Efficient and Robust Discrete Conformal Equivalence with Boundary

MARCEL CAMPEN, Osnabrück University, Germany
RYAN CAPOUELLEZ, HANXIAO SHEN, LEYI ZHU, DANIELE PANOZZO, DENIS ZORIN, NYU, USA

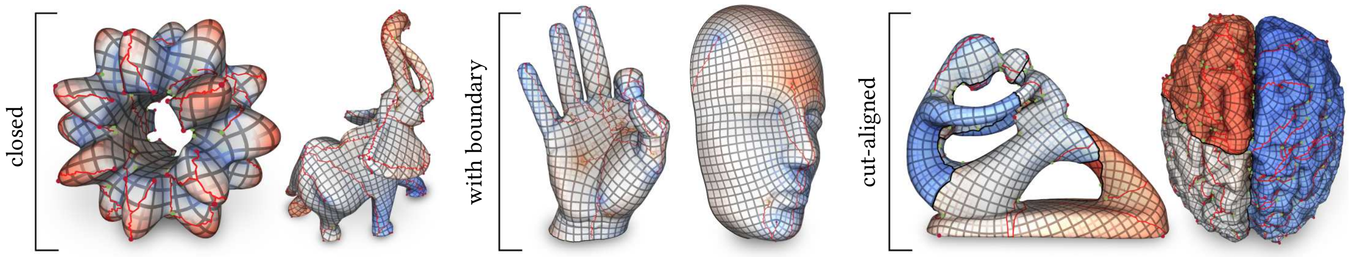


Fig. 1. Given target discrete curvature per vertex on a triangle mesh, we describe a method to efficiently and robustly compute a discrete conformal deformation of the mesh's metric to satisfy this prescription. Its basis is the mathematical foundation described in [Gu et al. 2018b; Springborn 2020]. The figure illustrates several flattenings (parametrizations over the plane) obtained from computed metrics that are flat except at a few cone singularities (green and red points). The parametrization is visualized by means of an adaptive grid texture (see Section 7); red paths indicate transitions in the parametrization due to the prescribed cones. Conformal scale distortion is indicated by shading, from blue over white to red. The method supports closed surfaces as well as surfaces with boundary. By prescribing the geodesic curvature along the boundary, alignment of the parametrization with the boundary can be enforced. On the right, closed surfaces are cut (along the black curves), turning them into surfaces with boundary, which enables us to enforce parametrization alignment also along the cut curves.

We describe an efficient algorithm to compute a discrete metric with prescribed Gaussian curvature at all interior vertices and prescribed geodesic curvature along the boundary of a mesh. The metric is (discretely) conformally equivalent to the input metric. Its construction is based on theory developed in [Gu et al. 2018b] and [Springborn 2020], relying on results on hyperbolic ideal Delaunay triangulations. Generality is achieved by considering the surface's intrinsic triangulation as a degree of freedom, and particular attention is paid to the proper treatment of surface boundaries. While via a double cover approach the case with boundary can be reduced to the case without boundary quite naturally, the implied symmetry of the setting causes additional challenges related to stable Delaunay-critical configurations that we address explicitly. We furthermore explore the numerical limits of the approach and derive continuous maps from the discrete metrics.

CCS Concepts: • **Computing methodologies** → **Computer graphics**; **Mesh models**; **Mesh geometry models**; *Shape modeling*.

Additional Key Words and Phrases: intrinsic Delaunay, intrinsic triangulation, edge flip, conformal parametrization, conformal map, cone metric

ACM Reference Format:
Marcel Campen and Ryan Capouellez, Hanxiao Shen, Leyi Zhu, Daniele Panozzo, Denis Zorin. 2021. Efficient and Robust Discrete Conformal Equivalence with Boundary. *ACM Trans. Graph.* 40, 6, Article 1 (December 2021), 16 pages. https://doi.org/10.1145/3478513.3480557

Marcel Campen, Institute for Computer Science, Osnabrück University, Germany. Ryan Capouellez, Hanxiao Shen, Leyi Zhu, Daniele Panozzo, Denis Zorin, Courant Institute of Mathematical Sciences, New York University, USA.

0730-0301/2021/12-ART1
https://doi.org/10.1145/3478513.3480557

1 INTRODUCTION

Computing discrete metrics with prescribed angles on meshes is a problem closely related to surface parametrization and quadrangulation, which is of interest in many geometric settings. Despite many years of efforts, only a few techniques for mesh parametrization provide theoretical guarantees, commonly derived from the same source: discrete harmonic mappings with convex boundary, based on Tutte's embedding theorem [Floater 1997].

Recent exciting advances concerning the theory of discrete metric uniformization [Gu et al. 2018b; Springborn 2020] provide a solid foundation for a much needed addition to this spectrum of methods. They enable the computation of discrete metrics with arbitrary prescribed discrete curvature at vertices, as long as the discrete Gauss-Bonnet theorem is respected. In particular, this allows to compute, with guarantees, flat metrics or almost-everywhere flat cone metrics with prescribed curvatures at cones—an essential component of global parametrization and quadrangulation algorithms. Guarantees follow from a reduction to an unconstrained convex optimization problem. However, compared to Tutte's method, the numerics involved are far more complex, in particular due to nonlinearity and large scale distortions inherent in conformal maps.

