EUROGRAPHICS 2019 / P. Alliez and F. Pellacini
(Guest Editors)

Volume 38 (2019), Number 2

A Subspace Method for Fast Locally Injective Harmonic Mapping

Eden Fedida Hefetz[1] Edward Chien[2] Ofir Weber[1]

[1]Faculty of Engineering, Bar-Ilan University, Israel
[2]CSAIL, MIT, USA

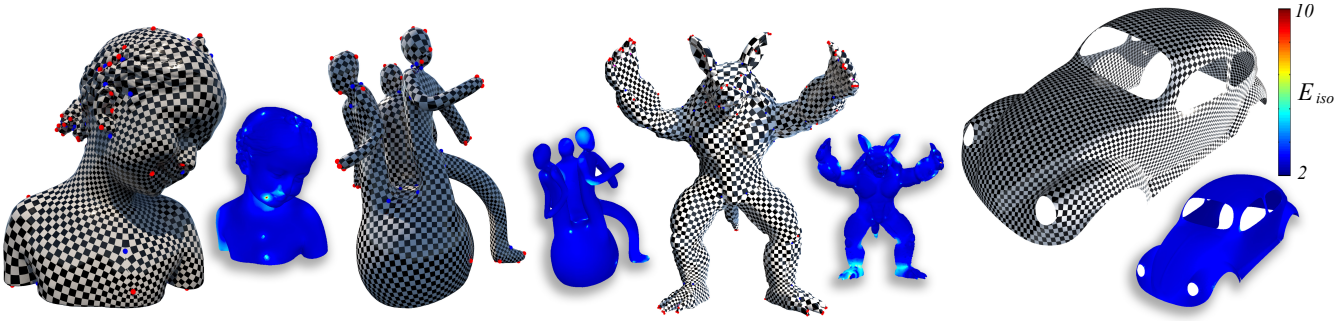

Figure 1: *Models parametrized with our method. Checkerboard textures are pulled back via the parametrizations and heat maps illustrate the symmetric Dirichlet energy* $E_{iso} = 0.5(\sigma_1^2 + \sigma_1^{-2} + \sigma_2^2 + \sigma_2^{-2})$. *The left three models contain between 40,000-100,000 triangles, and have cone singularities (colored spheres). Each took about a second to compute. The rightmost result contains 38,726 triangles, has no cones, and was computed in less than half a second. These represent an order-of-magnitude speedup over the solver of HGP [BCW17].*

Abstract

We present a fast algorithm for low-distortion locally injective harmonic mappings of genus 0 *triangle meshes with and without cone singularities. The algorithm consists of two portions, a linear subspace analysis and construction, and a nonlinear non-convex optimization for determination of a mapping within the reduced subspace. The subspace is the space of solutions to the Harmonic Global Parametrization (HGP) linear system [BCW17], and only vertex positions near cones are utilized, decoupling the variable count from the mesh density. A key insight shows how to construct the linear subspace at a cost comparable to that of a linear solve, extracting a very small set of elements from the inverse of the matrix without explicitly calculating it. With a variable count on the order of the number of cones, a tangential alternating projection method [HCW17] and a subsequent Newton optimization [CW17] are used to quickly find a low-distortion locally injective mapping. This mapping determination is typically much faster than the subspace construction. Experiments demonstrating its speed and efficacy are shown, and we find it to be an order of magnitude faster than HGP and other alternatives.*

CCS Concepts

• ***Computing methodologies*** → ***Mesh models; Mesh geometry models;*** • ***Mathematics of computing*** → *Topology;*

1. Introduction

The efficient computation of a surface parametrization lies at the foundation of geometry processing, and its solution is of great importance to many applications in graphics. Examples include remeshing, texture mapping, shape correspondence, fields-and-patterns design, compression, and high-level learning tasks, to mention just a few.

The problem is challenging because its mathematical formulation boils down to solving a nonconvex optimization problem in which the number of degrees of freedom is proportional to the number of mesh elements. In the common and important case of triangle meshes, models may consist of thousands to millions of elements, and each of these adds to the computational cost and complexity of parametrization.

The standard optimization problem that arises is minimization of some quality measure for angular distortion, area distortion, or smoothness over triangles. The measures that lead to the most satisfactory results are nonlinear/nonconvex. This poses an algorith-

Model name	#Triangles	#cones	#ATP iters	#Newton iters	Subspace construction time (sec)	ATP time (sec)	Newtom time (sec)	Total algorithm time (sec)	HGP time (sec)	Speedup	[Chien 2016] time (sec)	Speedup
retinal	7,282	18	2	12	0.06	0.00	0.07	0.14	0.81	×6	137	×1,005
homer	10,202	82	24	17	0.19	0.04	0.13	0.36	1.16	×3	43	×120
bumpy_sphere	11,444	10	1	9	0.07	0.01	0.06	0.15	1.22	×8	113	×756
smooth-feature	12,350	10	1	10	0.08	0.00	0.05	0.14	1.18	×8	113	×804
moai	20,000	24	2	14	0.14	0.00	0.09	0.23	1.97	×8	71	×304
pear	21,504	14	1	17	0.13	0.00	0.11	0.24	4.07	×17	1,174	×4,810
horse	39,698	52	4	17	0.30	0.00	0.11	0.42	4.75	×11	1,114	×2,684
armadillo	43,160	137	5	15	0.67	0.02	0.15	0.84	5.58	×7	477	×566
screwdriver	54,300	26	3	15	0.35	0.00	0.09	0.45	74.74	×166	199	×442
dilo	54,344	62	7	17	0.40	0.01	0.12	0.53	7.41	×14	6,143	×11,702
blade	58,546	70	3	16	0.50	0.01	0.11	0.62	26.91	×44	267	×432
uu-memento100k	99,932	107	19	16	0.98	0.02	0.13	1.13	11.00	×10	12,833	×11,387
sediapatch1_100K	99,968	36	4	15	0.65	0.00	0.10	0.75	12.48	×17	4,730	×6,298
pierrot100k	99,970	58	3	15	0.77	0.01	0.11	0.88	12.24	×14	892	×1,018
bimba100K	100,000	99	8	17	0.94	0.01	0.14	1.10	14.83	×14	1,138	×1,039
buste	100,000	71	5	16	0.74	0.01	0.12	0.86	11.70	×14	1,377	×1,594
camille_hand100K	100,000	72	4	15	0.81	0.01	0.11	0.93	12.44	×13	920	×994
eros100K	100,000	102	5	15	1.00	0.01	0.12	1.14	11.45	×10	5,348	×4,712
igea	100,000	80	5	14	0.89	0.01	0.11	1.01	17.43	×17	4,777	×4,753
magalie_hand	100,000	61	22	17	0.72	0.01	0.12	0.84	12.64	×15	905	×1,074
pensatore	100,000	64	3	15	0.83	0.01	0.11	0.94	11.67	×12	3,953	×4,201
ramses	100,000	32	3	15	0.66	0.00	0.09	0.76	35.10	×46	5,808	×7,652
torso	100,000	73	14	14	0.84	0.01	0.11	0.95	16.49	×17	16,886	×17,738
bunnyBotsch	111,364	44	1	9	0.87	0.00	0.06	0.94	131.76	×141	727	×778
santa	151,558	82	10	13	1.19	0.01	0.10	1.30	21.15	×16	14,339	×11,038
awakening	250,153	509	4	13	4.07	0.15	0.51	4.74	Fail	-	> 6 hours	-

Table 1: *Runtimes on a selection of 26 models with cone singularities from our experiments. Our method is an order of magnitude faster than HGP [BCW17], and 2–4 orders of magnitude faster than [CLW16]. Note that the majority of the runtime is spent during subspace construction, with ATP (alternating tangential projections) and PN (projected Newton) steps being very efficient.*

mic challenge as even small nonconvex problems are notoriously hard to solve. In the past decade, much work has been aimed at developing methods that are fast, robust, and can achieve strict constraints or guarantees of validity on the obtained result. Example constraints/guarantees include local injectivity conditions, user-specified bounds on the induced map distortion, as well as strict curvature constraints on the boundary or at singularities (cones).

These recent methods have constantly increased in robustness and speed, but are still significantly slower than "naive" methods that do not have quality or validity guarantees. Such methods are typically based on minimization of convex quadratic energies (e.g. [LPRM02, ZLS07, BGB08]) with or without additional linear constraints. Hence, their computational cost is dominated by that of solving a large sparse linear system with variable count proportional to the number of triangles. On the other hand, the recent robust nonconvex methods require solving tens or even hundreds of linear systems internally and are thus slower to compute.

There are several unique cases in which a linear method is both robust and capable of some guarantees on output. These can be limited in their scope and applicability. For example, the classic method of Tutte's [Tut63, Flo97] maps a disk-like mesh to a convex region bijectively and harmonically. The method of [GGT06] can parametrize a closed genus 1 surface bijectively via a discrete harmonic map. [AL15] uses the former result and generalizes the bijection such that it can be applied to genus 0 surfaces with either 3 or 4 cone singularities of a specific nature. As a last example, the recent method of [SC17] computes a discrete approximation to the *unique* conformal parametrization, given arbitrary prescription of cone singularities. It is more general than the former methods at the expense of not having strict injectivity guarantees (though it is empirically shown to be very robust).

We aim to develop a method with robustness and generality comparable to that of nonconvex methods, with a computational cost similar to that of linear methods. We search within the linear space of discrete harmonic maps with arbitrary (not convex) boundaries, and any number of cones with user-prescribed curvature. This space is a superset of the space of conformal maps (in a smooth sense) and was recently introduced by the Harmonic Global Parametrization (HGP) method [BCW17]. The additional desired constraint of local injectivity is nonconvex, but an efficient search is possible with a maximal convex subset, based on principles borrowed from [Lip12]. In HGP, the linear harmonicity constraints are softened with a quadratic error term, and the resulting convex program is solved with Mosek [ApS17], an off-the-shelf second-order cone solver. HGP is 1–2 orders of magnitude faster than alternative methods with similar robustness such as [CLW16], which also uses convex programming but requires solving many convex problems. Our algorithm provides an additional significant acceleration over HGP (Table 1).

In this paper, we construct a custom-made solver for searching within the harmonic space of [BCW17] without relaxing these harmonic constraints. We produce a method whose computational cost is comparable to that of 1-2 linear solves. Following HGP, our method is used to generate seamless parametrizations, but it may

also be applied in the case of arbitrary cone angles and holonomies (Observation 1 below holds by continuity in this case).

First, we perform a detailed algebraic analysis of the HGP linear system, $A_{hgp}\mathbf{z} = 0$, for genus 0 meshes, where $\mathbf{z}$ denotes a uv layout (Section 3). A_{hgp} is shown to be full rank and as it has more columns than rows, we characterize its nontrivial null space, $\mathcal{N}(A_{hgp})$, the space of harmonic maps from HGP. The dimensionality of this space is very low and scales favorably with respect to increasing mesh resolution: linearly with the number of cones and boundary vertices. Next, we rely on the following two observations to further reduce our problem:

Observation 1 For any harmonic map $\mathbf{z}$, determination of local injectivity can be formulated solely in terms of the behavior of the map around the boundary and cones [BCW17, Theorem 6.1].

Observation 2 The conformal distortion of a (smooth) locally injective harmonic map attains its maximum on the boundary of a disk-like domain [CW15]. Furthermore, this result was recently extended to multiply-connected domains [CW17]. The latter result can be immediately generalized to surfaces with cones which can be interpreted as infinitesimal punctures.

These observations allow us to control global injectivity and distortion by focusing on just the vertices neighboring the cones and boundary. For any basis element of $\mathcal{N}(A_{hgp})$, we need only extract those entries (uv positions), and may impose injectivity constraints and minimize distortion on this very small subset of the mesh. Again, this scales favorably with increasing mesh resolution.

