the Hessian will result in a PSD matrix for sufficiently large values. A trivial implementation of this is however far from optimal, as shown in Shtengel et al. [SPSH*17]. Computing the eigen-decomposition of the Hessian and clamping negative eigenvalues to zero gives the closest PSD matrix measured in Frobenius norm. This generally quickly becomes infeasible for large matrices. If the Hessian can be expressed as the sum of a series of matrices, modifying each matrix separately is an alternative. Although not optimal, it is generally faster to modify the individual matrices, and still maintains fast convergence in practice [TSIF05].

After an update step is produced by solving the linear system in the Newton iteration, the Newton's method is augmented with a line search procedure for an effective minimization. The initial step size $t = 1$ is iteratively reduced until the update produces a sufficient energy decrease fulfilling the Armijo condition [NW06]

$$f(Y + t\Delta) \leq f(Y) + ct\nabla f^T\Delta, \tag{1}$$

where f is the objective function, Y the current positions, Δ the update step and c a chosen parameter scaling the necessary energy decrease.

For locally injective mapping optimization based on flip preventing energies, the Newton iteration is guaranteed to produce an update direction that maintains the triangle orientations. However the line search step has to start from an orientation preserving step size t. The maximal such step can be computed based on the positive Jacobian determinant criterion [SS15]. Algorithm 1 outlines the Newton's method for computing locally injective mappings by minimizing some chosen objective function E, given a locally injective map Y of the source X as initialization.

Algorithm 1 Augmented Newton's method

```
1:  procedure NEWTON(X, Y)
2:      repeat
3:          G ← ∇E
4:          H ← ∇²E
5:          H ← H⁺                          ▷ PSD projection of H
6:          Δ ← H⁻¹G                        ▷ Compute update step
7:          t ← injectiveStep(Y, Δ)         ▷ Max step size for loc. inj.
8:          while ¬Armijo(Y, tΔ) do         ▷ Eq. 1
9:              t ← tγ                      ▷ Shorten step by γ
10:         end while
11:         Y ← Y + tΔ                      ▷ Perform update
12:     until converged
13:     return Y
14: end procedure
```

3.2. Linear Maps in the Plane: the Complex View

Our method relies on a complex reformulation of the isometric energies. Using complex numbers, we can express any planar map as a complex function $f(z,\bar{z})$, with $z = x + iy$, where x and y are the real and imaginary components of z.

As shown in [CG17], a planar linear mapping can be written as $f(z,\bar{z}) = az + b\bar{z} + c$, which can easily be separated into its holomorphic component $az + c$ and antiholomorphic component $b\bar{z}$, accordingly. Given $Z = (z_1, z_2, z_3)^T$ and $Y = (y_1, y_2, y_3)^T$, the source and target vertex positions of a triangle respectively, the complex constants a, b and c can be obtained by solving the following linear system:

$$\begin{bmatrix} z_1 & \bar{z}_1 & 1 \\ z_2 & \bar{z}_2 & 1 \\ z_3 & \bar{z}_3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = Y.$$

The solution gives the Wirtinger derivatives of f compactly as follows:

$$f_z = a = \frac{i}{4A}\overline{\begin{pmatrix} e_1 & e_2 & e_3 \end{pmatrix}}Y, \quad f_{\bar{z}} = b = -\frac{i}{4A}\begin{pmatrix} e_1 & e_2 & e_3 \end{pmatrix}Y,$$

where $e_1 = z_2 - z_3, e_2 = z_3 - z_1, e_3 = z_1 - z_2$ are the three edge vectors and A is the area of the triangle. This leads to a linear operator $D = \frac{i}{4A}\overline{\begin{pmatrix} e_1 & e_2 & e_3 \end{pmatrix}}$ with $DY = f_z$ and $\overline{D}Y = f_{\bar{z}}$.

Any rotation invariant distortion measure [RPPSH17] can be expressed as a function of the singular values of the Jacobian of the mapping, σ_1 and σ_2, which can be expressed via the Wirtinger derivatives [Lip12]:

$$\sigma_1 = |f_z| + |f_{\bar{z}}|, \quad \sigma_2 = |f_z| - |f_{\bar{z}}|$$

Note that we use the signed singular value convention, which indicates that the mapping is orientation reversing if and only if $\sigma_2 < 0$.

Other important identities include the determinant of the Jacobian $|J| = |f_z|^2 - |f_{\bar{z}}|^2$ and the Frobenius norm $\|J\|_F^2 = 2(|f_z|^2 + |f_{\bar{z}}|^2)$. These identities allow us to easily rewrite isometric energies in terms of the Wirtinger derivatives. Table 1 lists a few examples of common isometric energies.

	E	$E(x,y)$
E_{SD}	$\frac{1}{2}(\|J\|_F^2 + \|J^{-1}\|_F^2)$	$(x+y)(1+(x-y)^{-2})$
E_{exp}	$\exp(kE_{\text{SD}})$	$\exp(kE_{\text{SD}})$
E_{SARAP}	$(\sigma_1 - 1)^2 + (\sigma_2^{-1} - 1)^2$	$(\sqrt{x} + \sqrt{y} - 1)^2 + (\frac{1}{\sqrt{x} - \sqrt{y}} - 1)^2$
E_{NH}	$\frac{\mu}{2}(\|J\|_F^2 \|J\|^{-1} - 2) + \frac{\kappa}{2}(\|J\| - 1)^2$	$\mu\frac{2y}{x-y} + \frac{\kappa}{2}(x - y - 1)^2$
E_{BARAP}	$E_{\text{ARAP}} + k(\|J\| + \|J\|^{-1})$	$2(x + y - 2\sqrt{x} + 1) + k(x - y + \frac{1}{x-y})$
E_{BCONF}	$E_{\text{CONF}} + k(\|J\| + \|J\|^{-1})$	$y + k(x - y + \frac{1}{x-y})$

Table 1: *Different isometric energies E, additionally expressed in terms of the squared modulus of Wirtinger derivatives $x = |f_z|^2$ and $y = |f_{\bar{z}}|^2$.*

4. Method

The main steps of a modified Newton iteration include the computation of the gradient and Hessian of the objective function, projecting the Hessian to a positive definite matrix, and solving a linear system. As there exist efficient linear solvers, the main issues in developing a practical Newton iteration are then the efficient computation of the derivatives and projecting the Hessian to a positive definite matrix, which we will discuss in this section.

While there exist off-the-shelf solutions for computing the derivatives of nonlinear functions, e.g. finite differencing and automatic differentiation [NW06], they can be one order of magnitude slower than compact analytic expressions [SPSH*17]. For the case of piecewise linear mappings, we show it is easy to derive concise analytic expressions of the gradient and Hessian for isometric energies.

In the following, we focus the derivation on the energy of a single element, i.e. the *linear* mapping of one triangle face. For simplicity, we will use the real equivalents of the complex expressions from Section 3. In particular, the real equivalents of f_z and $f_{\bar{z}}$ are given as column 2-vectors $\boldsymbol{f_z} = \begin{pmatrix}\text{Re}(f_z) & \text{Im}(f_z)\end{pmatrix}^T$ and $\boldsymbol{f_{\bar{z}}} = \begin{pmatrix}\text{Re}(f_{\bar{z}}) & \text{Im}(f_{\bar{z}})\end{pmatrix}^T$, accordingly (note the bold symbol). While the real equivalent matrix of D is a 2×6 matrix

$$\boldsymbol{D} = \begin{pmatrix}\text{Re}(D) & -\text{Im}(D) \\ \text{Im}(D) & \text{Re}(D)\end{pmatrix}.$$

We concatenate the real and imaginary parts of the coordinates of the triangle vertices into a column 6-vector $\boldsymbol{Y} = \begin{pmatrix}\text{Re}(Y) \\ \text{Im}(Y)\end{pmatrix}$. With the energy E expressed as a function of $|f_z|^2$ and $|f_{\bar{z}}|^2$, we reuse the following definitions from [CW17]:

$$\alpha_1 = \nabla_{|f_z|^2} E, \quad \alpha_2 = \nabla_{|f_{\bar{z}}|^2} E$$
$$\beta_1 = \nabla^2_{|f_z|^2} E, \quad \beta_2 = \nabla^2_{|f_{\bar{z}}|^2} E, \quad \beta_3 = \nabla_{|f_z|^2} \nabla_{|f_{\bar{z}}|^2} E,$$