We present an efficient numerical algorithm based on these new theoretical ideas, extend it to support surfaces with boundary, and explore its practical performance, focusing on robustness.

Problem Summary. To define the problem more precisely, consider a manifold triangle mesh M, possibly with boundary. For a given discrete metric on M, i.e., an assignment of lengths to its edges that satisfy the triangle inequality, we can compute inner angles of triangles.

Let Θ_i be the total angle (the sum of incident inner angles) at vertex v_i, and κ_i its angle deficit, defined as $2\pi - \Theta_i$ for interior vertices and $\pi - \Theta_i$ for boundary vertices. This quantity κ_i can be viewed as the discrete Gaussian curvature if v_i is an interior vertex and the geodesic curvature of the boundary if v_i is on the boundary. Given *target* curvatures $\hat{\kappa}_i$ (respecting the discrete Gauss-Bonnet theorem) our goal is to compute edge lengths that exhibit exactly these curvatures. Flattenings, i.e., mesh parametrizations over the plane, are a special case corresponding to prescribing $\kappa_i = 0$ in the interior [Ben-Chen et al. 2008]. Seamless maps for quadrilateral remeshing are obtained by prescribing $\hat{\kappa}_i = k_i \frac{\pi}{2}$ with $k_i \in \mathbb{Z}$ [Campen et al. 2019; Myles and Zorin 2012].

Approach. As shown in [Gu et al. 2018b; Springborn 2020], a discrete metric realizing target curvatures $\hat{\kappa}_i$ always exists, *if re-triangulation of the surface is allowed.* When restricting to metrics *discretely conformally equivalent* to a given original metric, this metric is unique (up to scale) and can be computed by minimizing a convex function.

While the latter property has been exploited before for practical parametrization purposes [Springborn et al. 2008], the assumption of a fixed triangulation restricts the space of target curvatures that can be realized by a conformally equivalent metric. For example, a vertex v_i of valence k cannot, under any (Euclidean) metric, exhibit a discrete curvature $\kappa_i \leq (2 - k)\pi$, because inner angles are bounded by π. As a consequence, the resulting discrete metric's edge lengths violate the triangle inequality in some places. This limitation can be remedied by allowing changes to the triangulation of the input surface.

More concretely, the main requirement for triangulation changes needed to enable this is that at all times the triangulation remains an intrinsic Delaunay triangulation. This leads to a natural algorithm [Sun et al. 2015] in the spirit of kinetic data structures [Basch et al. 1999], which, however, requires the determination of the exact sequence of all individual Delaunay-critical events during the metric computation process.

Contributions. In this paper we describe an efficient and practical algorithm, performing triangulation changes with greater flexibility, enabled by the theoretical connection to hyperbolic metrics established by [Gu et al. 2018b; Springborn 2020]. While this theory is developed for closed surfaces, in practice many, if not most, relevant applications involve surfaces with boundaries. These cases can be reduced to the closed surface case by creating a surface double, but a number of algorithmic issues need to be addressed to reliably maintain symmetric intrinsic Delaunay triangulations in such cases. We introduce a number of additional improvements to the basic algorithm, to speed up convergence and increase accuracy and robustness. We furthermore perform extensive evaluations, with a focus on numerical aspects such as the effect of varying arithmetic accuracy. Numerical behavior of the algorithm is of critical relevance as conformal metrics and maps can unavoidably exhibit very large ranges of scales.

We discuss the relevant background in Section 3. An implementation of the main ideas, with particular attention to practical aspects is described in Section 4. Generalization to surfaces with boundary is presented in Section 5, followed by the construction of a surface mapping from the discrete metric in Section 6, and concluded by the evaluation of the algorithm in Section 7.

2 RELATED WORK

The problem of computing conformally equivalent metrics or, by implication, conformal maps of discrete surfaces, has been considered in a variety of works before. As there is no useful natural notion of conformality in the discrete (non-smooth) setting, a range of discrete counterparts of the continuous concept of conformality have been proposed and used.

Static Triangulation. Prominent examples of works addressing the computation of conformal metrics or conformal maps on discrete surfaces, based on various definitions of discrete conformality, while considering the triangulation fixed are based on least-squares formulations [Desbrun et al. 2002; Lévy et al. 2002], vertex scaling formulations [Ben-Chen et al. 2008; Jin et al. 2007; Sawhney and Crane 2017; Soliman et al. 2018; Springborn et al. 2008], circle patterns [Kharevych et al. 2006], or formulations based on holomorphic one-forms [Gu and Yau 2003].