This is referred to as a "reduced" subspace construction and extraction of these entries is described in Section 4. We use an algorithm that computes selected elements from the inverse of a nonsingular large sparse matrix [KLS13, VCKS17]. To the best of our knowledge this is the first use of a selected inversion algorithm for geometry processing (see Section 2 for a brief discussion), and as a service, Appendix D details our use of this algorithm in the PARDISO package [Sch]. Straightforward computation of $\mathcal{N}(A_{hgp})$ in its entirety would be orders of magnitude slower, drawing the method inefficient. Direct methods for null space basis computation such as: use of a sparse LU factorization, QR/SVD orthonormal basis extraction, and the sparse LUQ algorithm from [J*18], were tried and found to be much slower in our use case.

Once the reduced subspace is constructed, we impose local injectivity constraints and minimize distortion. For local injectivity, we again use the frames of [Lip12] to specify a maximal convex subset of locally injective maps, as was done in HGP. A feasible map in the intersection of this convex subset and $\mathcal{N}(A_{hgp})$ is found with an alternating tangential projection method [HCW17] (Section 5). The intersecting affine and convex subsets differ from the ones used in [HCW17], but the guarantees and efficiency of tangential projections are shared. It provably converges when the intersection is nonempty, and robust methods for frame determination usually achieve this (further discussion in Section 7.3). For distortion minimization, the projected Newton solver of [CW17] is used to drive down the distortion of the initial feasible map by lowering isometric energy (Section 6). The main advantage of this solver is its analytic projection of the Hessian to the positive semidefinite cone.

In Section 7 we demonstrate that our solver is several orders of magnitude faster compared to competing nonlinear state-of-the-art methods. The cost of finding an initial feasible map, and of distortion minimization is relatively negligible compared to that of the reduced subspace construction (which is also very fast). The speedup over other solvers is mostly attributed to the fact that they internally perform a handful of linear solves while the cost of our entire runtime is on the order of just 1-2 such solves.

2. Previous Work

The literature on mesh parametrization, or more generally the computation of mappings, is vast. In the following we will review only the previous approaches most relevant to ours and refer the interested reader to the excellent surveys [FH05, HLS07] for more complete expositions. We also recommend the surveys [BLP*13, VCD*16] for more specific global parametrization approaches, quadrangular remeshing, and directional field design as these are closely related to the types of parametrization supported by our method.

Similar to our method, a large range of parametrization techniques restrict the space of admissible maps to a subspace. A popular choice is that of angle-preserving (conformal) maps. These differ by the choice of variables in which the underlying optimization is formulated in. The angle-based flattening (ABF) [SdS01, SLMB05] uses angles of the image triangles as variables which result in a nonconvex optimization. [KSS06] uses triangle circumcircle radii as variables, while [SSP08, BGB08] employ conformal scale factors at the vertices and aim at finding the scale that will induce a new metric that is (discretely) conformally equivalent to the original metric while having a prescribed Gaussian curvature. The linear method of [BGB08], which can be understood as an approximation to [SSP08], flattens the metric only approximately, but both methods tend to lack robustness in extreme cases and are sensitive to the quality of the underlying mesh. On the other hand, methods that use spatial coordinates result in a metric that is completely flat as long as the induced map is locally injective. The method of [SC17] tries to benefit from both worlds by first using the conformal factors as variables to induce a nearly flat metric, and then switches to the spatial variables in which conformality is discretized based on the Cauchy-Riemann equations with a pair of harmonic conjugate functions. We note that while conformal maps possess elegant theory and many useful mathematical properties, they are somewhat more restricted than necessary, sometimes engendering large isometric (scale) distortion.

The space of harmonic maps offers a natural generalization to conformal maps as each of the components of a conformal map is harmonic. Methods that operate in this space are mostly based on spatial variables [Tut63, Flo97, GGT06, TACSD06, WZ14, AL15, CW15, CW17, BCW17], but recently, it was shown in [LW16] that similar to conformal maps, alternative variables can be used, for which the space of locally injective maps becomes convex. This fact was further used by [HCW17] for shape deformation and by [CCW16a] for shape interpolation.

More general parametrization and deformation methods operate in the larger space of continuous piecewise affine maps. They allow each vertex of the mesh to move freely. Since they cannot

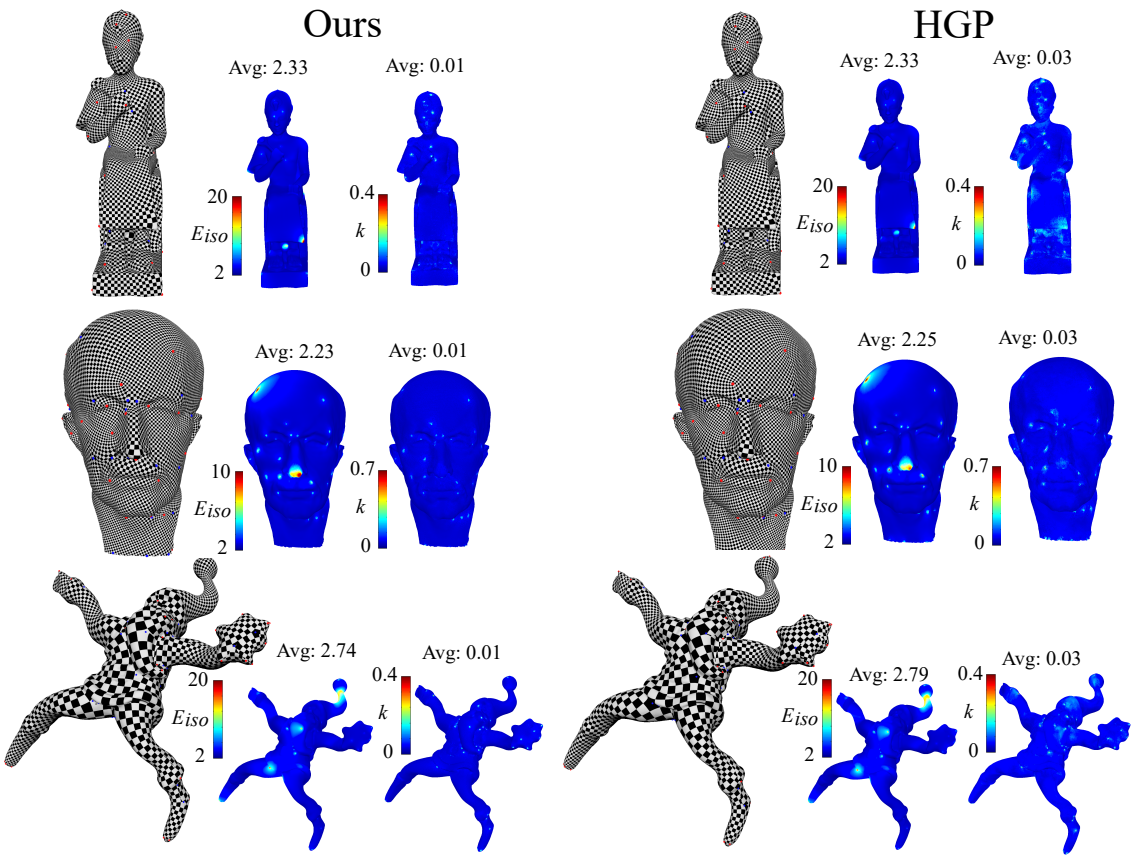

Figure 2: *Qualitative and Distortion Comparison to HGP [BCW17]. Textured results from our database are shown, with heat maps depicting symmetric Dirichlet energy E_{iso} and the conformal distortion $k : |f_{\bar{z}}| / |f_z|$. Our method is slightly better on E_{iso}, and three times better (on average) on k, and achieves these results much faster.*

rely on underlying properties of a specific family of maps, they require more care in order to guarantee certain properties such as local injectivity and/or bounded distortion. Namely, such constraints should be applied to every individual element of the mesh. The method of [Lip12] forces such constraints with a convex framework by carving a convex piece out of the nonconvex high-dimensional space of bounded distortion mappings with the help of frames. The algorithm uses sequential convex programming, where at each iteration a convex program is solved followed by an update of the frames which enables gradual exploration of the nonconvex space. A similar approach was used in [APL14, APL15] for computing shape correspondences. The advantage of these methods is that they can generate arbitrary maps including nonsmooth maps, and have the potential for a larger feasible space. This comes at a higher computational cost. The method of [CCW16a] uses the metric directly in the form of edge lengths squared. Distortion is convex in these variables, but the curvature conditions are not. A sequential convex programming is employed in order to gradually flatten the metric while adhering to the distortion bounds.

Another line of works that use the complete space of continuous piecewise affine maps, addresses injectivity without explicitly formulating a constraint. Instead, the optimization starts from a locally injective configuration and an energy with a builtin barrier term is used. An unconstrained minimization is applied based on either first order approximation [SS15, RPPSH17, CBSS17] or second order approximation [FLG15, SPSH*17, CW17]. In Section 6 we employ such a method to further optimize our map. In [ZBK18], the authors develop a barrier filter for descent direction that avoids collapse of elements more efficiently. A large number of additional works develop acceleration techniques for optimizations in this more general space including: momentum acceleration and a quadratic proxy [KGL16]; progressive modification of the reference mesh [LYNF18]; and analytic Newton projection and initialization scaling [GSC18].

Many parametrization methods address the problem of seamless parametrization specifically by incorporating cone placement and holonomy determination. These include [BGB08, BZK09, MZ12, MZ13, MPZ14, DVPSH15, CLW16, BCW17]. In our paper, we assume similarly to [FL16, CLW16, BCW17] that the cone points and their angles have been given, and aim to achieve these.

Selected inversion is a family of methods used to compute a user-specified subset of A^{-1}, without computing it in its entirety. A sparse direct method was initially shown in [Tak73].

However, efficient algorithms for general matrices (with public implementations) have only been developed in the past decade [LYM*11, KLS13, JLY17, VCKS17]. These methods use a sparse factorization of A (LU or LDL^T depending on whether the matrix is symmetric or not) and compute supersets of the required elements efficiently and in parallel. We have used the implementation that is provided by the PARDISO parallel direct solver [Sch] which is based on the algorithms described in [KLS13, VCKS17]. Within graphics, the work of [HDA17] used sparse Cholesky factorization insights to perform fast parametrization of localized patch mesh regions. In contrast, the structure of our problem is non-local, with boundaries spanned along the entire surface and cones scattered all over the mesh. Instead, we are able to leverage observations 1 and 2, as well as the selected inverse algorithm, for global parametrization with discrete harmonic maps.

3. HGP System Analysis

We begin with a deeper analysis of the linear harmonic system from [BCW17]. In this paper, a *seamless global parametrization* of an oriented triangular mesh surface $S = (V, E, T)$, potentially with boundary $\partial S = (\partial V, \partial E)$, is desired. An example is illustrated in Figure 3. A seamless global parametrization consists of a piecewise linear map $f : S_c \to \mathbb{R}^2$, where S_c is a mesh disc obtained by cutting S along a seam graph $G_s = (V_s, E_s)$. Furthermore, for each edge $e_{ij} \in E_s$, f must map the duplicates (from cutting) e^a_{ij}, e^b_{ij} to image segments that differ by an orientation-preserving isometry of $\mathbb{R}^2$, with rotational part $e^{r_{ij}(\pi \mathrm{i}/2)}$ (with $\mathbb{R}^2 \cong \mathbb{C}$ implicit). If f is locally injective over the interior of S_c, then the integers $r_{ij} \in \{0,1,2,3\}$ determine the holonomy for the cone metric induced by pullback by f. Correspondingly, given a cone metric with holonomies that are $\pi/2$ (also called *seamless*), cutting it to a disc and laying it flat will produce a seamless global parametrization.