It is straightforward to derive the gradient and Hessian of E w.r.t. the free variables $\boldsymbol{Y}$. For completeness, we include the derivation in Appendix A. In short, we have the following expression for the Hessian,

$$H = \nabla^2 E = M^T K M, \tag{2}$$

with $M = \begin{bmatrix}\boldsymbol{D} \\ \overline{\boldsymbol{D}}\end{bmatrix} \in \mathbb{R}^{4\times 6}$, $\nabla^2 E \in \mathbb{R}^{6\times 6}$ and $K \in \mathbb{R}^{4\times 4}$, which is similar to the matrix under the same notation in [CW17]:

$$K = \begin{bmatrix} 2\alpha_1 I + 4\beta_1 \boldsymbol{f_z}\boldsymbol{f_z}^T & 4\beta_3 \boldsymbol{f_z}\boldsymbol{f_{\bar{z}}}^T \\ 4\beta_3 \boldsymbol{f_{\bar{z}}}\boldsymbol{f_z}^T & 2\alpha_2 I + 4\beta_2 \boldsymbol{f_{\bar{z}}}\boldsymbol{f_{\bar{z}}}^T \end{bmatrix}. \tag{3}$$

Having the analytic expression of the Hessian matrix, we are ready to apply the per-element approach of Teran et al. [TSIF05] for PSD Hessian projection. We note that our Hessian (2) has similar structure to that of the harmonic maps in [CW17], except that our left and right multiplying matrix M is not orthonormal. However, we can apply the reduced RQ factorization, $M = RQ$, with $R \in \mathbb{R}^{4\times 4}$ and $Q \in \mathbb{R}^{4\times 6}$. We only need to modify two rows of M using the Gram-Schmidt process, as the rows in both $\boldsymbol{D}$ and $\overline{\boldsymbol{D}}$ are already orthogonal with equal norm. The lower triangular matrix R is given by:

$$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ a & b & c & 0 \\ -b & a & 0 & c \end{pmatrix},$$

where:

$$a = (\|\text{Re}(D)\|^2 - \|\text{Im}(D)\|^2)\|D\|^{-2}$$
$$b = 2\langle \text{Re}(D), \text{Im}(D)\rangle \|D\|^{-2}$$
$$c = \sqrt{1 - a^2 - b^2}$$

Then the orthonormal component of M is:

$$Q = R^{-1}M = \begin{bmatrix} \boldsymbol{D} \\ \frac{1}{c}(\overline{\boldsymbol{D}} - \left[\begin{smallmatrix} a & b \\ -b & a \end{smallmatrix}\right]\boldsymbol{D}) \end{bmatrix}.$$

The 6×6 PSD projection of $H = M^T K M = Q^T (R^T K R) Q$ is therefore equivalent to the 4×4 PSD projection of $R^T K R$, since Q is orthonormal. Alas, we are unable to obtain the eigen-decomposition of $R^T K R$ analytically, therefore we resort to numerical factorization for the optimal PSD projection of H.

We further note that H is PSD if and only if K is PSD. Therefore, projecting K to PSD will lead to a PSD version of H. Furthermore, the matrix R is close to the identity matrix in general, depending on the value of c. The PSD projection of K should thus imply near optimal projection of $R^T K R$, or equivalent H, the Hessian. We can then reuse the eigenvalue analysis and the analytic PSD projection

of K from [CW17], which we show in Appendix B for completeness. For certain energies such as the symmetric Dirichlet, only α_1 and β_1 have to be modified for the projection:

$$\alpha_1' = \max(\alpha_1, 0), \quad \beta_1' = \beta_1 + \frac{1}{2}\min(\alpha_1, 0)|f_z|^{-2}. \tag{4}$$

4.1. Isometric Energies

Many isometric energies have been proposed over the last decade for geometric optimization, including the symmetric Dirichlet energy E_{SD} proposed by Schreiner et al. [SAPH04], the exponential symmetric Dirichlet energy E_{exp} [RPPSH17], the symmetric as-rigid-as-possible energy E_{SARAP} introduced by Poranne and Lipman [PL16] and the strain density for Neo-Hookean material E_{NH} [XSZB15, SPSH*17]. The expressions for mentioned energies can be found in Table 1. The modifiers for the Hessian projection for some energies are already given in [CW17], and we add the symmetric As-Rigid-As-Possible and Neo-Hookean energies (see Table 2). The process of adding support for new energies to our method simply consists of finding an expression $E(|f_z|^2, |f_{\bar{z}}|^2)$ for the energy (see Table 1) and setting the modifiers $\alpha_1 ... \beta_3$ to its first and second order derivatives. Table 2 lists the modifiers for the energies shown in Table 1.

Newton's method works most effectively for convex functions. For general nonconvex functions, Newton's method may take many iterations to converge. For more efficient isometric mapping optimization, we introduce the following energies:

$$E_{\text{BARAP}} = E_{\text{ARAP}} + k(|J| + |J|^{-1})$$
$$E_{\text{BCONF}} = E_{\text{CONF}} + k(|J| + |J|^{-1}),$$

using the barrier term $|J| + |J|^{-1}$ (it is also possible to use E_{SD} as the barrier). Since the LSCM conformal [LPRM02] energy E_{CONF} is convex and the ARAP energy E_{ARAP} is also "largely convex" (the eigenvalues of K are $\{2-2|f_z|^{-1}, 2, 2, 2\}$), the less we weigh ($k$) the barrier, the more convex E_{BARAP} and E_{BCONF} will be. For E_{BARAP}, it can be shown that if $k \leq 2$, only the eigenvalue $2\alpha_1$ can be negative. Similarly for E_{BCONF}, only $2\alpha_1$ can be negative, if $k \leq 1$.

4.2. Initialization

Given different initializations, most nonconvex optimization solvers, including Newton's method, are likely to take different paths and therefore different number of iterations to converge to possibly different local minima. Therefore it is important to initialize the Newton iteration carefully, so that the optimization finishes in less iterations and hence shorter runtime.

In shape deformation, the user usually does not change the positional constraints abruptly, therefore the result from the last user interaction provides a good initialization. In surface parameterization, we choose Tutte's embedding [Tut63, Flo97] as it is the only linear method guaranteed to produce injective mappings. However, it is often observed that the Tutte's embedding contains severely stretched triangles, especially for highly complex surfaces. We speculate that these badly shaped triangles have negative impact on Newton's method as the energies on these triangles are highly nonconvex. Take the symmetric Dirichlet energy for example, when many triangles are mapped to be close to singular, i.e. with $|J| = |f_z|^2 - |f_{\bar{z}}|^2 \approx 0$, the eigenvalue $2\alpha_1 = 1 - (|f_z|^2 + 3|f_{\bar{z}}|^2)(|f_z|^2 - |f_{\bar{z}}|^2)^{-3}$ becomes a large negative value, which implies the energy is most likely to be highly nonconvex. This could be the cause for the projected Newton to have difficulty in reducing the energy in some parameterization scenarios [RPPSH17].

Noting that simple global scaling changes this energy characteristic, we propose to move the initialization towards where the energy becomes locally convex. With the spectra of K at hand, we can achieve more convex initialization as follows. For E_{SD}, knowing only the eigenvalue $2\alpha_1$ of K can become negative, we introduce the scaling factor s as a free variable of the initialization and insert it into the expression as follows,

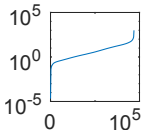
$$\begin{aligned} 2\alpha_1(s) &= 2 - 2(|sf_z|^2 + 3|sf_{\bar{z}}|^2)(|sf_z|^2 - |sf_{\bar{z}}|^2)^{-3} \\ &= 2 - 2(|f_z|^2 + 3|f_{\bar{z}}|^2)(|f_z|^2 - |f_{\bar{z}}|^2)^{-3} s^{-4}, \end{aligned}$$

from which we see that $2\alpha_1(s)$ increases as s increases. The minimal s that gives PSD K (consequently PSD Hessian H) is then,

$$s_{SD} = \sqrt[4]{(|f_z|^2 + 3|f_{\bar{z}}|^2)(|f_z|^2 - |f_{\bar{z}}|^2)^{-3}}.$$

We take the maximum of s_{SD} among all triangles as the scaling factor for the mesh, so that the energy becomes "strongly" convex. The same procedure can be repeated for E_{BCONF} and E_{BARAP}. For E_{BCONF} with $k \leq 1$, the scaling is $s_{\text{BCONF}} = (|f_z|^2 - |f_{\bar{z}}|^2)^{-\frac{1}{2}}$. For E_{BARAP} with $k \leq 2$, one has to solve the fourth order polynomial $(k+2)s^4 - 2|f_z|^{-1}s^3 - k(|f_z|^2 - |f_{\bar{z}}|^2)^{-2}$, however an alternative is simply taking s_{BCONF}, which provides a conservative estimation.