Dynamic Triangulation. A fixed triangulation restricts the space of metrics that can be achieved. By adjusting the triangulation depending on the prescribed target curvature, this limitation can be remedied. Two systematic approaches have been proposed to that end, both conceptually considering a continuous metric evolution from initial state to target state. [Luo 2004] proposes to adjust the triangulation by an intrinsic edge flip whenever an edge becomes triangle inequality critical (Figure 2 left). Implementation variants are described in [Campen et al. 2019; Campen and Zorin 2017a,b]. Differently, [Gu et al. 2018a,b; Springborn 2020] effectively consider the case of flipping an edge when it becomes Delaunay-critical, i.e., when four vertices become co-circular (Figure 2 right). Surfaces with boundary in this context are addressed in [Sun et al. 2015] using a double cover approach, reducing this case to the case without boundary. A correspondence map between the original triangulation and the modified triangulation can be kept track of by means of an overlay data structure [Fisher et al. 2007].

In concurrent work, [Gillespie et al. 2021] make use of the same theoretical results we use here and describe an algorithm that conceptually is very close to our core algorithm in Section 4. Main differences of our work are (i) a number of important details in the optimization procedure as described in Section 4, (ii) special combinatorial handling of symmetry in the double surface used to support meshes with boundary, and (iii) extensive evaluation in particular of numerical limits and numerical precision effects. In comparison, [Gillespie et al. 2021] propose a more lightweight data structure (than [Fisher et al. 2007] that we use) to keep track of the mesh overlay, and additionally consider the case of spherical parametrization.

3 BACKGROUND

We begin by considering the case of surfaces *without* boundary, i.e., we are given a closed manifold triangle mesh $M = (V, E, F)$. The case of surfaces with boundary can be reduced to the closed surface case with an additional symmetry structure, which we address in detail in Section 5.

The mesh M is equipped with an input metric defined by edge lengths $\boldsymbol{\ell} : E \to \mathbb{R}^{>0}$, satisfying the triangle inequality.

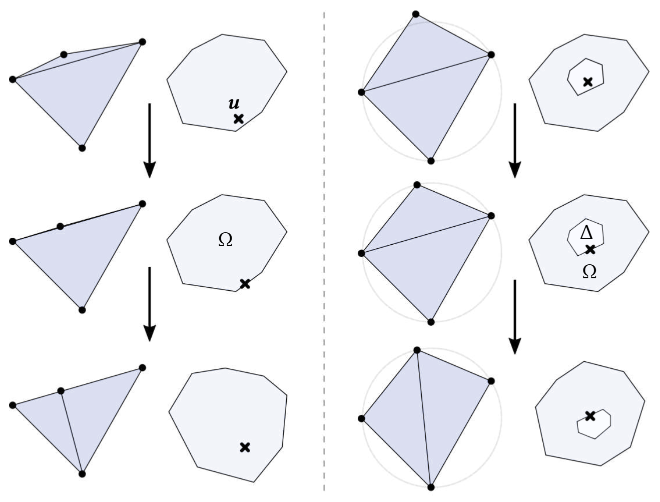


Fig. 2. Left: flip-on-degeneration. Right: flip-on-Delaunay-violation. Alongside a conceptual illustration of the valid region Ω (light blue) and Delaunay region Δ (white) is shown (cf. Section 3.2), containing the current point $\boldsymbol{u}$ (cross mark) and changing due to the flip.

3.1 Conformal Equivalence

A *conformally equivalent* discrete metric, for a fixed triangulation M, is defined by means of *logarithmic scale factors* $\boldsymbol{u} : V \longrightarrow \mathbb{R}$ associated with vertices $V = (v_1, \dots, v_n)$, by defining new edge lengths as

$$\ell_{ij}(\boldsymbol{u}) = \ell_{ij}\, e^{\frac{u_i+u_j}{2}} \tag{1}$$

per edge e_{ij} [Luo 2004]. Given per-vertex target angles $\hat{\Theta}_i$ a conformally equivalent metric with these angles is characterized by, for all i:

$$g_i(\boldsymbol{u}) := \hat{\Theta}_i - \Theta_i(\boldsymbol{u}) = \hat{\Theta}_i - \sum_{T_{ijk}} \alpha^i_{jk}(\boldsymbol{u}) = 0, \tag{2}$$

where the inner angle $\alpha^i_{jk}(\boldsymbol{u})$ is computed under the metric defined by $\boldsymbol{u}$ via Eq. (1), i.e., from edge lengths $\tilde{\boldsymbol{\ell}} = \boldsymbol{\ell}(\boldsymbol{u})$. We use shorthands $\tilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha}(\boldsymbol{u})$ and $\tilde{\boldsymbol{\ell}} = \boldsymbol{\ell}(\boldsymbol{u})$ for these scale factor dependent quantities in the following.

It is known that $\boldsymbol{g}(\boldsymbol{u}) = (g_1(\boldsymbol{u}), \dots, g_n(\boldsymbol{u}))$ is the gradient of a twice-differentiable convex function [Springborn et al. 2008]. Hence, one may obtain factors $\boldsymbol{u}$ satisfying Eq. (2) using (second-order) convex optimization methods, starting from arbitrary initializations (e.g. $\boldsymbol{u} \equiv \boldsymbol{0}$). This is true, however, only as long as there is a solution for which $\boldsymbol{u}$ stays in the feasible region $\Omega_M \subset \mathbb{R}^n$ where $\boldsymbol{\ell}(\boldsymbol{u})$ respects the triangle inequality for each triangle T_{ijk}; otherwise it does not define a Euclidean surface metric on M.