Let us recall briefly the Harmonic Global Parametrization (HGP) system. The variables are denoted $z_i := f(v_i)$, where we have again utilized the standard homeomorphism $\mathbb{R}^2 \cong \mathbb{C}$ to allow for more compact notation. Let $C \subset V_s$ denote the set of cones and $\mathring{V} := V \backslash (V_s \cup \partial V)$ denote the interior vertices of S_c. Further notational explanation follows presentation of the system.

$$z^a_j - z^a_i = e^{\mathrm{i}\frac{\pi r_{ij}}{2}}(z^b_j - z^b_i), \qquad e_{ij} \in G_s \tag{1}$$

$$\sum_{v_j \in N(v_i)} w_{ij}(z_i - z_j) = 0, \qquad v_i \in \mathring{V} \tag{2}$$

$$\sum_{v_j \in N^*(v^0_i)} w_{ij}(z^0_i - z_j) + \sum_{v_j \in N^*(v^1_i)} w_{ij} e^{\mathrm{i}\frac{\pi r_1}{2}}(z^1_i - z_j) = 0, \qquad v_i \in V_s \backslash (C \cup \partial V) \tag{3}$$

Conditions (1) are the rotation constraints for seam edges $e_{ij} \in E_s$, where v^a_i & v^a_j and v^b_i & v^b_j denote the endpoints of e^a_{ij} and e^b_{ij}, respectively. In [BCW17], the integers r_{ij} were determined by solving a linear system from Appendix B of [MZ12], and the same is done here. Conditions (2) are the standard harmonic constraints for interior vertices of S_c, where $N(v_i)$ denotes the set of adjacent vertices of v_i in S_c, and w_{ij} denote cotangent weights for the Laplacian. Conditions (3) are the rotated harmonicity constraints for seam vertices, and have only been expressed for vertices of degree 2 (in G_s), for simplicity. Such vertices have two duplicates v^0_i and v^1_i after cutting, with images denoted z^0_i and z^1_i. Figure 3 contains a schematic explaining the notation. Generally, for a vertex of degree d, cutting results in d duplicates, and the analogous equations are quiet cumbersome to state, but simple to understand. We refer the reader to Appendix A of [BCW17] for more details but note here that there is no need to explicitly construct conditions (3) in order to implement our method, as explained in Section 4.

Following Observation 1, it is sufficient, but not necessary, that the weights w_{ij} be positive for guarantee of local injectivity (assuming injectivity at boundary and cones). One possibility is to utilize the mean value coordinates [Flo03]. However, we opted for use of cotangent weights due to their convergence properties with respect to smooth harmonic maps, as well as the fact that their symmetry property leads to easier implementation and simplified analysis and construction of the harmonic subspace. To guarantee positivity of the cotangent weights, it is possible to construct an intrinsic Delaunay triangulation [FSSB07]. Nonetheless, we found in our experiments (Section 7) that the method is rarely influenced by negative cotangent weights and when it is, a few isolated flipped triangles emerge. We fixed these locally by repositioning a single vertex.

3.1. System Dimensions & Interpolation Conditions

The linear system presented by Equations (1)–(3) has fewer equations than variables and forms a homogeneous linear equation $A_{hgp}\mathbf{z} = 0$. In the next two subsections, we aim to characterize the null space $\mathcal{N}(A_{hgp})$. Having a basis for this null space will allow us to parametrize the space of solutions. Doing it efficiently is a key contribution in this paper. We begin with a simple determination of the dimensions of A_{hgp}, assuming that S is a sphere or a multiply-connected disc (so is of genus 0, with $n \in \mathbb{N}$ boundary components). While the theory of [BCW17] is readily applicable to the higher genus case, we leave its characterization and algebraic analysis for future work.

3.1.1. Sphere Case

Suppose S has no boundary components, so that it is topologically a sphere. We first note the following lemma:

Lemma 3 For $S \cong \mathbb{S}^2$, the seam graph G_s is a tree, and thus $\chi(G_s) = |V_s| - |E_s| = 1$.

Proof Cutting along G_s results in a disc S_c. If G_s is disconnected, multiple boundaries result, giving a contradiction. Furthermore, G_s may have no cycles, otherwise cutting would yield a disconnected set. Finally, all trees have Euler characteristic $\chi = 1$. □

The number of complex variables in the system is the number of vertices of S_c. There is one for each interior vertex and $\deg(v)$ copies for each $v \in V_s$. Thus, the column dimension of A_{hgp} is $|\mathring{V}| + 2|E_s|$. For the row dimension (number of relations), we get $|E_s| + |\mathring{V}| + (|V_s| - |C|)$, with each term from relations (1)–(3), respectively. The difference in row and column dimension is then $|E_s| - |V_s| + |C| = |C| - 1$ by Lemma 3.

In Section 3.2, we will argue that A_{hgp} is full rank, by arguing

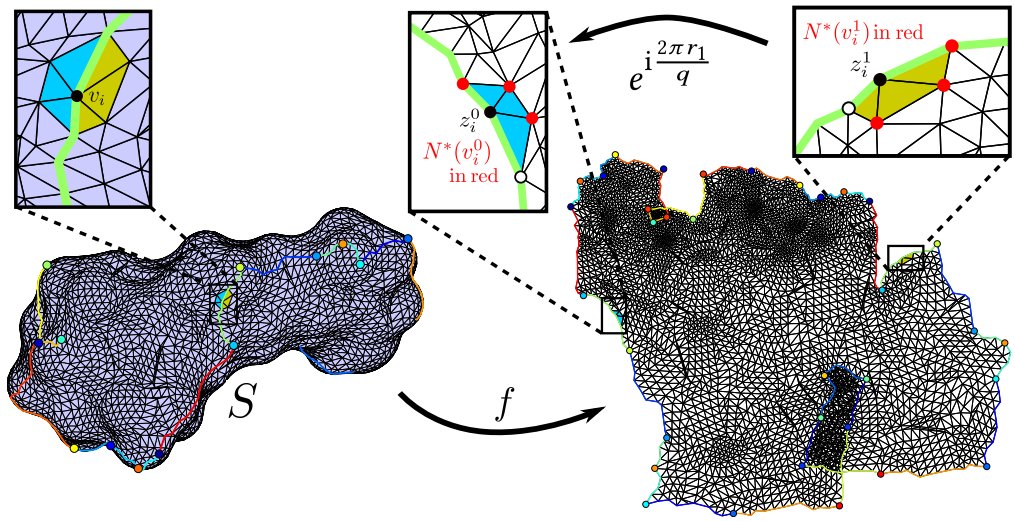

Figure 3: *An example seamless global parametrization, and a schematic illustrating notation for Equation (3). Figure from [BCW17]. The mesh surface S is mapped with parametrization f, and the image of S_c is depicted on the right. Cone points are represented by colored dots and the seam graph G_s is illustrated by colored edges. The resulting duplicate vertices and edges are correpondingly colored on the right (some repeat of colors). The upper half of the figure establishes notation for Equation (3) for a non-cone seam vertex v_i of degree 2. As can be seen, cutting along the seam splits its neighborhood into two components, and these rotated harmonicity conditions ask that f be locally harmonic up to the appropriate rotational correction.*

this for an augmented matrix:

$$A_{aug} := \begin{pmatrix} A_{hgp} \\ A_{int} \end{pmatrix} = \begin{pmatrix} A_{2,3} \\ A_1 \\ A_{int} \end{pmatrix},$$

where we've used $A_{2,3}$ and A_1 to denote the submatrices corresponding to Equations (2), (3) & Equations (1), respectively. The submatrix A_{int} contains $|C|-1$ *interpolation conditions* that "complete" A_{hgp} and make A_{aug} square.

To specify these interpolation conditions, first we find a cone that is of degree 1 in G_s. This must exist as G_s is a tree, and we may assume there are no non-cone leaves (these would unnecessarily introduce extra variables). For each of the other $|C|-1$ cones, we specify its position under f (or the position of one of its duplicates, if $\deg(v) > 1$ in G_s). This results in A_{int} consisting of rows that are all zero except for a 1 in the appropriate column index. Any solution $\mathbf{z}$ to $\begin{pmatrix} A_{hgp} \\ A_{int} \end{pmatrix}\mathbf{z} = \begin{pmatrix} 0 \\ c_{int} \end{pmatrix}$, where $c_{int} \in \mathbb{C}^{|C|-1}$, gives an element of $\mathcal{N}(A_{hgp})$. Linearly independent choices of c_{int} result in linearly independent solutions $\mathbf{z}$, allowing us to characterize $\mathcal{N}(A_{hgp})$. The positions of the cones (or one of their duplicates) under f are "harmonically" interpolated by the solution to the augmented system, motivating the "interpolation" moniker.

3.1.2. Disc & Punctured Disc Cases

The second case that we analyze is that of $n > 0$ boundary components, where S is a disc or multiply-connected disc. We establish two small lemmas characterizing G_s. The first is argued just as Lemma 3 is. The proof of the second lemma is more involved, and left to Appendix A.

Lemma 4 For S of genus 0 and $n > 0$ boundaries, with G_s having m connected components, G_s is a forest and $\chi(G_s) = |V_s| - |E_s| = m$.

Lemma 5 For S of genus 0 and $n > 0$ boundaries, with G_s having m connected components, $|\partial V \cap V_s| = n + (m-1)$.

With the column dimension, we again have one complex variable for each vertex of $\mathring{V}$ and each vertex of $\partial V \backslash V_s$. For vertices of $V_s \backslash \partial V$, there are $\deg(v)$ copies and corresponding variables. For vertices in $\partial V \cap V_s$, there are $\deg(v)+1$ variables. Thus the column dimension is $|\mathring{V}| + |\partial V| + 2|E_s|$. For the row dimension, we have $|E_s| + |\mathring{V}| + (|V_s| - |C| - |\partial V \cap V_s|)$, with each term from relations (1)–(3), respectively. Subtracting the two and utilizing Lemmas 4 and 5, we get a dimension difference of $|\partial V| + |C| + n - 1$. If we have $n = 0$ boundaries and are in the sphere case, this formula matches the result from the previous subsection.

The interpolation conditions here are similar to those in the sphere case. For each vertex of ∂V, we specify its position (or that of one of its copies if it's also in V_s). For each cone, we do the same. If $n > 1$, there are still additional rows of A_{int} to specify: $n-1$ to be exact. By Lemma 5, $|\partial V \cap V_s| = n + (m-1)$, and for each of these vertices, there are at least two duplicate copies after cutting. For $n-1$ of these we will specify the positions of another duplicate copy (one having already been specified). These are chosen such that the remaining m elements of $\partial V \cap V_s$ are distributed evenly amongst the m components of G_s. These constraints lead to rows of zeroes with a 1 in the appropriate index, as was in the sphere case.

3.2. A KKT system

In this section, we make a detailed argument for full rank of the augmented system. Consider the KKT system arising from the lin-

early constrained quadratic minimization:

$$\underset{\mathbf{z}=(z_1,z_2,\ldots)}{\text{minimize}} \quad \|D\mathbf{z}\|^2 \tag{4}$$
$$\text{subject to} \quad \begin{pmatrix} A_1 \\ A_{int} \end{pmatrix} \mathbf{z} = \begin{pmatrix} 0 \\ c_{int} \end{pmatrix}$$

where $\mathbf{z}$ denotes the images of the vertices of S_c under f; and D denotes a pair of (square root) area-weighted gradient operators, producing a pair of gradients for each triangle from the x- and y-coordinate values ($\mathrm{Re}(z_i)$ and $\mathrm{Im}(z_i)$) at the vertices. The objective is the sum of Dirichlet energies for the component coordinates over the mesh, and the energy's Hessian $2D^tD$ is simply a pair of standard cotangent Laplacians, one over each component [BS07].