Yet, it is unnecessary for the energy to be convex on each element, in order for the global energy to be convex. It is beneficial to use a smaller scaling factor than the maximum, since the energy also tends to increase. The inset shows an example of the scaling factors from Tutte's initialization, sorted by magnitude. Apparently, only a few factors are very large. A heuristic for separating these large factors is to start form the middle of this sorted sequence and choose the first factor whose difference to the previous factor exceeds a preset threshold. In our experiments, we found 0.1 works well as a threshold.



4.3. Smoothness Energy

While the results of minimizing isometric energies look pleasing overall, the energy i.e. the distortion can be concentrated in some areas. This can especially be observed near positional constraints for deformation. Some previous work tries to alleviate this artifact by forcing smooth mapping across neighboring triangles. Levi and Gotsman [LG15] minimize a modified ARAP energy that penalizes the difference in the rotations of adjacent triangles, which results in smoother and better volume preserving deformations. Martinez et al. [MERT14] propose an energy smoothness term as a regularizer for quadratic energies, which effectively penalizes the energy difference between neighboring triangles. We introduce a similar smoothness term based on per-face isometric energies, $E_{ij} = (E_i - E_j)^2 = E_{\text{diff}}^2$, with E_i and E_j being the energy on adjacent triangles T_i and T_j. As in [MERT14], we weigh each term

	E_{SD}	E_{SARAP}	E_{NH}	E_{BARAP}	E_{BCONF}
α_1	$1-(x+3y)(x-y)^{-3}$	$\sqrt{x}^{-1}(\sigma_1-1-\sigma_2^{-3}+\sigma_2^{-2})$	$-2\mu y(x-y)^{-2}+\kappa(x-y-1)$	$k(1-(x-y)^{-2})+2(1-x^{-\frac{1}{2}})$	$k(1-(x-y)^{-2})$
α_2	$1+(3x+y)(x-y)^{-3}$	$\sqrt{y}^{-1}(\sigma_1-1+\sigma_2^{-3}-\sigma_2^{-2})$	$2\mu x(x-y)^{-2}-\kappa(x-y-1)$	$k(x-y)^{-2}+2-k$	$k(x-y)^{-2}+1-k$
β_1	$2(x+5y)(x-y)^{-4}$	$\frac{1}{2}x^{-\frac{3}{2}}(\sigma_2^{-3}-\sigma_2^{-2}-\sigma_1+1+x(3\sigma_2^{-4}-2\sigma_2^{-3}+1))$	$4\mu y(x-y)^{-3}+\kappa$	$2k(x-y)^{-3}+x^{-\frac{3}{2}}$	$2k(x-y)^{-3}$
β_2	$2(5x+y)(x-y)^{-4}$	$\frac{1}{2}y^{-\frac{3}{2}}(\sigma_2^{-2}-\sigma_2^{-3}-\sigma_1+1+y(3\sigma_2^{-4}-2\sigma_2^{-3}+1))$	$4\mu x(x-y)^{-3}+\kappa$	$2k(x-y)^{-3}$	$2k(x-y)^{-3}$
β_3	$-6(x+y)(x-y)^{-4}$	$\frac{1}{2\sqrt{xy}}(2\sigma_2^{-3}-3\sigma_2^{-4}+1)$	$-2\mu(x+y)(x-y)^{-3}-\kappa$	$-2k(x-y)^{-3}$	$-2k(x-y)^{-3}$

Table 2: *The modifiers of K for different energies. x and y denote $|f_z|^2$ and $|f_{\bar{z}}|^2$ for brevity.*

by the length of the common edge between T_i and T_j, and the influence of this energy on the results can be weighted against the per-element distortion energy.

With the complex view, the energy E_{ij} can be expanded as $E_{ij}(|f_{iz}|^2,|f_{i\bar{z}}|^2,|f_{jz}|^2,|f_{j\bar{z}}|^2)$, where f_i and f_j are the maps on triangles T_i and T_j. The derivation for the gradient and the Hessian of this per edge energy is analogous to that of the per face energy. We denote the first and second order derivatives of E_{ij} as ρ and υ. Enumerating the parameters $\{|f_{iz}|^2,|f_{i\bar{z}}|^2,|f_{jz}|^2,|f_{j\bar{z}}|^2\}$ with $\{1,2,3,4\}$, we denote the respective partial derivatives in subscript, e.g. $\rho_1=\nabla_{|f_{iz}|^2}E_{ij}$ and $\upsilon_{23}=\nabla_{|f_{i\bar{z}}|^2|f_{jz}|^2}E_{ij}$. The gradient is given by

$$2\rho_1\boldsymbol{D}_i^T f_{iz}+2\rho_2\overline{\boldsymbol{D}}_i^T f_{i\bar{z}}+2\rho_3\boldsymbol{D}_j^T f_{jz}+2\rho_4\overline{\boldsymbol{D}}_j^T f_{j\bar{z}}.$$

For the Hessian we obtain a 8×8 matrix given by:

$$\begin{bmatrix}\boldsymbol{D}_i\\ \overline{\boldsymbol{D}}_i\\ \boldsymbol{D}_j\\ \overline{\boldsymbol{D}}_j\end{bmatrix}^T\begin{bmatrix}2\rho_1 I+4\upsilon_{11}f_{iz}f_{iz}^T & 4\upsilon_{12}f_{iz}f_{i\bar{z}}^T & 4\upsilon_{13}f_{iz}f_{jz}^T & 4\upsilon_{14}f_{iz}f_{j\bar{z}}^T\\ 4\upsilon_{21}f_{i\bar{z}}f_{iz}^T & 2\rho_2 I+4\upsilon_{22}f_{i\bar{z}}f_{i\bar{z}}^T & 4\upsilon_{23}f_{i\bar{z}}f_{jz}^T & 4\upsilon_{24}f_{i\bar{z}}f_{j\bar{z}}^T\\ 4\upsilon_{31}f_{jz}f_{iz}^T & 4\upsilon_{32}f_{jz}f_{i\bar{z}}^T & 2\rho_3 I+4\upsilon_{33}f_{jz}f_{jz}^T & 4\upsilon_{34}f_{jz}f_{j\bar{z}}^T\\ 4\upsilon_{41}f_{j\bar{z}}f_{iz}^T & 4\upsilon_{42}f_{j\bar{z}}f_{i\bar{z}}^T & 4\upsilon_{43}f_{j\bar{z}}f_{jz}^T & 2\rho_4 I+4\upsilon_{44}f_{j\bar{z}}f_{j\bar{z}}^T\end{bmatrix}\begin{bmatrix}\boldsymbol{D}_i\\ \overline{\boldsymbol{D}}_i\\ \boldsymbol{D}_j\\ \overline{\boldsymbol{D}}_j\end{bmatrix}.$$

For this 8×8 matrix, we can also perform PSD projection analytically. Starting with the observation that the structure of the center matrix is analogous to that of K in Eq. 3, we obtain four of the eight eigenvalues: $2\rho_1,2\rho_2,2\rho_3,2\rho_4$. After clamping these eigenvalues to zero using the same modification as in Eq. 4, the remaining matrix can be written as the following triple product:

$$4\begin{bmatrix}f_{iz}&0&0&0\\0&f_{i\bar{z}}&0&0\\0&0&f_{jz}&0\\0&0&0&f_{j\bar{z}}\end{bmatrix}\begin{bmatrix}\upsilon'_{11}&\upsilon_{12}&\upsilon_{13}&\upsilon_{14}\\\upsilon_{12}&\upsilon'_{22}&\upsilon_{23}&\upsilon_{24}\\\upsilon_{13}&\upsilon_{23}&\upsilon'_{33}&\upsilon_{34}\\\upsilon_{14}&\upsilon_{24}&\upsilon_{34}&\upsilon'_{44}\end{bmatrix}\begin{bmatrix}f_{iz}&0&0&0\\0&f_{i\bar{z}}&0&0\\0&0&f_{jz}&0\\0&0&0&f_{j\bar{z}}\end{bmatrix}^T$$