3.2 Dynamic Triangulation

The feasible region Ω_M can be altered by adjusting the triangulation dynamically *during* the evolution of $\boldsymbol{u}$ from $\boldsymbol{0}$ towards $\boldsymbol{u}^*$.

Note that a change of triangulation is possible without *intrinsically* changing the surface. M together with given edge lengths defines a surface S_V with a metric which is flat everywhere except at V. There are many triangulations (besides M) with vertices V and their own associated edge lengths, defining the same surface S_V (cf. [Sharp et al. 2019]); hence the differentiation between M and S_V. In particular, an edge flip replacing a pair of triangles (T_{ijk}, T_{jim}) sharing an edge e_{ij}, with triangles (T_{kim}, T_{mjk}) sharing edge e_{km} can be performed without intrinsically changing the surface S_V, by setting the length of the new edge e_{km} to the length of the diagonal of the planar quadrilateral obtained by unfolding T_{ijk}, T_{jim} [Fisher et al. 2007]. This is referred to as *intrinsic flip.*

Delaunay Flips. [Gu et al. 2018b; Springborn 2020] prove a remarkable fact: the convex energy can be extended to all of the space $\mathbb{R}^n$ of scale factors $\boldsymbol{u}$ defined at vertices, *if a particular change of triangulation is allowed.* Specifically, the triangulation is modified so that it stays (intrinsically) Delaunay at all times as $\boldsymbol{u}$ evolves. More specifically, whenever the Delaunay condition is violated as a result of a change in $\boldsymbol{u}$, a flip is performed to maintain the Delaunay property. As the resulting energy is a globally convex function, it can be minimized by an unconstrained Newton method, and the resulting choice of $\boldsymbol{u}$ satisfies (2) with respect to the resulting triangulation.

Definition 1 (Intrinsic Delaunay). *A triangulation is intrinsically Delaunay if the angles of two triangles T_{ijk} and T_{jim} opposite a common edge e_{ij} satisfy the Delaunay condition:*

$$\tilde{\alpha}^k_{ij} + \tilde{\alpha}^m_{ij} \le \pi \tag{3}$$

or equivalently $\cos\tilde{\alpha}^k_{ij} + \cos\tilde{\alpha}^m_{ij} \ge 0$. *Expressed directly in terms of edge lengths this condition is equivalent to*

$$\frac{\tilde{\ell}^2_{jk} + \tilde{\ell}^2_{ki} - \tilde{\ell}^2_{ij}}{\tilde{\ell}_{jk}\tilde{\ell}_{ki}} + \frac{\tilde{\ell}^2_{jm} + \tilde{\ell}^2_{mi} - \tilde{\ell}^2_{ij}}{\tilde{\ell}_{jm}\tilde{\ell}_{mi}} \ge 0. \tag{4}$$

This latter version of the Delaunay condition is particularly important for our construction.

Generically (iff these weak inequalities hold strictly), the intrinsic Delaunay triangulation is unique, but for special configurations (four or more intrinsically co-circular vertices resulting in equality in Eq. (4)) it is not.

For a given triangulation M, the *Penner cell* $\Delta_M \subset \mathbb{R}^n$ denotes the set of scale factors $\boldsymbol{u}$ for which M, along with the modified metric defined by $\boldsymbol{u}$, is intrinsic Delaunay. Clearly, $\Delta_M \subset \Omega_M$, and when $\boldsymbol{u} \in \partial\Delta_M$ the Delaunay triangulation is not unique. Whenever $\boldsymbol{u}$ reaches the boundary of Δ_M, we can switch to another Delaunay triangulation M' by means of an intrinsic flip, thereby changing the region (from Δ_M to $\Delta_{M'}$), enabling $\boldsymbol{u}$ to evolve further, see Figure 2. The cells Δ_M form a partition of $\mathbb{R}^n$ [Gu et al. 2018b].

Such changes of scale factors together with intrinsic Delaunay flips lead to the following generalized notion of discrete conformal equivalence of two metrics [Gu et al. 2018b]:

Definition 2 (Discrete Conformal Equivalence). *Two metrics $(M_1, \boldsymbol{\ell}_1)$ and $(M_m, \boldsymbol{\ell}_m)$ are* discretely conformally equivalent, *if there is a sequence of meshes with the same vertex set, $(M_s, \boldsymbol{\ell}_s)$, $s = 1, \dots, m$, such that, for each s, M_s is an intrinsic Delaunay triangulation for the metric $\boldsymbol{\ell}_s$ and either*

- *$(M_s, \boldsymbol{\ell}_s)$ and $(M_{s+1}, \boldsymbol{\ell}_{s+1})$ are different metrics with the same triangulation (i.e., $M_s = M_{s+1}$) and the edge lengths are related by Eq.* (1) *for a choice of* $\boldsymbol{u}_s : V \longrightarrow \mathbb{R}$.
- *$(M_s, \boldsymbol{\ell}_s)$ and $(M_{s+1}, \boldsymbol{\ell}_{s+1})$ are different Delaunay triangulations for the same metric.*