Written in terms of real variables $x_i := \mathrm{Re}(z_i)$ and $y_i := \mathrm{Im}(z_i)$, the objective is positive-semidefinite with linear constraints. Thus, KKT conditions are both necessary and sufficient for global optimality [BV04]. This produces a linear KKT system:

$$\begin{pmatrix} 2D^tD & A_{lin}^t \\ A_{lin} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{z} \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ c_{lin} \end{pmatrix}, \tag{5}$$

where $A_{lin} := \begin{pmatrix} A_1 \\ A_{int} \end{pmatrix}$, $c_{lin} := \begin{pmatrix} 0 \\ c_{int} \end{pmatrix}$, and λ are Lagrange multipliers.

3.2.1. Invertibility

Let K denote the square matrix in Equation (5). We may show invertibility of K by arguing that A_{lin} is full rank and $\mathcal{N}(D) \cap \mathcal{N}(A_{lin}) = \emptyset$. To see this, suppose there exists $\begin{pmatrix} \mathbf{z} \\ \lambda \end{pmatrix} \in \mathcal{N}(K)$ not equal to zero. Then:

$$\begin{aligned} 2D^tD\mathbf{z} + A_{lin}^t\lambda &= 0 \\ A_{lin}\mathbf{z} &= 0 \end{aligned} \tag{6}$$

Thus, we have $\mathbf{z} \in \mathcal{N}(A_{lin})$. If $\mathbf{z} = 0$, then we have $\lambda \in \mathcal{N}(A_{lin}^t)$ which is nonzero, contradicting full rank of A_{lin}. If $\mathbf{z} \neq 0$, then we can multiply Equation (6) on the left by $\mathbf{z}^t$, causing the second term to disappear and implying that $2\|D\mathbf{z}\|^2 = 0$. Thus, $\mathbf{z} \in \mathcal{N}(D) \cap \mathcal{N}(A_{lin})$, a contradiction.

Theorem 6 *K is invertible, and optimization problem (4) has a unique solution.*

Proof Proposition 11 in Appendix B proves that A_{lin} is full rank. For the other condition, note that $\mathcal{N}(D)$ consists of the $\mathbf{z}$ that are constant vectors, as this is the only way the gradients in each component will vanish entirely. Of these, only the constant zero vector is in $\mathcal{N}(A_{lin})$ due to the interpolation constraints, which pick out the coordinates of the cone positions. □

3.2.2. Full Rank of A_{hgp}

We now relate the KKT system to the augmented HGP system. Consider a solution $\begin{pmatrix} \mathbf{z} \\ \lambda \end{pmatrix}$ of Equation (5). Looking at the rows of K corresponding to interior vertices of S_c, it's clear that $\mathbf{z}$ satisfies Equations (2). A row of A_{lin}^t (column of A_{lin}) is nonzero iff it corresponds to an index of a vertex in G_s or ∂V.

For rows corresponding to vertices in $V_s \backslash (C \cup \partial V)$, the rows of A_{lin}^t are non-zero and the equations satisfied contain Lagrange multipliers. In general, elimination of these Lagrange multiplier variables, will show that $\mathbf{z}$ satisfies Equations (3). The argument is tedious to express in general, so we simply demonstrate it specifically for an example with vertex v_i from Figure 3. Notation is set in Figure 4. We use the x-, y-component real values to be more explicit.

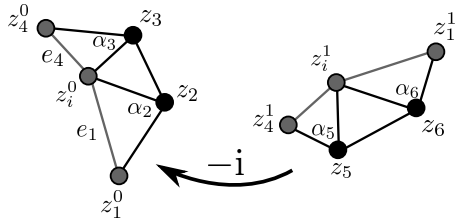

Figure 4: *Notation for an example of eliminating Lagrange multiplier variables. The $-\mathrm{i}$ denotes the rotation that should be applied to match the two halves.*

First we show the relevant relations from A_1 (for seam edges e_4 and e_1) and display the corresponding Lagrange multipliers (preceding the equations):

$$\begin{aligned} \lambda_4^x: &\quad (x_4^0 - x_i^0) + (-y_4^1 + y_i^1) = 0 \\ \lambda_4^y: &\quad (y_4^0 - y_i^0) + (x_4^1 - x_i^1) = 0 \\ \lambda_1^x: &\quad (x_i^0 - x_1^0) + (-y_i^1 + y_1^1) = 0 \\ \lambda_1^y: &\quad (y_i^0 - y_1^0) + (x_i^1 - x_1^1) = 0. \end{aligned}$$

This allows us to express the relations from the top half of the KKT system corresponding to the components of z_i^0 and z_i^1.

$$\begin{aligned} \text{row } x_i^0: &\quad \frac{1}{2}\cot(\alpha_2)(x_i^0 - x_1^0) + w_{i2}(x_i^0 - x_2) + w_{i3}(x_i^0 - x_3) \\ &\quad + \frac{1}{2}\cot(\alpha_3)(x_i^0 - x_4^0) - \lambda_4^x + \lambda_1^x = 0 \\ \text{row } y_i^0: &\quad \frac{1}{2}\cot(\alpha_2)(y_i^0 - y_1^0) + w_{i2}(y_i^0 - y_2) + w_{i3}(y_i^0 - y_3) \\ &\quad + \frac{1}{2}\cot(\alpha_3)(y_i^0 - y_4^0) - \lambda_4^y + \lambda_1^y = 0 \\ \text{row } x_i^1: &\quad \frac{1}{2}\cot(\alpha_5)(x_i^1 - x_4^1) + w_{i5}(x_i^1 - x_5) + w_{i6}(x_i^1 - x_6) \\ &\quad + \frac{1}{2}\cot(\alpha_6)(x_i^1 - x_1^1) - \lambda_4^y + \lambda_1^y = 0 \\ \text{row } y_i^1: &\quad \frac{1}{2}\cot(\alpha_5)(y_i^1 - y_4^1) + w_{i5}(y_i^1 - y_5) + w_{i6}(y_i^1 - y_6) \\ &\quad + \frac{1}{2}\cot(\alpha_6)(y_i^1 - y_1^1) + \lambda_4^x - \lambda_1^x = 0. \end{aligned}$$

As can be seen, combining rows x_i^0 & y_i^1 and rows y_i^0 & x_i^1 gives us Equation (3) for vertex v_i. In general, a vertex of degree k will require $2k$ rows to be combined to achieve the corresponding rotational conditions by eliminating Lagrange multipliers.

Finally, the bottom half of the KKT system is exactly Equations (1) and the interpolation conditions, so it is clear that $\mathbf{z}$ also satisfies these equations, and is a solution of $A_{aug}\mathbf{z} = \begin{pmatrix} 0 \\ c_{int} \end{pmatrix}$. We have argued for the following proposition:

Proposition 7 Given a solution $\begin{pmatrix} \mathbf{z} \\ \lambda \end{pmatrix}$ of the KKT system, $\mathbf{z}$ is a solution of the augmented HGP system.

Invertibility of K also implies linear independence of its rows. As noted, Equations (2), Equations (1), and the interpolation conditions are already present as rows. Equations (3) can be obtained by

elimination of Lagrange variables. This amounts to linear combinations of the rows corresponding to duplicates of the seam vertex, and the resulting equations will still be linearly independent. Thus, via invertibility of K we have shown the following theorem:

Theorem 8 A_{aug} is invertible, A_{hgp} is of full rank, $\dim(\mathcal{N}(A_{hgp})) = 2(|\partial V| + |C| + n - 1)$ (real-dimensional), and a basis for $\mathcal{N}(A_{hgp})$ may be found by solving $A_{aug}\mathbf{z} = \binom{0}{c_{int}}$ for a set of c_{int} that form a basis for $\mathbb{R}^{2(|\partial V|+|C|+n-1)}$.

4. Subspace Construction

Theorem 8 and Proposition 7 give us a way to construct $\mathcal{N}(A_{hgp})$. In particular, for choices of c_{int} from the standard basis for $\mathbb{R}^{2(|\partial V|+|C|+n-1)}$, we use the parallel selected inversion algorithm [KLS13, VCKS17] to compute selected entries of the inverse of the matrix K. Note that if c_{int} is an element of the standard basis, then the solution to the system is merely a column of K^{-1}. The resulting $\mathbf{z}$'s will form a basis and vectors of interpolation constraints in $\mathbb{R}^{2(|\partial V|+|C|+n-1)}$ specify elements of $\mathcal{N}(A_{hgp})$.

Using the KKT system formulation instead of the augmented HGP system greatly eases implementation. This is because we do not need to explicitly construct the cumbersome rotated harmonicity conditions (3) (nor (2)). Instead, we simply construct the Dirichlet energy's Hessian $2D^tD$ over the cut-to-disk mesh S_c in the same fashion the standard Laplacian is constructed. Namely, we iterate over all the triangles in S_c and for each triangle, we add the 3 corresponding cotangent weights to the upper left block of K. These correspond to the gradients of the Dirichlet energy with respect to the function value at the triangle vertices.

4.1. Reduced Variables

We do not need to use the selected inversion to extract all entries of the relevant columns of K^{-1}. Recall that the main theorem of [BCW17] told us that a solution to the HGP system can be proven to be locally injective (over the interior of S_c) by looking merely at the behavior of the triangles near cone points and boundary points. With positive Laplacian weights, the theorem was:

Theorem 9 Let f denote a solution of the HGP system for S, with specified cone points and holonomy angles determining the rotation constraints. If the cone and boundary triangles are mapped in an orientation-preserving manner, and the induced metric on S achieves the desired cone angles and turning angles, then f is locally injective (over the interior of S_c).

Specifically cone/boundary triangles refer to triangles of S containing cone/boundary vertices, respectively. We refer the reader to Appendix C of [BCW17] for a specific definition for turning angle, but intuitively, it tracks the amount that the image of a boundary curve (under f) turns. Above, we omitted some technical conditions (like a 3-connected 1-skeleton) that practically always hold.

This theorem allows us to drastically reduce the number of variables when searching for a locally injective map by just tracking the vertices of cone and boundary triangles. This decouples the variable count from mesh density, and instead ties it to the number of cone and boundary vertices.

Ultimately, we use the selected inversion to extract just the rows corresponding to cone and boundary vertex positions. This method of subspace construction leads to an order of magnitude speedup over an approach where K is factorized and multiple linear solves are performed to extract full columns of K^{-1}. The selected inversion algorithm still requires a factorization of similar computational cost, but is able to extract the greatly reduced number of variables much more quickly than solving and throwing out elements.

Lastly, let us note that we never explicitly construct the full basis for $\mathcal{N}(A_{hgp})$. In Sections 5 and 6, we optimize over the interpolation constraints c_{int}, and to obtain the final map, we solve the KKT system with the optimized c_{int}, utilizing the factorization from application of the selected inversion algorithm. This again gives us greater computational efficiency.

4.2. Virtual Vertices for Boundaries

For closed meshes, the dimension of the subspace depends on the number of cones but is independent of $|V|$. For open meshes, the dimension depends on the boundary as well. While ∂V is typically significantly smaller than V ($\mathcal{O}(|\partial V|) = \mathcal{O}(|V|^{1/2})$ for a unit disk) for very dense meshes with long boundaries, the dimension increase may lead to a slowdown. To alleviate this, we specify optional "virtual vertices" along ∂S, which are a small subset of ∂V consisting of every k-th vertex along the boundary, where k is a user-defined parameter. In the resulting mapping, the position of any boundary vertex v between two virtual vertices v_a and v_b is constrained to a convex combination of the two. Specifically, $v = (1-t)v_a + tv_b$, where t is the arclength along ∂S from v_a to v divided by the arclength from v_a to v_b. In the case that the mesh is *not* 3-connected, the virtual vertices should be selected more carefully to avoid degeneracies. For example, if the mesh contains an "ear" triangle, its 3 consecutive boundary vertices must be included in the virtual vertices subset. More general treatment is almost never needed in practice but is described in Appendix E for completion.