After evaluating the off-diagonal elements of the center 4×4 matrix using the partial derivatives of the per element energies E_i and E_j, we can further decompose it into the sum :

$$2\begin{bmatrix}\upsilon'_{11}/2-\alpha_{i1}^2 & E_{\text{diff}}\beta_{i3} & 0 & 0\\ E_{\text{diff}}\beta_{i3} & \upsilon'_{22}/2-\alpha_{i2}^2 & 0 & 0\\ 0 & 0 & \upsilon'_{33}/2-\alpha_{j1}^2 & -E_{\text{diff}}\beta_{j3}\\ 0 & 0 & -E_{\text{diff}}\beta_{j3} & \upsilon'_{44}/2-\alpha_{j2}^2\end{bmatrix}$$
$$+2\begin{bmatrix}\alpha_{i1} & \alpha_{i2} & -\alpha_{j1} & -\alpha_{j2}\end{bmatrix}^T\begin{bmatrix}\alpha_{i1} & \alpha_{i2} & -\alpha_{j1} & -\alpha_{j2}\end{bmatrix}.$$

Note, that variables with subscripts including triangle indices, e.g. α_{i1}, are the derivatives of the corresponding per element energies E_i and E_j as in Section 4. The block diagonal structure allows us to derive the eigenvalues of the first matrix analytically, while the second matrix is PSD by construction. The details for the respective eigenvectors and eigenvalues are given in Appendix B. The overall modification of the smoothness energy Hessian consists of two steps. First we project the block diagonal matrix and add the outer product. Second, we perform the triple product to obtain the 8×8 matrix and add the contributions of the non-negative eigenvalues $2\rho_1,2\rho_2,2\rho_3,2\rho_4$.

5. Comparison to Composite Majorization

For piecewise linear mappings, we can also apply the complex formulation of the isometric energies to Composite Majorization [SPSH*17], which allows us to reveal the close relationship between CM and our simple PSD Hessian projection approach.

CM constructs a majorizer for energies expressed as a composite function $h(g(x))$, following the same notations from [SPSH*17]. Since all the energies presented in [SPSH*17] are expressible in terms of Wirtinger derivatives, the CM Hessian can be produced by equivalently altering the modifiers $\alpha_1...\beta_3$ in the matrix K. We derive the expressions for these energies in Appendix C. It turns out that although the two methods follow seemingly very different approaches, they share significant similarities. Table 3 shows how CM equivalently alters our modifiers in their Hessian construction. We point out that for energies constructed based on the decomposition $h(\sigma_1,\sigma_2)$ with convex h, CM produces the same Hessian as our method. The symmetric Dirichlet energy is such an example.

To apply CM to new energies, one needs to construct a convex-concave decomposition of the objective function. The choice of this decomposition is not unique and is likely to result in different PSD Hessians and consequently affects the convergence behavior. In contrast, our approach only requires the partial derivatives of a simple-to-obtain energy formulation. Additionally, our method has the property that the per face Hessians are not modified if they are already PSD, which is not necessarily the case for CM.

6. Results

We compare our algorithm to state-of-the-art methods in two applications: deformation and parameterization. The tested algorithms are the following:

- **KP-Newton**: Our method, PSD projection of K with the given eigendecomposition.
- **CM**: In our implementation we use modifiers introduced in Table 3 for the general form of K [SPSH*17].
- **AKVF**: We implemented this method following the description in [CBSS17] and the reference code.

Energy	Term modification
$E(\sigma_1,\sigma_2)$	$\alpha_1' = (\alpha_1)_+$
	$\alpha_2' = (\alpha_2)_+$
	$\beta_1' = \beta_1 + \frac{1}{2}\lvert f_z\rvert^{-2}(\alpha_1)_- - \frac{1}{4}(h^-_{uu} + h^-_{vv} + 2h^-_{uv})\lvert f_z\rvert^{-2}$
	$\beta_2' = \beta_2 + \frac{1}{2}\lvert f_{\bar z}\rvert^{-2}(\alpha_2)_- - \frac{1}{4}(h^-_{uu} + h^-_{vv} - 2h^-_{uv})\lvert f_{\bar z}\rvert^{-2}$
	$\beta_3' = \beta_3 - \frac{1}{4}(h^-_{uu} - h^-_{vv})(\lvert f_z\rvert\lvert f_{\bar z}\rvert)^{-1}$
E_{NH}	$\alpha_1' = \mu(\lvert f_z\rvert^2 - \lvert f_{\bar z}\rvert^2)^{-1} + (h_v)_+$
	$\alpha_2' = \mu(\lvert f_z\rvert^2 - \lvert f_{\bar z}\rvert^2)^{-1} - (h_v)_-$

Table 3: *We can express CM as a projection of K, by altering the modifiers in Table 2. Here we give the alterations for two types of energies. $h(u,v)$ is the energy function as in [SPSH*17] under the same notation and h^- is its concave component. $(\cdot)_+$ and $(\cdot)_-$ clamp negative and positive values to zero respectively.*

- **FP-Newton**: We use the presented orthogonalization procedure to reduce the full (6×6) projection to the 4×4 case.
- **SLIM**: We implemented SLIM based on the description in the paper [RPPSH17].

Experiment Setup: The experiments are performed in Matlab.[†] All methods share the same pipeline and we only replace the system matrix according to the method. Computationally intensive steps are delegated to OpenMP parallelized C++ code via mex. We use the Pardiso solver to benefit from its symbolic pre-factorization and numeric solve routines. The test machine has an Intel Xeon E5-1650 v3 (3.5 GHz, 6 cores) and 32GB of RAM. We use Tikhonov regularization with a scaling factor of 10^{-10}. For the line search, the step size is reduced by a factor of 0.5 in each step, and the gradient scaling for the sufficient energy decrease criterion is $c = 0.2$.

Note that for the energy plots we have subtracted the minimal energy achieved over all methods from the energy values, to gain more visible detail in the regions where the energy is small. We omit comparisons with CM for E_{BCONF} and E_{BARAP}. For the chosen parameter ($k = 0.1$), both energies are convex w.r.t. the singular values, so the computed Hessians are identical to ours.

6.1. Parameterization

The testing datasets are taken from [MPZ14] and [RPPSH17]. For closed meshes, the datasets include cuts that turn the meshes into disk-like topology. In these cases, we observe that the system matrix is often badly conditioned and causes the Cholesky factorization to fail. Experimentally, we find that this can be mitigated by applying the symmetric Jacobi preconditioning [CCSY98] before the Tikhonov regularization. In the supplemental material, we provide detailed results. Noted values are taken upon convergence or after exceeding a prescribed maximum of 1000 iterations. For parameterization the convergence criterion is $\|\nabla E\| < 10^{-3}$, in accordance to [CBSS17]. For E_{SARAP}, we terminate if $\Delta E < 10^{-8}$, since the gradient indicator for convergence fails in this case. This is because its gradient is ill-defined for close to conformal maps (including rigid maps), as the denominator $|f_{\bar z}|$ in α_2 goes towards zero.

For parameterization, we initialize with Tutte's embedding [Tut63, Flo97]. For E_{SD}, E_{BCONF} and E_{BARAP} we used the scaling factor with heuristic introduced in Section 4.2 for our method. While the initial energy may increase, the convex initialization shows advantage in the optimization. The effect is visible in Figure 2, as throughout the optimization, the energy stays largely convex.

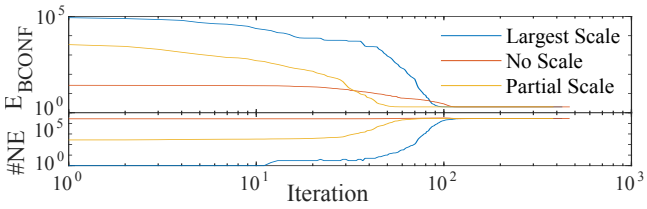


Figure 2: *Comparison of different initialization scalings for surface parameterization with E_{BCONF}. The graphs show the convergence behavior of KP-Newton on the bumpy torus mesh, and accordingly, how #NE, the number of triangles with indefinite energy Hessian, increases during the iterations.*

Since our method is essentially identical to CM [SPSH*17] for the symmetric Dirichlet energy, we can confirm many of their observations. The second order methods typically outperform the first order methods in terms of number of iterations. Since extensive comparison of CM and SLIM has already been presented in [SPSH*17], we will omit a detailed comparison here.