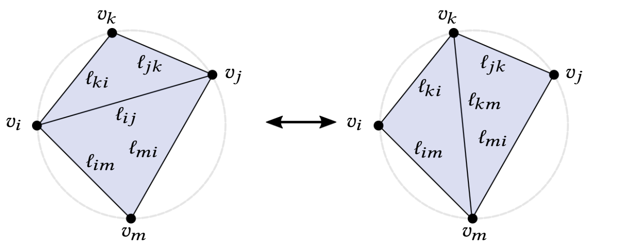


Fig. 3. Ptolemy flip of an edge e_{ij} shared by two triangles forming an inscribed quadrilateral, i.e., a Delaunay-critical edge.

Degeneration Flips. The alternative of performing a triangulation change only when $\boldsymbol{u}$ reaches the boundary $\partial\Omega$ of the currently feasible region was considered by [Luo 2004]. This occurs when a triangle becomes a degenerate cap. An intrinsic flip of this triangle's longest edge yields a non-degenerate triangulation, effectively changing the valid region Ω such that $\boldsymbol{u}$ lies strictly in its interior. Figure 2 left illustrates this. An implementation of this approach is described and applied in [Campen and Zorin 2017b]. Open theoretical questions remain regarding the finiteness of the flip sequence and the sound handling of simultaneous adjacent degeneracies.

At first sight, the approach based on maintaining an intrinsic Delaunay triangulation may seem less efficient in comparison. Due to $\Delta \subset \Omega$, at least as many, but often many more cells Δ need to be traversed. Practically, this suggests a large number of small steps between flips in the process of optimizing $\boldsymbol{u}$, compared to, e.g., the use of (less frequent) degeneration flips, and much smaller steps compared to typical unconstrained optimization.

Remarkably, however, this Delaunay approach permits an implementation that is in general more efficient and more robust (see Section 7.2 for a comparison). As we will see, exploiting a relation to *hyperbolic* Delaunay triangulation, arbitrarily large steps can be made, beyond Δ and even beyond Ω (unconstrained by Euclidean triangle inequalities). Flips can be performed collectively *after the fact* and in *arbitrary order*. This is detailed in Section 3.4.

3.3 Evolution Step

Assume we are given a triangulation M that is intrinsic Delaunay under the metric defined by some $\boldsymbol{u}_{\vdash}$. Consider an evolution of $\boldsymbol{u}$ from $\boldsymbol{u}_{\vdash}$ to $\boldsymbol{u}_{\dashv}$, e.g., linear:

$$\boldsymbol{u}(t) = (1-t)\boldsymbol{u}_{\vdash} + t\boldsymbol{u}_{\dashv},\ t \in [0,1].$$

As we move along the interval $[0,1]$, whenever four vertices forming triangles T_{ijk} and T_{jim} become co-circular under the metric defined by $\boldsymbol{\ell}(\boldsymbol{u}(t))$, an intrinsic flip of edge e_{ij} is performed. Due to the special configuration (the two triangles forming an *inscribed* quadrilateral, see Figure 3) the length that the new edge e_{km} needs to take can be computed following Ptolemy's theorem as

$$\tilde{\ell}_{km} = \frac{1}{\tilde{\ell}_{ij}}(\tilde{\ell}_{jk}\tilde{\ell}_{im} + \tilde{\ell}_{ki}\tilde{\ell}_{mj}), \tag{5}$$

where we use $\tilde{\boldsymbol{\ell}}$ as a shorthand for $\boldsymbol{\ell}(\boldsymbol{u}(t))$. For $\tilde{\ell}_{km} = \ell_{km}(u(t)) = \ell_{km}\, e^{\frac{u_k+u_m}{2}}$ to take this value for the current $\boldsymbol{u}(t)$, we need to set:

$$\ell_{km} := \frac{1}{\ell_{ij}}(\ell_{jk}\ell_{im} + \ell_{ki}\ell_{mj}). \tag{6}$$

Notice that this is Ptolemy's formula, Eq. (5), applied to the *original* metric, as all scale factors cancel. In other words: applying the formula in the current ($\boldsymbol{u}(t)$-scaled) metric $\tilde{\boldsymbol{\ell}}$ is equivalent to applying it in the original metric $\boldsymbol{\ell}$, followed by scaling. Remarkably, this holds *even though the vertices are not co-circular under the original metric* in general. Moreover, the edge lengths $\boldsymbol{\ell}$ set in this way may *not even satisfy the triangle inequality*. This is no issue, though, as certainly the relevant scaled lengths $\tilde{\boldsymbol{\ell}} = \boldsymbol{\ell}(\boldsymbol{u}(t))$ do, by construction.

It was shown that the number of flip events along the path is finite [Wu 2014], which means that after a finite number of flips we will obtain the triangulation and edge length assignment needed for the target $\boldsymbol{u}(1) = \boldsymbol{u}_{\dashv}$.