This procedure leads to a smaller subspace of $\mathcal{N}(A_{hgp})$, accelerating computations. It is applied sparingly in our experiments (only when there are > 1000 boundary vertices; see Table 2), and is only needed for very large meshes with long boundaries. The quality of the resulting map is usually not greatly affected, i.e., only in the rare case when the boundary is very jagged. Our tactic is similar to the "seam straightening" of [LFJG17], and distortion increases might be mitigated with strategies similar to ones found there.

5. ATP in Reduced Variables

The method of Alternating Tangential Projections (ATP, for short) was used in [HCW17] to converge to the intersection of a linear subspace and a convex subset, projecting "globally" onto the subspace and then "locally" onto the convex subset. It differs from standard alternating projection methods in that the global projection is normal to the previous local projection step (except the first step, of course). This leads to faster convergence times empirically [HCW17], and is guaranteed to converge on a feasible intersection point, if it exists.

In our setting, we apply this method with a focus on the

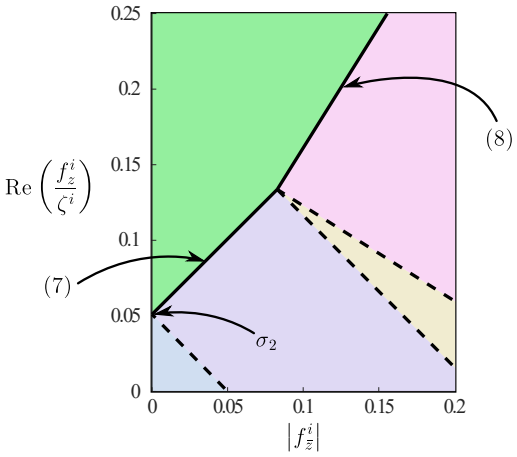

Figure 5: *A schematic illustrating $\mathfrak{B}_i$. Due to symmetries of inequalities (7) and (8), they are illustrated in the $\left(\mathrm{Re}\left(f_z^i/\zeta^i\right), \left|f_{\bar{z}}^i\right|\right)$-plane. The inequalities are represented by solid black lines. The dashed lines are orthogonal to them, and the green region represents $\mathfrak{B}_i$. In this image, $k = 0.6$ and $\sigma_2 = 0.05$. To visualize the full space, one would rotate the shaded green region about the $\mathrm{Re}\left(f_z^i/\zeta^i\right)$-axis and then product it with a copy of $\mathbb{R}$ for the potential values of $\mathrm{Im}\left(f_z^i/\zeta^i\right)$.*

cone and boundary triangles, denoted T_{cb} henceforth. The subsets we consider are within $\mathbb{C}^{2|T_{cb}|} \cong \mathbb{R}^{4|T_{cb}|}$. A point in this space $(f_z^1, \ldots, f_z^{|T_{cb}|}, f_{\bar{z}}^1, \ldots, f_{\bar{z}}^{|T_{cb}|})$ is to be interpreted as a candidate collection of similarity and antisimilarity parts of the Jacobians for the triangles of T_{cb}. These are the complex Wirtinger derivatives (see [CCW16b, Section 3.1]). The linear subspace projected to globally is the set of such Jacobian components which actually arise as solutions of our HGP system. Ideally, the local projection would be onto the set of orientation-preserving Jacobians $\left|f_z^i\right| > \left|f_{\bar{z}}^i\right|$, but this is a nonconvex subset.

In order to try to achieve these aims, we find a maximal convex subset dictated by convexification frames [Lip12]. The frames are determined in a fashion which encourages the correct cone and turning angles. In particular, we utilize frames determined with the frame field from [BZK09], as was done in [BCW17]. Despite the fact that the convexified set may not intersect with the global space, the method appears to perform well empirically (see Section 7).

5.1. Local space and projection

Let us define the local space for triangle i, denoted by $\mathfrak{B}_i \subset \mathbb{C}^2$, with frame ζ^i (a complex number with unit length). The following inequalities define the space:

$$\mathrm{Re}\left(\frac{f_z^i}{\zeta^i}\right) - \left|f_{\bar{z}}^i\right| \geq \sigma_2 \tag{7}$$

$$\mathrm{Re}\left(\frac{f_z^i}{\zeta^i}\right) \geq \frac{1}{k}\left|f_{\bar{z}}^i\right| \tag{8}$$

where σ_2 is a small positive constant and k is slightly less than 1. As the notation suggests, the expressions above derive from the expressions for the little dilatation [WMZ12] and the smaller singular value of the Jacobian in terms of $|f_z|$ and $|f_{\bar{z}}|$. One simply replaces $|f_z|$ with the component of f_z in the direction of the frame. As this is less than $|f_z|$, we obtain injectivity guarantees and rough alignment with the frames. A schematic and a description of $\mathfrak{B}_i$ is given in Figure 5 and its caption. The full convex space is the product space $\mathfrak{B} = \prod_i \mathfrak{B}_i$.

As can be seen in Figure 5, projection to the boundary of $\mathfrak{B}_i$ in one component $(f_z^i, f_{\bar{z}}^i)$ is reduced simply to projection in a 2-dimensional plane. In each of the colored region, the projection to the solid black curve is described by a simple formula, so we omit it for brevity. Note that only $\mathrm{Re}\left(f_z^i/\zeta^i\right)$ and $\left|f_{\bar{z}}^i\right|$ are changed, while $\mathrm{Im}\left(f_z^i/\zeta^i\right)$ and $\arg f_{\bar{z}}^i$ remain unchanged. Projection to $\mathfrak{B}$ is achieved by component-wise projection to each of the $\mathfrak{B}_i$.

5.2. Global space and projection

The global space is defined as the image of a linear map. With the selected inversion algorithm, we may collect the extracted vertex positions at and adjacent to cones and boundaries for basis elements of $\mathcal{N}(A_{hgp})$ as columns of a matrix H. This matrix has dimensions $2(|V_{cb}|) \times 2(|\partial V| + |C| + n - 1)$ where V_{cb} denotes the vertices of T_{cb} in S_c. We compose this with another linear map J which calculates the Jacobian over triangles of T_{cb} and decomposes them into similarity and antisimilarity parts: f_z and $f_{\bar{z}}$ (this matrix is essentially D from Equation (4), restricted to V_{cb} and up to a change of basis). It is a simple matrix to derive, but if additional background is desired, we refer the reader to Section 3 of [CCW16a]. Let us use $\tilde{J} := J \circ H$ to denote the composition which takes an element of $\mathcal{N}(A_{hgp})$ and produces the resulting f_z^i and $f_{\bar{z}}^i$ for all T_{cb}.

Lemma 10 *$\tilde{J} : \mathbb{R}^{2(|\partial V|+|C|+n-1)} \to \mathbb{R}^{4|T_{cb}|}$ has a null space of dimension 2, consisting of constant complex vectors (x- and y- components constant but perhaps not equal).*

The proof is delayed to Appendix C, but it's intuitively clear that the elements of $\mathcal{N}(\tilde{J})$ consist of elements that co-locate all vertices dictated by the interpolation constraints, so that Jacobians vanish. It is sometimes useful to eliminate this null space and make our global projections simpler, so we add one more complex row of all ones. We denote this matrix $\tilde{J}_1$.

As noted, the image of $\tilde{J}$ defines the linear subspace that we perform global projections toward. The first step of the algorithm is a simple global projection, without any restriction to a normal hyperplane. For this, we look for an area-weighted least-squares solution to $\tilde{J}c = p$, where p is our initial starting point. The point $\tilde{J}c$ will be the result of global projection. This amounts to finding a solution to the following minimization problem $\min_c \left\|W^{1/2}(\tilde{J}c - p)\right\|^2$, where W denotes a diagonal matrix containing areas of the triangles in T_{cb}. A simple matrix derivative shows that any least-squares solution satisfies an area-weighted normal equation:

$$\tilde{J}^t W \tilde{J} c = \tilde{J}^t W p.$$

However, $\tilde{J}^t W \tilde{J}$ is not invertible, with a 2-dimensional space of solutions c. As we are only after their image $\tilde{J}c$, we run through the same analysis looking for a least squares solution to $\tilde{J}_1 c = \binom{p}{0}$,

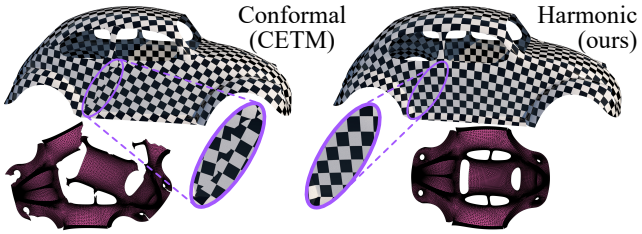

Figure 6: *A comparison on a multiply-connected car mesh of our method with CETM [SSP08]. Conformal mapping of such annular domains is difficult and fails to keep the map continuous, while ours succeeds due to its use of spatial variables.*

and then extract all but the last two rows of $\tilde{J}_1 c$. The unique solution here is determined by the equation:

$$c = (\tilde{J}_1^t W_1 \tilde{J}_1)^{-1} \tilde{J}_1^t W_1 p, \tag{9}$$

where W_1 has its last two diagonal entries given a weight equal to the average of the areas.

For subsequent global projections, we have a map that needs to be restricted to the hyperplane $H_l := \{q \in \mathbb{R}^{4|T_{cb}|} \mid (a_l - b_l)^t (b_l - q) = 0\}$, where a_l and b_l denote the results of the most recent global and local projections, respectively (mirroring notation from [HCW17]). For brevity, we skip the minimization problem formulation and derivation and give the formula directly:

$$c_{l+1} = c_l - \frac{n_l^t W n_l}{\begin{pmatrix} n_l \\ 0 \end{pmatrix}^t W_1 \tilde{J}_1 \tilde{M}^{-1} \tilde{J}_1^t W_1 \begin{pmatrix} n_l \\ 0 \end{pmatrix}} \tilde{M}^{-1} \tilde{J}_1^t W_1 \begin{pmatrix} n_l \\ 0 \end{pmatrix} \tag{10}$$

where c_l denotes the previous least squares solution, $\tilde{M} := \tilde{J}_1^t W_1 \tilde{J}_1$, and $n_l := a_l - b_l$, again mimicking notation from [HCW17]. The derivation is similar to the one performed there for Equation 20. In the same fashion, we may use just one linear solve $\tilde{M}^{-1} \tilde{J}_1^t W_1 \begin{pmatrix} n_l \\ 0 \end{pmatrix}$ and precompute $\tilde{M}^{-1} \tilde{J}_1^t W_1$.

With both global and local steps described, the method can be applied. The frames given by [BZK09] determine the starting f_z^i, while the starting $f_{\bar{z}}^i$ are set to 0. A concise outline of the ATP method with relevant parameters is given in Section 7.1.

5.3. Boundary Case with Trivial Holonomy

The above ATP method was aimed at finding coordinates c for a locally injective map in our subspace. In the disc and punctured disc case with trivial holonomy (e.g., Figure 6), this may be done simply, with guarantees. We use the original result of [GGT06] and place one boundary along the boundary of a convex set, and ask that the other boundary vertices lie in the convex hull of their neighbors (with the existing cotangent weights). This is a small dense linear system in the variables corresponding to V_{cb}, especially if virtual vertices are being used (Section 4.2). The result is guaranteed to be a locally injective member of our subspace.