Except for E_{SARAP}, the energies upon convergence for all methods match. We attribute differences for E_{SARAP} to its instability close to rigid transformations. We list two metrics for the comparing methods in Table 4. We provide histograms of the performance data in the supplemental material as well as detailed results.

it/t_v	E_{SD}	E_{SARAP}	E_{BCONF}	E_{BARAP}
KP	67.3/0.188	209.9/0.661	89.1/0.249	158.9/0.447
FP	119.2/0.402	265.3/0.973	109.5/0.369	144.4/0.490
AKVF	347.7/1.037	436.9/1.460	543.9/1.651	463.9/1.385
CM	93.2/0.260	210.9/0.662		

Table 4: *Statistics for parameterization: The number of iterations (it) and the runtime (in ms) divided by the number of vertices (t_v) of the mesh, averaged over the testing datasets including nonconverged cases (terminated by iteration limit).*

Our method outperforms the other methods, except in two cases: 1) for E_{SARAP}, our method is on par with CM; 2) for E_{BARAP}, the performance of FP-Newton is comparable to ours. Although FP-Newton takes fewer iterations for E_{BARAP}, our per iteration cost is lower due to the analytical modification. For some input, such as the Hilbert curve, AKVF is a competitive first order method, as it is

[†] Source code is available at https://github.com/renjiec/meshNewton

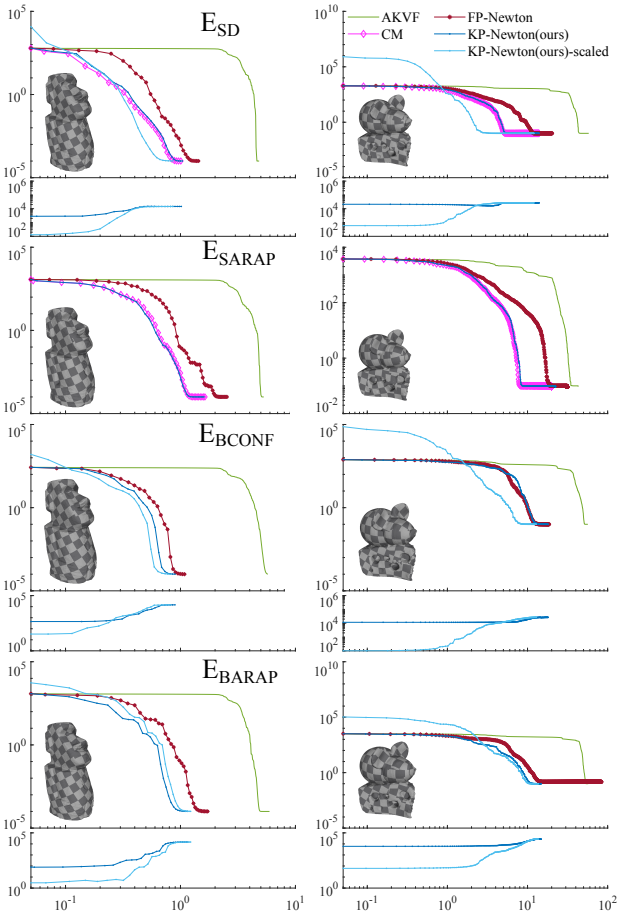


Figure 3: *Comparison of different optimization methods for surface parameterization on two meshes with different energies. For E_{SD}, E_{BCONF} and E_{BARAP}, we additionally show the number of negative eigenvalues over all local Hessians below the energy plots.*

very effective in resolving near rigid motions. In cases where this advantage does not apply, it struggles comparatively.

Figure 3 compares the convergence behavior of different methods for four energies on two meshes. We also include KP-Newton without scaling the initialization to additionally showcase the improvement by global scaling. We plotted the amount of negative eigenvalues in the local Hessians, which shows that global scaling is effective in reducing their count and consequently improved the convergence rate.

Table 5 compares the per iteration runtime of the analytic projection against the two different sized numerical projection variants, also in relation to the linear system solve runtime. Although the runtime ratio between the projection and the linear solve decreases as the problem size increases, numerical factorization still takes a substantial amount of time for medium sized meshes. On average we have observed 20% slow-down of the runtime per iteration for FP-Newton comparing to the analytic KP-Newton.

Model (#vertices)	Times(s) KP	$FP_{4\times4}$	$FP_{6\times6}$	Time(s) Solve	Ratio KP	$FP_{4\times4}$	$FP_{6\times6}$
Bimba (5691)	0.0033	0.0088	0.0122	0.0102	0.3235	0.8627	1.1961
Bunny (71141)	0.0305	0.0702	0.0945	0.1498	0.2036	0.4686	0.6308
Buddha (235771)	0.0743	0.2345	0.3139	0.6732	0.1104	0.3483	0.4663

Table 5: *Runtime comparison of different projection methods for surface parameterization. The per iteration runtime is shown on the left. The middle column shows the time for solving the linear system (including numerical factorization). The right column gives the ratio of the projection in relation to that of the linear system solve. For KP, the majority of the runtime is spent on the triple product (2) which converts K to the Hessian H.*

6.2. Deformation

The deformation shapes are taken from [CW17]. For comparison, we fix a number of positional constraints and initialize from the identity map. The handles are shown as dots in the figures. The characters are meshed with around 8300 vertices. Positional constraints are added as soft constraints. We also provide detailed comparisons for deformation in the supplemental material. For deformation, convergence is determined if an update reduces the energy by less than 10^{-6}, since the visual appearance for smaller differences is nearly indistinguishable. We include a comparison with SLIM in Figure 1, where we compare it against the other methods using E_{SD}. There, SLIM is slow in resolving large rotations, as has been noticed in previous work. AKVF at times also shows good performance in the tests, when it can maintain near rigid motion. However, it often oscillates during the iterations, which occurred in the Archer example (Figure 1). This is even more pronounced in interactive sessions. For deformation we do not divide the runtimes by the number of vertices, since all meshes have a similar vertex count.

it/t	E_{SD}	E_{SARAP}	E_{BCONF}	E_{BARAP}
KP	24.8/0.533	28.5/0.650	22.1/0.486	36.9/0.795
FP	25.0/0.775	27.3/0.830	21.9/0.631	38.8/1.150
AKVF	61.3/1.411	80.5/1.893	41.8/0.943	100.0/2.360
CM	24.8/0.536	28.5/0.633		

Table 6: *Statistics for shape deformation: The average number of iterations (it) and the average runtime(t in seconds) .*

For the tested energies, all methods except AKVF perform similarly for deformation in terms of number of iterations. However, FP-Newton takes longer due to its higher per iteration time. We show an example deformation using E_{NH} in Figure 4, where KP-Newton is faster in propagating the deformation. After only 20 iterations the visual change is very small, while CM takes 80 iterations for a similar result. In Figure 5 we show the influence of the barrier term for E_{BCONF} and E_{BARAP}. Reducing the weight for the barrier in E_{BCONF} makes the Newton iterations converge faster, although to a noticeably more conformal result.

6.3. Smoothness

We applied the energy smoothness term in shape deformation. The smoothness term visibly changes the appearance of the deformed

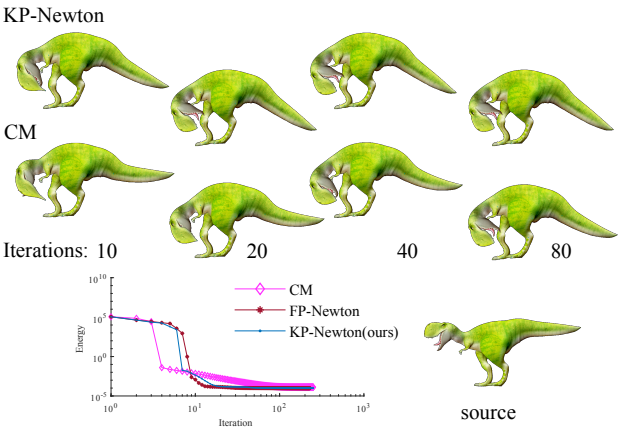


Figure 4: *Deformation of the rex shape using the Neo-Hookean energy. We show snapshots of the deformed shapes at four instances. We choose* $\frac{\mu}{\kappa} = 0.25$.