One practical downside of this procedure, in which the necessary flips along the evolution path are detected and performed one-by-one sequentially [Sun et al. 2015], is that it requires solving precisely for the sequence of flips. An alternative approach, whose correctness can be shown based on an interpretation of the involved edge lengths as defining hyperbolic metrics instead of Euclidean metrics, improves on this.

3.4 Hyperbolic Metric Approach

Instead of moving t along the interval $[0,1]$, determining the sequence of flip events and executing them in order, let us directly consider $t = 1$. The initial triangulation M may not be Delaunay under $\boldsymbol{u}(1)$, and the edge lengths $\boldsymbol{\ell}(\boldsymbol{u}(1))$ may not even respect the triangle inequality. Nevertheless, we can test each edge for violation of the Delaunay criterion using Eq. (4) applied to $\boldsymbol{\ell}(\boldsymbol{u}(1))$, and incrementally flip (using Eq. (6)) all violating edges in arbitrary order following the classical flip algorithm until a Delaunay triangulation is reached [Bobenko and Springborn 2007]. While in case of triangle inequality violations this criterion lacks the geometric justification via Eq. (3) (the involved quantities are no longer (cotangents of) Euclidean angles), this algorithm nevertheless succeeds.

Hyperbolic Delaunay. The reasons for applicability of Eq. (4) and use of Eq. (6) are direct consequences of an elegant correspondence between hyperbolic and conformal metric structures used in the proofs of [Gu et al. 2018b; Springborn 2020] and introduced in [Rivin 1994], given by mapping edge lengths to Penner coordinates of a hyperbolic metric, and Euclidean triangulations to ideal triangulations. Detailed explanations can be found in these papers and an overview given in [Crane 2020, §5, §6]. We go into more detail in Section 6, as this relation is important when the conformal metric is used to establish a conformal map, for purposes of evaluation of the map at arbitrary points. Here we just present a proposition summarizing the aspect of this theory relevant to our algorithm.

Proposition 1. *Suppose lengths $\tilde{\boldsymbol{\ell}}$ (possibly not satisfying triangle inequality) are assigned to edges in a triangle mesh M, conformally equivalent to a set of Euclidean metric lengths $\boldsymbol{\ell}$. If the flip algorithm is applied to $\tilde{\boldsymbol{\ell}}$, with the Delaunay criterion in algebraic form* (4) *used to determine which edges need to be flipped, and the Ptolemy formula* (6) *used for length updates, the algorithm produces a triangulation M' with lengths $\tilde{\boldsymbol{\ell}}'$ that satisfy the triangle inequality. This triangulation is intrinsic Delaunay. Moreover, the discrete metric defined by $(M', \tilde{\boldsymbol{\ell}}')$ is discrete conformally equivalent to $(M, \boldsymbol{\ell})$.*

In summary, instead of performing flips following an expensive-to-compute sequence required to maintain a valid Euclidean metric on triangles at all times, the algorithm performs the flips in arbitrary order, yielding edge lengths $\tilde{\boldsymbol{\ell}}$ satisfying the triangle inequality only in the end. This version of the flip algorithm is referred to as *Weeks algorithm* [Weeks 1993].

These observations ensure that whenever we modify scale factors $\boldsymbol{u}$ while computing the conformal metric, the flip algorithm can be used to recover a Delaunay triangulation, which can then be used to evaluate the value of the convex function we need to minimize, its gradient, and its Hessian.

4 ALGORITHM

Using this background, we can now formulate an efficient algorithm for the computation of a conformally equivalent metric, respecting prescribed target angles $\hat{\Theta}$. The algorithm, spelled out in Algorithm 1, is based on a standard Newton's method with line search, but incorporates several important details and modifications.

Delaunay. Initially, if M is not already intrinsically Delaunay, it is turned into a Delaunay mesh using standard intrinsic edge flips. Then, whenever $\boldsymbol{u}$ is updated (during the line search), before the gradient and Hessian are evaluated the triangulation is turned into an intrinsic Delaunay triangulation with respect to the metric defined by $\boldsymbol{u}$ using Weeks flip algorithm—now using the Ptolemy length computation rule from Eq. (6).

Energy-free Line Search. The function $E(\boldsymbol{u})$ that needs to be minimized is known explicitly [Springborn et al. 2008]:

$$E(\boldsymbol{u}) = \sum_{T_{ijk}} \left(2f(\tilde{\lambda}_{ij}, \tilde{\lambda}_{jk}, \tilde{\lambda}_{ki}) - \pi(u_i + u_j + u_k)\right) + \hat{\Theta}^\top \boldsymbol{u},$$

where $\tilde{\lambda}_{ij} = 2\log \ell_{ij} + u_i + u_j$ and f is a per-triangle function involving Milnor's Lobachevsky function [Springborn et al. 2008, Eq. (8)]. The gradient of $E(\boldsymbol{u})$ is $\boldsymbol{g}(\boldsymbol{u}) = \hat{\Theta} - \Theta(\boldsymbol{u})$ (Eq. (2)) and its Hessian $H(\boldsymbol{u})$ simply is the well-known positive semi-definite cotan-Laplacian in terms of the scaled angles $\boldsymbol{\alpha}(\boldsymbol{u})$.