6. Projected Newton Distortion Minimization

The last third of our method takes a locally injective map in our subspace and uses a projected Newton method [TSIF05] to optimize for an area-weighted symmetric Dirichlet energy $E_{iso} = 0.5(\sigma_1^2 + \sigma_1^{-2} + \sigma_2^2 + \sigma_2^{-2})$ over T_{cb}. A line search ensures that local injectivity, and cone and turning angles are kept valid throughout. As hinted at by Observation 2, minimizing this energy for triangles in T_{cb} tends to minimize the energy over all triangles in S. This can be seen empirically in Figure 7.

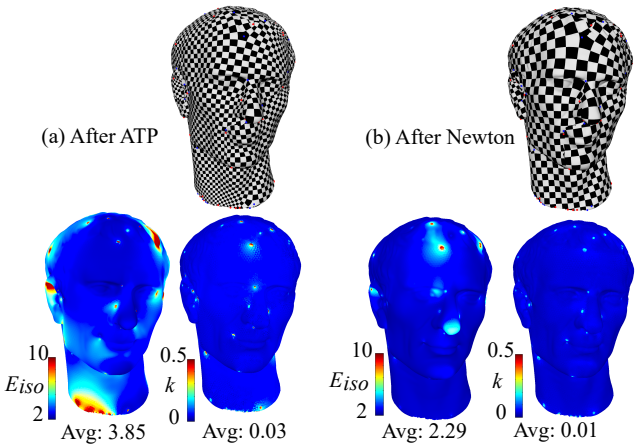

Figure 7: *Example of Projected Newton (PN) Optimization. On the left is our textured result without PN optimization, and the right is our textured result after. The color maps show the symmetric Dirichlet energy E_{iso} and the conformal distortion k. Note the marked improvement and distortion being maximized near the cones and boundaries.*

Note that this energy is non-convex, so the Hessian will not be positive semi-definite (PSD) and a naive Newton method will not work. One needs to first project the Hessian into the PSD cone, and here we note that the Hessian is the sum of the per-triangle Hessians, and if we can project these individually to PSD Hessians, their sum will be PSD also. In other settings, the per-element Hessians are small (say 3×3 of 6×6), but in our setup, we have the entries of c as variables, and not the vertex positions, so our per-triangle Hessians are $2(|\partial V| + |C| + n - 1)$-dimensional. To efficiently make these Hessians PSD, we follow the route of [CW15] and note that these per triangle Hessians are just $(\tilde{J}_i)^t Q \tilde{J}_i$ where $\tilde{J}_i$ are the four rows of $\tilde{J}$ giving the components of f_z^i and $f_{\bar{z}}^i$, and Q is the Hessian of E_{iso} in terms of these components. An analytic formula for projection of Q is present in [CW15].

Once the Hessian projection determines direction of travel, two line searches are performed: one ensures preservation of local injectivity while another ensures decreasing energy. Again this is described in detail in [CW15].

7. Results

We begin with an overview of our method, and then present some results of experiments.

7.1. Algorithm Overview and Parameters

Our algorithm consists of three parts: a subspace construction (Section 4), a search for a locally injective map within it (Section 5),

and a Projected Newton (PN) distortion minimization step (Section 6). Our code is available in a documented public repository at https://github.com/eden-fed/HGP.

For the subspace construction, we leverage Theorem 8 and Proposition 7 to determine $\mathcal{N}(A_{hgp})$, solutions to our HGP system. The KKT system, Equation (5), may be solved for choices of c_{int} as standard bases vectors of $\mathbb{R}^{2(|\partial V|+|C|+n-1)}$. The resulting $\mathbf{z}$'s form a basis and may be extracted as partial columns of K^{-1}. Moreover, Theorem 9 from [BCW17] allows us to restrict ourselves to extracting only z_i for vertices $v_i \in V_{cb}$. In our experiments, we used the selected inversion algorithm available in the PARDISO suite [Sch]. The documentation and interface for selecting elements is technically challenging, so a brief explanation is provided in Appendix D. Finally, for dense meshes with boundary, virtual vertices may be used to even further reduce dimensionality of our subspace.

For a locally injective map within $\mathcal{N}(A_{hgp})$ in a general scenario, we use an ATP method from [HCW17] operating in the space of per-triangle Jacobians (expressed as complex derivatives f_z^i and $f_{\bar{z}}^i$). The space of locally injective derivatives is convexified using frames taken from a frame field generated by [BZK09]. This choice of frames encourages satisfaction of the cone and turning angle requirements of Theorem 9. These frames also provide the initializing value for the f_z^i, and $f_{\bar{z}}^i = 0$ at the start. This starting point p is projected first globally (Equation (9)), and then alternates between local (described at end of Section 5.1) and global tangential projections (Equation (10)). For our experiments, the local projection step is governed by bounds with $k = 0.9$ and $\sigma_2 = 0.01$, and projection terminates when $\left\|W^{1/2} n_l\right\| < 10^{-4}$. Finally, in the special case of boundaries and trivial holonomy, we use the classical result of [GGT06], solving a dense linear system to guarantee a locally injective map, as explained in Section 5.3.

For the PN distortion minimization step, a brief description was already given in the previous section. To obtain the final map, once we have a distortion-optimized c_{int}^*, we use the factorization (from the selected inversion algorithm) to solve the KKT system Equation (5) for our final map $\mathbf{z}^*$, avoiding full construction of any basis elements of $\mathcal{N}(A_{hgp})$.

7.2. Experiments

For the following experiments, our algorithm was implemented as a plugin to Autodesk Maya 2018 and executed on a Windows 10 machine with Xeon E5-1650 3.60 GHz CPU, 32GB RAM and Nvidia Quadro K6000 graphics processor.

The subspace construction was implemented in C++ utilizing the selective inverse implementation of PARDISO 6.0, while the ATP and PN solvers were implemented in Matlab and were invoked from C++ through Matlab's engine. We note that in [HCW17] and [CW17], the ATP and PN procedures were implemented in CUDA and run autonomously on the GPU. Since in our case the cost of these two nonlinear optimizations is relatively small compared to that of the subspace construction (which is also very fast), we opted for simplicity and implemented the solvers in Matlab. Further speedup can be obtained by pursing the full GPU implementations [HCW17, CW17]. Our Matlab implementation of the ATP solver is purely CPU based. Unlike [CW17], we solve the dense linear system that arises in each Newton iteration on the CPU, and only the Hessian construction is partially GPU-accelerated by utilizing Matlab's gpuArray functionality.

The method was tested on 77 models, split into two groups: those with cones and those without. Group A consists of 61 spheres and disks, mostly from the benchmark dataset of [MPZ14] equipped with prescription of cone singularities. There is only one addition of the large (250K triangles) "awakening" model with 509 cones, for stress testing. Group B is an assortment of 16 genus 0 models with one or more boundaries, i.e. multiply-connected disk domains, without cones.

Our algorithm succeeded in producing locally injective parametrizations on 66 of these models. The resulting parametrizations are available for download on the authors' websites. Four of these are illustrated in Figure 1. 9 had 2–3 localized foldovers due to negative cotangent weights corresponding to badly-shaped source mesh triangles. These were all fixed with a simple heuristic that repositions a single vertex to the kernel of its 1-ring polygon.

Table 1 provides the timing of different steps of our algorithm for a selection of models from group A (including "awakening"), as well as a comparison of the total runtime to that of HGP and the metric-based approach of [CLW16]. Our method is an order of magnitude faster than HGP, and 2–4 orders of magnitude faster than [CLW16]. Figure 2 shows that it produces at least comparable and often higher quality maps compared to HGP, as HGP does not optimize the map distortion.

Further timing comparisons to composite majorization (CM) [SPSH*17] on selected group B models are in Table 2. Here, our method also possesses a significant speedup, and can process large models with millions of triangles in several seconds.

Figure 6 compares our harmonic parametrization of a group B Beetle model with a conformal map obtained using [SSP08]. The conformal map fails to produce a continuous parametrization since the obtained conformally equivalent flat metric cannot be immersed in the plane without first cutting the mesh into a simply-connected domain. The use of spatial variables ensures by construction the continuity of our low distortion harmonic parametrization.

Finally, Figure 7 demonstrates the efficacy of the projected Newton (PN) step. As can be seen, both E_{iso} and k are decreased, and the distortion tends to bunch around cones and boundaries.

7.3. Robustness

For the following discussion, Laplacian weights are assumed to be positive. This sufficient condition is often not necessary, and even in cases where foldovers do occur, they are easy to fix in practice (as explained above). Besides numerical issues, the only step in the algorithm with failure potential is ATP. The subspace construction is provably robust, and the PN solver is guaranteed to preserve local injectivity if initialized with such a map, due to the use of the flip-preventing line search. Models from group B (without cones) do not use ATP for feasible initialization, instead relying on [GGT06]. This guarantees an initial feasible map, hence, our algorithm is theoretically fully robust in this case. Experimentally, our algorithm successfully parametrized all 16 models in group B.

Model name	#Triangles	#Boundary vertcies	#DOF	#Newton iters	Subspace construction time (sec)	Newtom time (sec)	Total algorithm time (sec)	#CM Iters	CM time (sec)
aircraft	4,656	40	40	4	0.03	0.03	0.06	45	0.32
beetle_refined	38,726	1,136	138	18	0.31	0.15	0.47	34	2.79
gargoyle	98,803	439	439	9	0.56	0.21	0.86	34	8.47
max_plank	100,000	222	222	15	0.37	0.16	0.53	23	5.83
bear	296,409	557	557	19	2.17	0.91	3.08	26	20.07
buddha	470,507	1,033	137	23	1.75	0.17	1.92	26	48.14
mask_650k	644,952	3,390	339	6	5.38	0.22	5.62	12	19.43
mannequin_mc_900k	847,959	687	687	4	5.29	0.47	5.78	17	58.78
mannequin-devil_2m	1,815,057	929	929	6	12.31	1.25	13.59	22	185.17
nicolo_da_uzzano_2m	1,933,918	3,438	394	14	16.51	0.50	17.05	12	101.17
julius_2.5m	2,510,341	2,529	253	7	15.21	0.13	15.34	12	172.83

Table 2: *Time comparison to CM [SPSH*17] on discs and multiply-connected discs without cones. Our method is significantly faster.*

The 11 failure cases were all from group A. 2 failures were due to inaccurate results from PARDISO. The remaining 9 were due to lack of ATP convergence, which only occurs when the convex space it is trying to converge to is empty [HCW17]. Thus, in these cases $\mathcal{N}(A_{hgp}) \cap \mathfrak{B} = \emptyset$ and the convexified space cut out by the frames does not intersect our harmonic space of maps.

For models with cones, our method is thus less robust than HGP which employs a series of helpful heuristics that we didn't use, such as: adjusting the Laplacian weights, updating the frames upon failure in an iterative manner, and homotopy frame fixing. HGP also softens the harmonicity constraints, satisfying them in the least squares manner. This can be interpreted as searching within a slightly larger space of biharmonic maps which further decreases the chance for infeasibility. We argue that in the smooth case, the space of harmonic maps is large enough to always contain a feasible solution of a locally injective map with prescribed flat cone metric. This is because for genus 0 surfaces, there exists such a conformal map [BGB08] which is in particular harmonic. Hence, we speculate that adjusting the frames might be sufficient for obtaining increased robustness.

8. Conclusion

Our method provides a fast method for computing harmonic seamless global parametrizations via solutions of the HGP system. A full analysis of this system was performed and an efficient method for a low-dimensional subspace construction was developed, using a selected inversion algorithm. An ATP method is utilized to quickly find a locally injective mapping within this subspace, and the end result is optimized for distortion with a Projected Newton method. Ultimately, we produce a method which is comparable to nonlinear methods, but has nearly the same cost as linear methods (on the order of 1–2 linear solves).