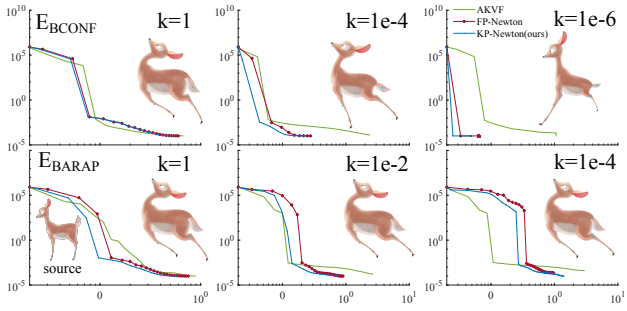


Figure 5: *Convergence comparison with different weights of the barrier term for* E_{BCONF} *and* E_{BARAP}. *The mapping becomes visually more conformal as the weight decreases for* E_{BCONF}.

shape. Figure 6 compares the deformation with and without the term. The weight for the smoothness is 0.25. It is apparent that the energy penalizes the strong distortions near handles, which are visible in the energy plot and produce artifacts in the texture map. Compared to our method without the smoothness term, the number of nonzero entries in the system matrix of Newton iteration almost doubled, and the runtime of each iteration increases by a factor of around 3.5 on average. For the raptor shape meshed with around 8,300 vertices, the deformation runs at interactive rates.

Table 7 shows the runtime of the Hessian projection for the smoothness energy. For small size problems, the numerical projection costs more than solving the linear system. The analytical projection is faster and takes less than half the time of the linear solve. The overall per iteration runtime increased 46% using numerical projection, compared to the analytic projection.

Small (6268)			Large (32540)		
Solve	Analytic	8×8	Solve	Analytic	8×8
0.028	0.012	0.040	0.113	0.043	0.113

Table 7: *Comparison of per iteration runtimes (seconds) of solving the linear system (including numerical factorization) and Hessian projection steps for the smoothness energy. The experiments were performed for two mesh resolutions.*

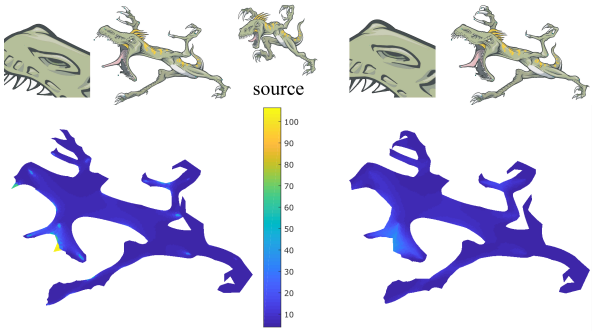


Figure 6: *The deformation results without (left) and with (right) smoothness term. Concentrated distortions near handles are visible as highlights in the energy plot (*E_{SD}*). The distortion is distributed more evenly when applying the smoothness term. The distortion leads to smeared textures at the nose.*

7. Discussion & Conclusion

We have applied the complex view to the piecewise linear mapping, and shown that simple analytic expressions of the Hessian are obtained, which allows simple and close to optimal analytic PSD projection. We have also sped-up the numerical projection for FP-Newton by reducing the matrix size. Still, we observe that using numerical projection was 20% slower than the analytic approach per Newton iteration. Our results show that the Newton iteration is an effective optimization approach for up to medium scale mesh problems. For parameterization, we additionally introduced the global scaling strategy, which is shown often effective in improving the convergence rate in comparison to CM and FP-Newton. This is possible, since we have analytic expressions directly related to the PSDness of the per-element Hessians.

As demonstrated, our PSD projection of the Hessian can be easily adapted to new isometric energies in a straightforward way. For E_{BCONF}, our assumption that increased weight for the convex energy term improves the convergence was confirmed in deformation test cases. The complex formulations also allow us to effortlessly add a smoothness term, which improves the resulting mapping near large distortions.

Our method is limited to planar mappings by construction. Deformations in 3D are thus out of reach of our method. Like [RPPSH17, CBSS17, SPSH*17], the scalability of our method is limited by the size of the sparse linear system. In spite of the low number of iterations, the setup and solve of the linear systems becomes prohibitive for large meshes. Alternative optimiza-

tion schemes such as a matrix-free truncated Newton method are imaginable as a direction for future work.

Appendix A: Gradient and Hessian of Isometric Energies

With the multivariate chain rule, we arrive at the gradient of E,

$$\nabla_{\boldsymbol{Y}} E = \alpha_1 \nabla_{\boldsymbol{Y}} \|\boldsymbol{D}\boldsymbol{Y}\|^2 + \alpha_2 \nabla_{\boldsymbol{Y}} \|\overline{\boldsymbol{D}}\boldsymbol{Y}\|^2 = 2(\alpha_1 \boldsymbol{D}^T\boldsymbol{D} + \alpha_2 \overline{\boldsymbol{D}}^T\overline{\boldsymbol{D}})\boldsymbol{Y}.$$

To derive the Hessian, we first apply the product rule:

$$\begin{aligned}\nabla^2_{\boldsymbol{Y}} E = &2\big(\alpha_1 \nabla_{\boldsymbol{Y}}(\boldsymbol{Y}^T\boldsymbol{D}^T\boldsymbol{D}) + \nabla_{\boldsymbol{Y}}\alpha_1 (\boldsymbol{Y}^T\boldsymbol{D}^T\boldsymbol{D})\big)\\ &+2\big(\alpha_2 \nabla_{\boldsymbol{Y}}(\boldsymbol{Y}^T\overline{\boldsymbol{D}}^T\overline{\boldsymbol{D}}) + \nabla_{\boldsymbol{Y}}\alpha_2 (\boldsymbol{Y}^T\overline{\boldsymbol{D}}^T\overline{\boldsymbol{D}})\big),\end{aligned}$$

then apply the chain rule to α_1 and α_2, and with further simplification we get:

$$\begin{aligned}&= \boldsymbol{D}^T(2\alpha_1 I + 4\beta_1 \boldsymbol{f_z}\boldsymbol{f_z}^T)\boldsymbol{D} + \overline{\boldsymbol{D}}^T(4\beta_3 \boldsymbol{f_{\bar z}}\boldsymbol{f_z}^T)\boldsymbol{D}\\ &+\boldsymbol{D}^T(4\beta_3 \boldsymbol{f_z}\boldsymbol{f_{\bar z}}^T)\overline{\boldsymbol{D}} + \overline{\boldsymbol{D}}^T(2\alpha_2 I + 4\beta_2 \boldsymbol{f_{\bar z}}\boldsymbol{f_{\bar z}}^T)\overline{\boldsymbol{D}},\end{aligned}$$

which can be written in the following shape,

$$= \begin{bmatrix}\boldsymbol{D}\\ \overline{\boldsymbol{D}}\end{bmatrix}^T \begin{bmatrix}2\alpha_1 I + 4\beta_1 \boldsymbol{f_z}\boldsymbol{f_z}^T & 4\beta_3 \boldsymbol{f_z}\boldsymbol{f_{\bar z}}^T\\ 4\beta_3 \boldsymbol{f_{\bar z}}\boldsymbol{f_z}^T & 2\alpha_2 I + 4\beta_2 \boldsymbol{f_{\bar z}}\boldsymbol{f_{\bar z}}^T\end{bmatrix}\begin{bmatrix}\boldsymbol{D}\\ \overline{\boldsymbol{D}}\end{bmatrix}.$$