The obvious approach is to use the standard Newton's method with backtracking line search, using $E(\boldsymbol{u})$, $\boldsymbol{g}(\boldsymbol{u})$, $H(\boldsymbol{u})$ (cf. [Gillespie et al. 2021]). However, computing the energy $E(\boldsymbol{u})$, in particular evaluating the Lobachevsky function, presents numerical challenges, and efficient Chebyshev-polynomial approximations, like the one used in the implementation of [Springborn et al. 2008], may not yield sufficient accuracy, while incurring additional computational

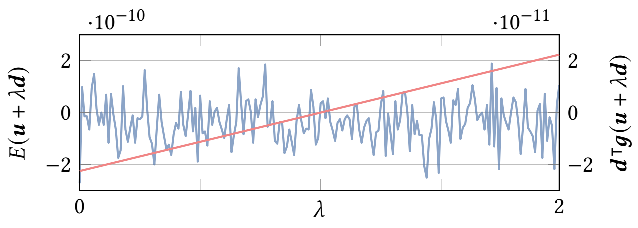


Fig. 4. Energy (blue; mean (20202.12) subtracted) and projected gradient (red) along a descent direction $\boldsymbol{d}$. Notice that the numerical noise in the energy computation dominates the actual change in energy, making it less suitable to be a measure of progress in the line search. By contrast, the sign of the projected gradient (red) can be determined much more precisely.

Algorithm 1: FindConformalMetric

```
Input  : triangle mesh $M = (V, E, F)$, closed, manifold,
         edge lengths $\boldsymbol{\ell} > 0$ satisfying triangle inequality,
         target angles $\hat{\Theta} > 0$ respecting Gauss-Bonnet
Output : triangle mesh $M' = (V, E', F')$,
         edge lengths $\tilde{\boldsymbol{\ell}} > 0$ satisfying triangle inequality,
         such that $\|\Theta_{(M',\tilde{\boldsymbol{\ell}})} - \hat{\Theta}\|_\infty \le \varepsilon_{\text{tol}}$

Function FindConformalMetric($M, \boldsymbol{\ell}, \hat{\Theta}$):
    $\boldsymbol{u} \leftarrow \boldsymbol{0}$
    $(M, \boldsymbol{\ell}) \leftarrow$ MakeDelaunay($M, \boldsymbol{\ell}, \boldsymbol{u}$)
    while not converged($M, \boldsymbol{\ell}, \boldsymbol{u}$) do
        $\boldsymbol{g} \leftarrow g(M, \boldsymbol{\ell}, \boldsymbol{u})$                      // gradient
        $H \leftarrow H(M, \boldsymbol{\ell}, \boldsymbol{u})$                      // Hessian
        $\boldsymbol{d} \leftarrow solve(H\boldsymbol{d} = -\boldsymbol{g})$              // Newton direction
        $(M, \boldsymbol{\ell}, \boldsymbol{u}) \leftarrow$ LineSearch($M, \boldsymbol{\ell}, \boldsymbol{u}, \boldsymbol{d}$)   // Newton step
    $\tilde{\boldsymbol{\ell}} \leftarrow$ScaleConformally($M, \boldsymbol{\ell}, \boldsymbol{u}$)
    return $(M, \tilde{\boldsymbol{\ell}})$

Function LineSearch($M, \boldsymbol{\ell}, \boldsymbol{u}, \boldsymbol{d}$):
    $(M_1, \boldsymbol{\ell}_1) \leftarrow$ MakeDelaunayPtolemy($M, \boldsymbol{\ell}, \boldsymbol{u} + \boldsymbol{d}$)
    $(M_{1/2}, \boldsymbol{\ell}_{1/2}) \leftarrow$ MakeDelaunayPtolemy($M, \boldsymbol{\ell}, \boldsymbol{u} + \frac{1}{2}\boldsymbol{d}$)
    if $\frac{1}{2}\left(\boldsymbol{d}^\top g(M_1, \boldsymbol{\ell}_1, \boldsymbol{u}{+}\boldsymbol{d}) + \boldsymbol{d}^\top g(M_{1/2}, \boldsymbol{\ell}_{1/2}, \boldsymbol{u}{+}\frac{1}{2}\boldsymbol{d})\right) \le$
     $\alpha\boldsymbol{d}^\top g(M, \boldsymbol{\ell}, \boldsymbol{u})$ then          // Armijo-like condition
        return $(M_1, \boldsymbol{\ell}_1, \boldsymbol{u} + \boldsymbol{d})$                // full step
    while true do                                   // line search
        $(M, \boldsymbol{\ell}) \leftarrow$ MakeDelaunayPtolemy($M, \boldsymbol{\ell}, \boldsymbol{u} + \boldsymbol{d}$)
        if $\boldsymbol{d}^\top g(M, \boldsymbol{\ell}, \boldsymbol{u} + \boldsymbol{d}) \le 0$ then              // Eq. (7)
            return $(M, \boldsymbol{\ell}, \boldsymbol{u} + \boldsymbol{d})$
        $\boldsymbol{d} \leftarrow \frac{1}{2}\boldsymbol{d}$                               // backtracking

Function converged($M, \boldsymbol{\ell}, \boldsymbol{u}$):
    return $\|\hat{\Theta} - \Theta(M, \tilde{\boldsymbol{\ell}})\|_\infty \le \varepsilon_{\text{tol}}$

Function $g$ ($M, \boldsymbol{\ell}, \boldsymbol{u}$):
    return $\hat{\Theta} - \Theta(M, \tilde{\boldsymbol{\ell}})$                         // Eq. (2)

Function $H$ ($M, \boldsymbol{\ell}, \boldsymbol{u}$):
    return CotanLaplacian($M, \tilde{\boldsymbol{\ell}}$)

Function $\Theta(M, \boldsymbol{\ell}, \boldsymbol{u})$:                         // angle computation
    for $v_i \in V$ do                                   // Eq. (2)
        $\Theta_i \leftarrow \sum_{T_{ijk} \in M'} \arccos\left((\tilde{\ell}_{ij}^2 + \tilde{\ell}_{ki}^2 - \tilde{\ell}_{jk}^2)/(2\tilde{\ell}_{ij}\tilde{\ell}_{ki})\right)$
    return $(\Theta_0, \ldots, \Theta_n)$

Function MakeDelaunayPtolemy($M, \boldsymbol{\ell}, \boldsymbol{u}$):
    while NonDelaunay($M, \boldsymbol{\ell}, \boldsymbol{u}, e_{ij}$) for any edge $e_{ij}$ do
        $(M, \boldsymbol{\ell}) \leftarrow$ PtolemyFlip($M, \boldsymbol{\ell}, e_{ij}$)
    return $(M, \boldsymbol{\ell})$

Function NonDelaunay($M, \boldsymbol{\ell}, \boldsymbol{u}, e_{ij}$):
    return $(\tilde{\ell}_{jk}^2 + \tilde{\ell}_{ki}^2 - \tilde{\ell}_{ij}^2)/(\tilde{\ell}_{jk}\tilde{\ell}_{ki})$
        $+ (\tilde{\ell}_{jm}^2 + \tilde{\ell}_{mi}^2 - \tilde{\ell}_{ij}^2)/(\tilde{\ell}_{jm}\tilde{\ell}_{mi}) < 0$        // Eq. (4)

Function PtolemyFlip($M, \boldsymbol{\ell}, e_{ij}$):
    $M \leftarrow$ Flip($M, e_{ij}$)
    $\ell_{km} \leftarrow (\ell_{jk}\ell_{im} + \ell_{ki}\ell_{mj})/\ell_{ij}$                 // Eq. (6)
    return $(M, \boldsymbol{\ell})$
```