For future work, we would like to extend our method to surfaces with high genus. Another obvious direction for further investigation, is the development of better methods for frame determination, and other strategies to increase the robustness of the method when applied to models with cones. Additionally, experiments with multiply-connected disc models suggests that the additional flexibility of harmonic maps (as opposed to conformal maps) will allow for quick low-distortion mapping of such domains without tearing.

Finally, our methods also offer a possibility of allowing for positional constraints, something that may also be hard to handle in the conformal setting. This may be useful for a quadrangulation, in which cone positions are required to be integral.

Acknowledgments

This research was partially funded by the Israel Science Foundation (grants No. 1869/15 and 2102/15). We thank: Alec Jacobson, for useful discussions on null spaces of sparse matrices; David Bommes, for discussions on the structure of our linear system; Zohar Levi, for the Beetle CETM result of Figure 6; and Olaf Schenk, Fabio Verbosio, Renjie Chen, Ladislav Kavan, Petr Kadleček, Roi Poranne, and Alon Bright, for assisting with PARDISO utilization.

References

[AL15] AIGERMAN N., LIPMAN Y.: Orbifold Tutte embeddings. *ACM Transactions on Graphics 34*, 6 (2015), 190. 2, 3

[APL14] AIGERMAN N., PORANNE R., LIPMAN Y.: Lifted bijections for low distortion surface mappings. *ACM Transactions on Graphics 33*, 4 (2014), 69. 4

[APL15] AIGERMAN N., PORANNE R., LIPMAN Y.: Seamless surface mappings. *ACM Transactions on Graphics 34*, 4 (2015), 72. 4

[ApS17] APS M.: *The MOSEK optimization software*, 2017. URL: http://www.mosek.com/. 2

[BCW17] BRIGHT A., CHIEN E., WEBER O.: Harmonic Global Parametrization with Rational Holonomy. *ACM Trans. Graph. 36*, 4 (July 2017), 89:1–89:15. 1, 2, 3, 4, 5, 6, 8, 9, 11, 15

[BGB08] BEN-CHEN M., GOTSMAN C., BUNIN G.: Conformal Flattening by Curvature Prescription and Metric Scaling. *Computer Graphics Forum 27*, 2 (Apr. 2008), 449–458. 2, 3, 4, 12

[BLP*13] BOMMES D., LÉVY B., PIETRONI N., PUPPO E., SILVA C., TARINI M., ZORIN D.: Quad-mesh generation and processing: A survey. *Computer Graphics Forum 32*, 6 (2013), 51–76. 3

[BS07] BOBENKO A. I., SPRINGBORN B. A.: A Discrete Laplace-Beltrami Operator for Simplicial Surfaces. *Discrete & Computational Geometry 38*, 4 (Dec. 2007), 740–756. 7

[BV04] BOYD S., VANDENBERGHE L.: *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. 7

[BZK09] BOMMES D., ZIMMER H., KOBBELT L.: Mixed-integer Quadrangulation. In *ACM SIGGRAPH 2009 Papers* (New York, NY, USA, 2009), SIGGRAPH '09, ACM, pp. 77:1–77:10. 4, 9, 10, 11

[CBSS17] CLAICI S., BESSMELTSEV M., SCHAEFER S., SOLOMON J.: Isometry-aware preconditioning for mesh parameterization. In *Computer Graphics Forum* (2017), vol. 36, Wiley Online Library, pp. 37–47. 4

[CCW16a] CHIEN E., CHEN R., WEBER O.: Bounded Distortion Harmonic Shape Interpolation. *ACM Trans. Graph. 35*, 4 (July 2016), 105:1–105:15. 3, 4, 9

[CCW16b] CHIEN E., CHEN R., WEBER O.: Bounded distortion harmonic shape interpolation. *ACM Transactions on Graphics 35*, 4 (2016), Article 105, 15 pages. 9

[CLW16] CHIEN E., LEVI Z., WEBER O.: Bounded distortion parametrization in the space of metrics. *ACM Transactions on Graphics 35*, 6 (2016), Article 215, 16 pages. 2, 4, 11

[CW15] CHEN R., WEBER O.: Bounded distortion harmonic mappings in the plane. *ACM Transactions on Graphics 34*, 4 (2015), Article 73, 12 pages. 3, 10

[CW17] CHEN R., WEBER O.: GPU-accelerated Locally Injective Shape Deformation. *ACM Trans. Graph. 36*, 6 (Nov. 2017), 214:1–214:13. 1, 3, 4, 11

[DVPSH15] DIAMANTI O., VAXMAN A., PANOZZO D., SORKINE-HORNUNG O.: Integrable polyvector fields. *ACM Transactions on Graphics 34*, 4 (July 2015), 38:1–38:12. 4

[FH05] FLOATER M. S., HORMANN K.: Surface Parameterization: a Tutorial and Survey. *Advances In Multiresolution For Geometric Modelling* (2005). 3

[FL16] FU X.-M., LIU Y.: Computing inversion-free mappings by simplex assembly. *ACM Transactions on Graphics 35*, 6 (2016), 216. 4

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

[Flo97] FLOATER M. S.: Parametrization and smooth approximation of surface triangulations* 1. *Computer Aided Geometric Design 14*, 3 (1997), 231–250. 2, 3

[Flo03] FLOATER M. S.: Mean value coordinates. *Computer Aided Geometric Design 20*, 1 (2003), 19–27. 5

[FSSB07] FISHER M., SPRINGBORN B., SCHRÖDER P., BOBENKO A. I.: An algorithm for the construction of intrinsic delaunay triangulations with applications to digital geometry processing. *Computing 81*, 2-3 (2007), 199–213. 5

[GGT06] GORTLER S. J., GOTSMAN C., THURSTON D.: Discrete one-forms on meshes and applications to 3d mesh parameterization. *Computer Aided Geometric Design 23*, 2 (Feb. 2006), 83–112. 2, 3, 10, 11, 15

[GSC18] GOLLA B., SEIDEL H.-P., CHEN R.: Piecewise linear mapping optimization based on the complex view. *Computer Graphics Forum 37*, 7 (2018), 233–243. 4

[HCW17] HEFETZ E. F., CHIEN E., WEBER O.: Fast Planar Harmonic Deformations with Alternating Tangential Projections. *Computer Graphics Forum 36*, 5 (2017), 175–188. 1, 3, 8, 10, 11, 12

[HDA17] HERHOLZ P., DAVIS T. A., ALEXA M.: Localized solutions of sparse linear systems for geometry processing. *ACM Trans. Graph. 36*, 6 (Nov. 2017), 183:1–183:8. 5

[HLS07] HORMANN K., LÉVY B., SHEFFER A.: Mesh parameterization: Theory and practice. *SIGGRAPH Course Notes* (2007). 3

[J*18] JACOBSON A., ET AL.: gptoolbox: Geometry processing toolbox, 2018. http://github.com/alecjacobson/gptoolbox. 3

[JLY17] JACQUELIN M., LIN L., YANG C.: Pselinv - A distributed memory parallel algorithm for selected inversion: The symmetric case. *ACM Transactions on Mathematical Software (TOMS) 43*, 3 (2017), 21. 5

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

[KLS13] KUZMIN A., LUISIER M., SCHENK O.: Fast methods for computing selected elements of the green's function in massively parallel nanoelectronic device simulations. In *Euro-Par 2013 Parallel Processing* (Berlin, Heidelberg, 2013), Wolf F., Mohr B., an Mey D., (Eds.), Springer Berlin Heidelberg, pp. 533–544. 3, 5, 8

[KSS06] KHAREVYCH L., SPRINGBORN B., SCHRÖDER P.: Discrete conformal mappings via circle patterns. *ACM Transactions on Graphics 25*, 2 (2006), 412–438. 3

[LFJG17] LIU S., FERGUSON Z., JACOBSON A., GINGOLD Y.: Seamless: Seam erasure and seam-aware decoupling of shape from mesh resolution. *ACM Trans. Graph. 36*, 6 (Nov. 2017), 216:1–216:15. 8

[Lip12] LIPMAN Y.: Bounded Distortion Mapping Spaces for Triangular Meshes. *ACM Trans. Graph. 31*, 4 (July 2012), 108:1–108:13. 2, 3, 4, 9

[LPRM02] LÉVY B., PETITJEAN S., RAY N., MAILLOT J.: Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics 21*, 3 (2002), 362–371. 2

[LW16] LEVI Z., WEBER O.: On the convexity and feasibility of the bounded distortion harmonic mapping problem. *ACM Transactions on Graphics 35*, 4 (2016), Article 106, 15 pages. 3

[LYM*11] LIN L., YANG C., MEZA J. C., LU J., YING L., ET AL.: Selinv—an algorithm for selected inversion of a sparse symmetric matrix. *ACM Transactions on Mathematical Software (TOMS) 37*, 4 (2011), 40. 5

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

[MPZ14] MYLES A., PIETRONI N., ZORIN D.: Robust field-aligned global parametrization. *ACM Transactions on Graphics 33*, 4 (July 2014), 135:1–135:14. 4, 11

[MZ12] MYLES A., ZORIN D.: Global Parametrization by Incremental Flattening. *ACM Trans. Graph. 31*, 4 (July 2012), 109:1–109:11. 4, 5

[MZ13] MYLES A., ZORIN D.: Controlled-distortion constrained global parametrization. *ACM Transactions on Graphics 32*, 4 (July 2013), 105:1–105:14. 4

[RPPSH17] RABINOVICH M., PORANNE R., PANOZZO D., SORKINE-HORNUNG O.: Scalable locally injective mappings. *ACM Transactions on Graphics 36*, 2 (2017). 4

[SC17] SAWHNEY R., CRANE K.: Boundary first flattening. *ACM Transactions on Graphics 37*, 1 (Dec. 2017), 5:1–5:14. 2, 3

[Sch] SCHENK O.: PARDISO. URL: https://www.pardiso-project.org/. 3, 5, 11, 15

[SdS01] SHEFFER A., DE STURLER E.: Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers 17*, 3 (2001), 326–337. 3

[SLMB05] SHEFFER A., LÉVY B., MOGILNITSKY M., BOGOMYAKOV A.: ABF++: fast and robust angle based flattening. *ACM Trans. Graph. 24*, 2 (2005), 311–330. 3

[SPSH*17] SHTENGEL A., PORANNE R., SORKINE-HORNUNG O., KOVALSKY S. Z., LIPMAN Y.: Geometric Optimization via Composite Majorization. *ACM Trans. Graph. 36*, 4 (July 2017), 38:1–38:11. 4, 11, 12

[SS15] SMITH J., SCHAEFER S.: Bijective parameterization with free boundaries. *ACM Transactions on Graphics 34*, 4 (2015). 4

[SSP08] SPRINGBORN B., SCHRÖDER P., PINKALL U.: Conformal equivalence of triangle meshes. *ACM Transactions on Graphics 27*, 3 (2008), 77. 3, 10, 11

[TACSD06] TONG Y., ALLIEZ P., COHEN-STEINER D., DESBRUN M.: Designing quadrangulations with discrete harmonic forms. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, Switzerland, 2006), SGP '06, Eurographics Association, pp. 201–210. 3

[Tak73] TAKAHASHI K.: Formation of sparse bus impedance matrix and its application to short circuit study. In *Proc. PICA Conference, June, 1973* (1973). 4

[TSIF05] TERAN J., SIFAKIS E., IRVING G., FEDKIW R.: Robust quasistatic finite elements and flesh simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer Animation* (2005), ACM, pp. 181–190. 10

[Tut63] TUTTE W. T.: How to Draw a Graph. *Proceedings of the London Mathematical Society s3-13*, 1 (Jan. 1963), 743–767. 2, 3

[VCD*16] VAXMAN A., CAMPEN M., DIAMANTI O., PANOZZO D., BOMMES D., HILDEBRANDT K., BEN-CHEN M.: Directional field synthesis, design, and processing. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 545–572. 3