Appendix B: Eigen Value Decompositions of Energy Hessians

First, we restate the eigenvalues and eigenvectors of K in Eq. 3 given by [CW17] :

$$\begin{bmatrix}\mathrm{Im}(f_z) & 0 & \mathrm{Re}(f_z) & \mathrm{Re}(f_z)\\ -\mathrm{Re}(f_z) & 0 & \mathrm{Im}(f_z) & \mathrm{Im}(f_z)\\ 0 & \mathrm{Im}(f_{\bar z}) & t_1\mathrm{Re}(f_{\bar z}) & t_2\mathrm{Re}(f_{\bar z})\\ 0 & -\mathrm{Re}(f_{\bar z}) & -t_1\mathrm{Im}(f_{\bar z}) & -t_2\mathrm{Im}(f_{\bar z})\end{bmatrix},$$

where the columns are the eigenvectors corresponding to the eigenvalues

$$\begin{matrix}2\alpha_1\\ 2\alpha_2\\ s_1+\sqrt{s_2^2+16\beta_3^2|f_z|^2|f_{\bar z}|^2}\\ s_1-\sqrt{s_2^2+16\beta_3^2|f_z|^2|f_{\bar z}|^2}\end{matrix}, \text{ with } \begin{matrix}s_{1,2} = \alpha_1 + 2\beta_1|f_z|^2 \pm (\alpha_2 + 2\beta_2|f_{\bar z}|^2)\\ t_{1,2} = \dfrac{\lambda_{3,4} - 2\alpha_1 - 4\beta_1|f_z|^2}{4\beta_3|f_{\bar z}|^2}\end{matrix}$$

Second, we continue with the eigen-decompositions needed for the smoothness energy Hessian projection. The eigenvectors for the four observable eigenvalues of the center 8×8 matrix in Section 4.3 are analogous to the previous and given by the columns of:

$$\begin{bmatrix}f_{iz}^{\perp} & 0 & 0 & 0\\ 0 & f_{i\bar z}^{\perp} & 0 & 0\\ 0 & 0 & f_{jz}^{\perp} & 0\\ 0 & 0 & 0 & f_{j\bar z}^{\perp}\end{bmatrix},$$

where $f^{\perp} = \begin{bmatrix}\mathrm{Im}(f) & -\mathrm{Re}(f)\end{bmatrix}^T$.

Computing the eigenvectors and eigenvalues of the 4×4 block diagonal matrix is then reduced to finding those of the 2×2 submatrices. The four eigenvalues are:

$$\frac{1}{2}\begin{pmatrix}p_1-\sqrt{q_1}\\ p_1+\sqrt{q_1}\\ p_2-\sqrt{q_2}\\ p_2+\sqrt{q_2}\end{pmatrix}, \text{ with } \begin{matrix}p_1 = \upsilon'_{11}/2 - \alpha_{i1}^2 + \upsilon'_{22}/2 - \alpha_{i2}^2\\ q_1 = (\upsilon'_{11}/2 - \alpha_{i1}^2 - \upsilon'_{22}/2 + \alpha_{i2}^2)^2 + 4E_{\mathrm{diff}}^2\beta_{i3}^2\\ p_2 = \upsilon'_{33}/2 - \alpha_{j1}^2 + \upsilon'_{44}/2 - \alpha_{j2}^2\\ q_2 = (\upsilon'_{33}/2 - \alpha_{j1}^2 - \upsilon'_{44}/2 + \alpha_{j2}^2)^2 + 4E_{\mathrm{diff}}^2\beta_{j3}^2\end{matrix}.$$

And the corresponding eigenvectors are

$$\begin{bmatrix}p_3+\sqrt{q_1} & p_3-\sqrt{q_1} & 0 & 0\\ -2E_{\mathrm{diff}}\beta_{i3} & -2E_{\mathrm{diff}}\beta_{i3} & 0 & 0\\ 0 & 0 & p_4+\sqrt{q_2} & p_4-\sqrt{q_2}\\ 0 & 0 & 2E_{\mathrm{diff}}\beta_{j3} & 2E_{\mathrm{diff}}\beta_{j3}\end{bmatrix},$$

with $p_3 = \upsilon'_{22}/2 - \upsilon'_{11}/2 + \alpha_{i1}^2 - \alpha_{i2}^2$ and $p_4 = \upsilon'_{44}/2 - \upsilon'_{33}/2 + \alpha_{j1}^2 - \alpha_{j2}^2$. For the special case that $E_{\mathrm{diff}}\beta_{i3}$ or $E_{\mathrm{diff}}\beta_{j3}$ is zero, the corresponding block (top left or bottom right) becomes the identity.

Appendix C: Derivation of the Composite Majorizer Modifiers

With the definition of the composite majorizer [SPSH*17] concretized for a two parameter function $h(u,v)$, we get:

$$\begin{aligned}H = \frac{\partial g}{\partial x}^T \nabla^2 h^+ \frac{\partial g}{\partial x} &+ \left(\frac{\partial h}{\partial u}\right)_+ \nabla^2 u^+ + \left(\frac{\partial h}{\partial v}\right)_+ \nabla^2 v^+\\ &+ \left(\frac{\partial h}{\partial u}\right)_- \nabla^2 u^- + \left(\frac{\partial h}{\partial v}\right)_- \nabla^2 v^-\end{aligned} \tag{5}$$

Here, $(\cdot)_+$ and $(\cdot)_-$ mean the values are clamped if they are smaller and larger than 0 respectively. $(\cdot)^+$ and $(\cdot)^-$ are the convex and concave component of a function. For $g(x) = (\sigma_1, \sigma_2)$, we have:

$$\frac{\partial g}{\partial x} = \frac{\partial}{\partial x}\begin{bmatrix}\sigma_1\\ \sigma_2\end{bmatrix} = \begin{bmatrix}|f_z|^{-1}\boldsymbol{f_z}^T\boldsymbol{D} + |f_{\bar z}|^{-1}\boldsymbol{f_{\bar z}}^T\overline{\boldsymbol{D}}\\ |f_z|^{-1}\boldsymbol{f_z}^T\boldsymbol{D} - |f_{\bar z}|^{-1}\boldsymbol{f_{\bar z}}^T\overline{\boldsymbol{D}}\end{bmatrix}.$$

Substitute this into Eq. 5, simplify and apply term gathering, we can then write the CM Hessian in the shape of Eq. 2 with $K = \begin{bmatrix}K_{11} & K_{12}\\ K_{12}^T & K_{22}\end{bmatrix}$, where

$$\begin{aligned}K_{11} =& (h_u + h_v)_+ |f_z|^{-3}(|f_z|^2 I - \boldsymbol{f_z}\boldsymbol{f_z}^T) + (h_{uu}^+ + h_{vv}^+ + 2h_{uv}^+)|f_z|^{-2}\boldsymbol{f_z}\boldsymbol{f_z}^T\\ K_{12} =& (h_{uu}^+ - h_{vv}^+)(|f_z||f_{\bar z}|)^{-1}\boldsymbol{f_z}\boldsymbol{f_{\bar z}}^T\\ K_{22} =& (h_u - h_v)_+ |f_{\bar z}|^{-3}(|f_{\bar z}|^2 I - \boldsymbol{f_{\bar z}}\boldsymbol{f_{\bar z}}^T) + (h_{uu}^+ + h_{vv}^+ - 2h_{uv}^+)|f_{\bar z}|^{-2}\boldsymbol{f_{\bar z}}\boldsymbol{f_{\bar z}}^T.\end{aligned}$$

On the other hand, for the composite function $h(\sigma_1,\sigma_2)$, we have $E(x,y) = h(\sqrt{x}+\sqrt{y}, \sqrt{x}-\sqrt{y})$, with $x = |f_z|^2$ and $y = |f_{\bar z}|^2$, which allows us to express the derivatives of E in terms of those of h:

$$\begin{matrix}\alpha_1 = \frac{h_u + h_v}{2|f_z|}\\ \alpha_2 = \frac{h_u - h_v}{2|f_{\bar z}|}\end{matrix} \quad \begin{matrix}\beta_1 = \frac{1}{4}|f_z|^{-3}\big((h_{uu} + h_{vv} + 2h_{uv})|f_z| - (h_u + h_v)\big)\\ \beta_2 = \frac{1}{4}|f_{\bar z}|^{-3}\big((h_{uu} + h_{vv} - 2h_{uv})|f_{\bar z}| - (h_u - h_v)\big)\\ \beta_3 = \frac{1}{4}(|f_z||f_{\bar z}|)^{-1}(h_{uu} - h_{vv})\end{matrix}.$$

We can then express the CM Hessian equivalently as alterations to our modifiers, as shown in Table 3.

For the Neo-Hookean energy, we repeat the same procedure. In this case, we have:

$$\begin{aligned}K_{11} =& \quad (16\mu|f_{\bar z}|^2|J|^{-3} + \kappa)\boldsymbol{f_z}\boldsymbol{f_z}^T + 2\mu|J|^{-1}I + 2\big(\kappa(|J|-1) - \mu m\big)_+ I\\ K_{12} =& -4\big(2\mu|J|^{-1}m + \kappa\big)\boldsymbol{f_z}\boldsymbol{f_{\bar z}}^T\\ K_{22} =& \quad (16\mu|f_z|^2|J|^{-3} + \kappa)\boldsymbol{f_{\bar z}}\boldsymbol{f_{\bar z}}^T + 2\mu|J|^{-1}I - 2\big(\kappa(|J|-1) - \mu m\big)_- I,\end{aligned}$$

where $m = |J|^{-2}(|f_z|^2 + |f_{\bar z}|^2)$.

References

[Ahl79] AHLFORS L.: *Complex analysis: an introduction to the theory of analytic functions of one complex variable*. McGraw-Hill, 1979. 3

[AL13] AIGERMAN N., LIPMAN Y.: Injective and bounded distortion mappings in 3d. *ACM TOG 32*, 4 (2013), 106:1–106:14. 2

[And65] ANDERSON D. G.: Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM) 12*, 4 (1965), 547–560. 2

[CBSS17] CLAICI S., BESSMELTSEV M., SCHAEFER S., SOLOMON J.: Isometry-aware preconditioning for mesh parameterization. In *Computer Graphics Forum* (2017), vol. 36, pp. 37–47. 1, 2, 6, 7, 9

[CCSY98] CHAN T. F., CHOW E., SAAD Y., YEUNG M.-C.: Preserving symmetry in preconditioned krylov subspace methods. *SIAM Journal on Scientific Computing 20*, 2 (1998), 568–581. 7

[CG17] CHEN R., GOTSMAN C.: Approximating planar conformal maps using regular polygonal meshes. In *Computer Graphics Forum* (2017), vol. 36, pp. 629–642. 3

[CPSS10] CHAO I., PINKALL U., SANAN P., SCHRÖDER P.: A simple geometric model for elastic deformations. *ACM TOG 29*, 4 (2010), 38:1–38:6. 3

[CW17] CHEN R., WEBER O.: GPU-accelerated locally injective shape deformation. *ACM TOG 36*, 6 (2017), 214:1–214:13. 1, 2, 3, 4, 5, 8, 10

[FLG15] FU X.-M., LIU Y., GUO B.: Computing locally injective mappings by advanced mips. *ACM TOG 34*, 4 (2015), 71:1–71:12. 2

[Flo97] FLOATER M. S.: Parametrization and smooth approximation of surface triangulations. *Computer aided geometric design 14*, 3 (1997), 231–250. 5, 7

[HCW17] HEFETZ E. F., CHIEN E., WEBER O.: Fast planar harmonic deformations with alternating tangential projections. In *Computer Graphics Forum* (2017), vol. 36, pp. 175–188. 2

[HG00] HORMANN K., GREINER G.: *MIPS: An efficient global parametrization method*. Tech. rep., DTIC Document, 2000. 2

[KABL14] KOVALSKY S. Z., AIGERMAN N., BASRI R., LIPMAN Y.: Controlling singular values with semidefinite programming. *ACM TOG 33*, 4 (2014), 68:1–68:13. 2

[KABL15] KOVALSKY S. Z., AIGERMAN N., BASRI R., LIPMAN Y.: Large-scale bounded distortion mappings. *ACM TOG 34*, 6 (2015), 191:1–191:10. 2

[KGL16] KOVALSKY S. Z., GALUN M., LIPMAN Y.: Accelerated quadratic proxy for geometric optimization. *ACM TOG 35*, 4 (2016), 134:1–134:11. 2

[LBK17] LIU T., BOUAZIZ S., KAVAN L.: Quasi-Newton methods for real-time simulation of hyperelastic materials. *ACM TOG 36*, 3 (2017), 116a. 2

[LG15] LEVI Z., GOTSMAN C.: Smooth rotation enhanced as-rigid-as-possible mesh animation. *IEEE transactions on visualization and computer graphics 21*, 2 (2015), 264–277. 5

[Lip12] LIPMAN Y.: Bounded distortion mapping spaces for triangular meshes. *ACM TOG 31*, 4 (2012), 108:1–108:13. 2, 4

[LPRM02] LÉVY B., PETITJEAN S., RAY N., MAILLOT J.: Least squares conformal maps for automatic texture atlas generation. *ACM TOG 21*, 3 (2002), 369–371. 1, 5

[LYNF18] LIU L., YE C., NI R., FU X.-M.: Progressive parameterizations. *ACM TOG 37*, 4 (2018), 41:1–41:12. 3

[LZX*08] LIU L., ZHANG L., XU Y., GOTSMAN C., GORTLER S. J.: A local/global approach to mesh parameterization. *Computer Graphics Forum 27*, 5 (2008), 1495–1504. 2

[MERT14] MARTINEZ ESTURO J., RÖSSL C., THEISEL H.: Smoothed quadratic energies on meshes. *ACM TOG 34*, 1 (2014), 2:1–2:12. 5

[MPZ14] MYLES A., PIETRONI N., ZORIN D.: Robust field-aligned global parametrization. *ACM TOG 33*, 4 (2014), 135:1–135:14. 7

[NW06] NOCEDAL J., WRIGHT S. J.: *Numerical Optimization*. Springer Science & Business Media, 2006. 3, 4

[PDZ*18] PENG Y., DENG B., ZHANG J., GENG F., QIN W., LIU L.: Anderson acceleration for geometry optimization and physics simulation. *ACM TOG 37*, 4 (2018), 42:1–42:14. 2

[PL16] PORANNE R., LIPMAN Y.: *Simple approximations of planar deformation operators*. Tech. rep., ETH Zurich, 2016. 5

[RPPSH17] RABINOVICH M., PORANNE R., PANOZZO D., SORKINE-HORNUNG O.: Scalable locally injective mappings. *ACM TOG 36*, 2 (2017), 37a. 1, 2, 4, 5, 7, 9

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Eurographics Symposium on Geometry processing* (2007), pp. 109–116. 2

[SAPH04] SCHREINER J., ASIRVATHAM A., PRAUN E., HOPPE H.: Inter-surface mapping. *ACM TOG 23*, 3 (2004), 870–877. 5

[SBCBG11] SOLOMON J., BEN-CHEN M., BUTSCHER A., GUIBAS L.: As-killing-as-possible vector fields for planar deformation. In *Computer Graphics Forum* (2011), vol. 30, pp. 1543–1552. 2

[SGK18] SMITH B., GOES F. D., KIM T.: Stable neo-hookean flesh simulation. *ACM TOG 37*, 2 (2018), 12:1–12:15. 2

[SHST12] STOMAKHIN A., HOWES R., SCHROEDER C., TERAN J. M.: Energetically consistent invertible elasticity. In *Proceedings of the 11th ACM SIGGRAPH/Eurographics conference on Computer Animation* (2012), pp. 25–32. 2

[SKPSH13] SCHÜLLER C., KAVAN L., PANOZZO D., SORKINE-HORNUNG O.: Locally injective mappings. *Computer Graphics Forum 32*, 5 (2013), 125–135. 2

[SPSH*17] SHTENGEL A., PORANNE R., SORKINE-HORNUNG O., KOVALSKY S. Z., LIPMAN Y.: Geometric optimization via composite majorization. *ACM TOG 36*, 4 (2017), 38:1–38:11. 1, 2, 3, 4, 5, 6, 7, 9, 10

[SS15] SMITH J., SCHAEFER S.: Bijective parameterization with free boundaries. *ACM TOG 34*, 4 (2015), 70:1–70:9. 2, 3

[TSIF05] TERAN J., SIFAKIS E., IRVING G., FEDKIW R.: Robust quasistatic finite elements and flesh simulation. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2005), pp. 181–190. 2, 3, 4

[Tut63] TUTTE W. T.: How to draw a graph. *Proceedings of the London Mathematical Society 3*, 1 (1963), 743–767. 5, 7

[XSZB15] XU H., SIN F., ZHU Y., BARBIČ J.: Nonlinear material design using principal stretches. *ACM TOG 34*, 4 (2015), 75:1–75:11. 2, 5

[ZBK18] ZHU Y., BRIDSON R., KAUFMAN D. M.: Blended cured quasi-newton for distortion optimization. *ACM TOG 37*, 4 (2018), 40:1–40:14. 2