[VCKS17] VERBOSIO F., CONINCK A. D., KOUROUNIS D., SCHENK O.: Enhancing the scalability of selected inversion factorization algorithms in genomic prediction. *Journal of Computational Science 22* (2017), 99 – 108. 3, 5, 8

[WMZ12] WEBER O., MYLES A., ZORIN D.: Computing extremal quasiconformal maps. *Computer Graphics Forum 31*, 5 (2012), 1679–1689. **Best Paper Award**. 9

[WZ14] WEBER O., ZORIN D.: Locally injective parametrization with arbitrary fixed boundaries. *ACM Transactions on Graphics 33*, 4 (2014), Article 75, 12 pages. 3

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

[ZLS07] ZAYER R., LÉVY B., SEIDEL H.-P.: Linear Angle Based Parameterization. In *Fifth Eurographics Symposium on Geometry Processing - SGP 2007* (Barcelone, Spain, July 2007), Alexander Belyaev M. G., (Ed.), Eurographics Association, pp. 135–141. 2

Appendix A: Proof of Lemma 5

Proof Consider a cell decomposition of S: G_s and ∂S serve as the 1-skeleton with the interior of S_c giving a single 2-cell. The number of vertices is $|\partial V| + |V_s| - |\partial V \cap V_s|$, while the number of edges is $|\partial E| + |E_s|$. To determine $|\partial V \cap V_s|$, consider each component of G_s at a time. For the first component, it will intersect k_1 boundaries, and may only intersect each boundary once, producing k_1 elements of $\partial V \cap V_s$. If there were multiple intersections with a single boundary, cutting along it would produce a disconnected result. Now if we cut along the component, we obtain a genus 0 mesh with $n - k_1 + 1$ boundary components, with a new boundary consisting of duplicate edges from the component and edges from the k_1 components it intersected. Take next a component of G_s that touches the new boundary, which intersects k_2 of the original boundaries. This component may not intersect the first, and may only touch each boundary once, producing $1 + k_2$ elements of $\partial V \cap V_s$. This process iterates until all components have been considered, counting all elements of $\partial V \cap V_s$. We find that:

$$|\partial V \cap V_s| = k_1 + 1 + k_2 + \cdots + 1 + k_m = n + (m - 1),$$

where the second equality follows, as each new component considered may not intersect boundary components that have already been touched, lest the result of cutting be disconnected. □

Appendix B: Full Rank of A_{lin}

Proposition 11 A_{lin} is full rank.

Proof It suffices to show that the rows of $A_{lin} = \begin{pmatrix} A_1 \\ A_{int} \end{pmatrix}$ are linearly independent. Let $\{a_k\}_{k=1}^{|E_s|}$ denote the rows of A_1, corresponding to the rotation constraints associated with each seam edge $e_k \in E_s$ (and its duplicates in S_c). Let $\{a_l\}_{l=1}^{|\partial V|+|C|+n-1}$ denote the rows of A_{int}, corresponding to the interpolation conditions specified in Section 3.1. These are associated with vertices $v_l \in C \cup \partial V$ (or one or two of their duplicates). We consider a linear combination of these rows a_k and a_l that equal 0, and argue that it must be the trivial linear combination.

Recall that A_{lin} and its rows are very sparse, and if a column of A_{lin} contains only one non-zero entry in a row a_k or a_l, then the coefficient in front of a_k or a_l must be 0. Similarly, if a column has only a few non-zero entries, and if the coefficients for all but one of the rows, a_k or a_l, is 0, then the coefficient in front of a_k or a_l must also disappear. A chain of such deductions will show disappearance of all row coefficients. We will argue separately for the sphere and disc (potentially punctured) cases, but there are two important common facts.

Fact 1 For two seam edges e_k and e_{k+1} sharing a non-cone non-boundary seam vertex v of degree 2, the vanishing of the coefficient for one of a_k or a_{k+1} implies the vanishing of the other. This follows as the columns of A_{lin} corresponding to either duplicate copy of v in S_c only have two non-zero entries: in rows a_k and a_{k+1}. Chaining together seam edges meeting at such vertices v into *meta-edges*, we see that a vanishing coefficient in front of any component edge, implies vanishing coefficients for all others in a meta-edge.

Fact 2 For a cone vertex v_l and incident seam edges $\{e_k, e_{k+1}, \ldots\}$: if any coefficient for an incident a_k disappears, then the coefficients in front of a_l and all other incident $\{a_{k+1}, a_{k+2}, \ldots\}$ must also disappear. When v_l is degree 1 in G_s, this is clear as the column corresponding to v_l contains only two nonzero entries: in rows a_l and a_k. In the higher degree case, we first consider columns corresponding to v_l duplicates not associated with a_l (not having position specified). These contain only two nonzero entries, in rows corresponding to two particular incident edges. If incident relation a_k has vanishing coefficient, it must be in at least one such pair of rows for a v_l duplicate column, and the coefficient in front of the other row must vanish. Following these columns, we may "cycle" around v_l and deduce that coefficients for all incident $\{a_k, a_{k+1}, \ldots\}$ must disappear. Finally, this implies that the coefficient of a_l must vanish by inspection of the column corresponding to the copy of v_l with position specified.

Using the above two facts, we may traverse ∂S and G_s, seeing that the coefficients of all relations must disappear. For the sphere case, in Section 3.1.1, we left one cone vertex c of degree 1 without its position specified. Thus, the coefficient for incident a_k must disappear, by inspection of the column corresponding to c. By Fact 1, the coefficients for an entire meta-edge disappear until we reach a cone vertex v_l. By the second fact, the coefficient for a_l vanishes along with the coefficients for all adjacent seam edge rows $\{a_k, a_{k+1}, \ldots\}$. These are part of outgoing meta-edges, and the pattern continues until the entire tree of G_s (Lemma 3) is covered. Thus, all coefficients disappear.

In the case with boundaries, vertices $v_l \in \partial V \backslash V_s$ have corresponding columns with only one nonzero entry: in row a_l, so these must have vanishing coefficient. Thus, we are again simply looking for traversal of G_s and its m components. Unlike in the sphere case,

there are not necessarily any cones to start from. Recall from Section 3.1.2 that $n-1$ vertices of $\partial V \cap V_s$ have *two* of the positions of their duplicate copies specified, and that each component of G_s was left with at least one vertex having only one position specification. For such a vertex v_l, all coefficients for incident edge relations and for a_l must disappear (by reasoning similar to the argument for Fact 2). As in the sphere case, each of the m components may now be traversed, and we see that all coefficients must vanish. □

Appendix C: Proof of Lemma 10

Proof As J calculates gradients, $\mathcal{N}(J)$ is $2r$-dimensional where r is the number of connected components of T_{cb}. Any element of $\mathcal{N}(J)$ co-locates all vertices in any such component to single points (potentially differing across components). Thus, $\tilde{J}(c) = 0$ iff Hc is in this null space. Recall that c specifies a solution to the HGP system, and any such solution implies a pair of discrete harmonic 1-forms on a 4-fold branched cover (see Section 5 of [BCW17]). Arguments similar to those in Appendix B of [GGT06] imply that any pair of such 1-forms inducing faces of zero area or edges of zero length must vanish entirely. If $Hc \in \mathcal{N}(J)$, then it has such elements, so the map implied by Hc must be constant and co-locate all r components to the same point. □

Appendix D: PARDISO Selected Inversion Usage

We used PARDISO 6.0 [Sch] for the subspace construction. We set *mtype* to -2 (real and symmetric indefinite) for the matrix from Equation (5), and executed the following phases in PARDISO:

1. Phase 1: Fill-reduction analysis and symbolic factorization
2. Phase 2: Numerical factorization
3. Phase -22: Selected Inversion
4. Phase -1: Termination and Memory Release Phase

For the reported run-times we used the following flags, with the last two recommended for highly indefinite symmetric matrices:

1. iparm(24)=0 : Parallel Numerical Factorization
2. iparm(28)=1 : Parallel reordering METIS
3. iparm(11)=1 : Scaling vectors
4. iparm(13)=1 : Improved accuracy using weighted matchings

A matrix A is transferred to PARDISO in compressed sparse row (CSR) format. If the matrix is symmetric, as in our case, only the upper triangular part is transferred. A usage example of the CSR format can be found in Figure 2 of the PARDISO manual. Note that PARDISO expects diagonal elements to be present for symmetric matrices, even if they are zero, e.g. element $(6,6)$ in the example is explicitly set to 0.0.

The selected inversion algorithm calculates by construction every entry of the inverse matrix A^{-1} which corresponds to a non-zero entry in the factor L (lower triangle matrix) of the LDL^T factorization of A. In our setting, we need to calculate additional entries of A^{-1} which correspond to zero elements in the factor. This is achieved by "selecting" the additional elements for calculation. Potentially, this can be done by explicitly adding numerical 0.0 at corresponding entries in the factor L in the same fashion zero diagonal elements are handled. Unfortunately, PARDISO does not provide a user interface for altering L. To overcome this (purely technical) limitation, we utilize the fact that entries which are present in A persist in L, and thus we set the numerical zeros in A.

For example, consider again the matrix from Figure 2 in the manual and assume we would like to calculate the additional entry $A^{-1}(3,6)$. To this end, we alter the nonzero structure of A by adding a single element with **0.0** value. The corresponding CSR representation is:

	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*
IA:	1	5	8	11	13	16	18	19	20										
JA:	1	3	6	7	2	3	5	3	6	8	4	7	5	6	7	6	8	7	8
A:	7.	1.	2.	7.	-4.	8.	2.	1.	**0.**	5.	7.	9.	5.	-1.	5.	0.	5.	11.	5.

The corresponding inverse matrix elements are returned "in place", and the selected elements can be easily extracted.

We note that changing the sparsity pattern of L indirectly through A is suboptimal as it leads to denser than necessary L. Thus, we hope that future versions of PARDISO will provide a direct means for selecting elements for calculation. This will immediately provide additional speedup for the subspace construction.

Appendix E: Virtual Vertex Degeneracies

Figure 8 shows two examples of degeneracies that can occur when meshes which are not 3-connected are used in conjunction with wrong choice of virtual vertices. A "diagonal" is a vertex pair such that its removal would disconnect the mesh. We spot diagonals by iterating over the boundary vertices ∂V. For each $v \in \partial V$, we look for a neighbor vertex $N(v)$ such that $N(v)$ is on the boundary but the edge $(v, N(v))$ is not. If such $N(v)$ is found, we add v and $N(v)$, as well as an additional arbitrary vertex along the boundary between them to the virtual vertices subset. This ensures that a locally injective harmonic map within the reduced space defined by the virtual vertices exists.

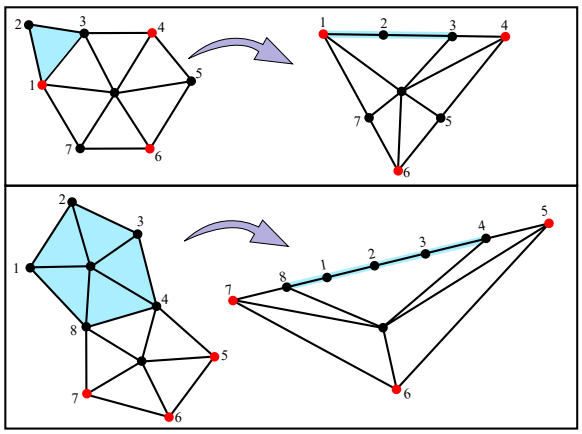

Figure 8: *Wrong choice of virtual vertices (marked in red). Left: source meshes where the cyan region is degenerated under the map on the right. Top:* $(1,3)$ *is a diagonal. Bottom:* $(4,8)$ *is a diagonal.*