overhead. We observe that the energy can be very flat along the search direction, so using the decrease of energy evaluated this way as a criterion in the line search may lead to the algorithm stalling due to numerical noise (see Figure 4). This is particularly problematic in cases requiring high conformal distortion or if we want to compute the conformal metric with high precision, as needed for instance to derive an implied conformal map (cf. Section 7).

The evaluation of the gradient and Hessian, both of which are simple functions of angles (not involving the Lobachevsky function), by contrast, is more efficient and numerically robust than the energy itself (see Figure 4). Fortunately, we are able to formulate our algorithm such that it relies on $\boldsymbol{g}(\boldsymbol{u})$ and $H(\boldsymbol{u})$ only. This is possible for the following reason: As $E(\boldsymbol{u})$ is convex, it is also convex along the search direction $\boldsymbol{d}$, i.e., $E(\boldsymbol{u} + \lambda\boldsymbol{d})$ for fixed $\boldsymbol{u}$ and $\boldsymbol{d}$ is convex in the step size λ. Therefore its derivative

$$\frac{\partial}{\partial\lambda}E(\boldsymbol{u} + \lambda\boldsymbol{d}) = \boldsymbol{d}^\intercal\boldsymbol{g}(\boldsymbol{u} + \lambda\boldsymbol{d}), \tag{7}$$

i.e., the gradient's projection onto the search direction, has at most one zero. Hence, if we require that the step size λ is selected in the line search such that $\boldsymbol{d}^\intercal\boldsymbol{g}(\boldsymbol{u} + \lambda\boldsymbol{d}) \leq 0$, this guarantees that $E(\boldsymbol{u})$ decreases, without the need for checking the function value itself.

Note that avoiding energy evaluation precludes the use of standard *sufficient decrease conditions* (most commonly, Armijo condition) in the line search. However, a simple backtracking search, starting with $\lambda = 1$, for a point along the search direction with negative projected gradient, ensures that the Newton step, when it is less than one, is always in the range $[\lambda_m/2, \lambda_m]$, where λ_m is the function's (unknown) minimum point along the search line. One can show that this is sufficient for convergence by following the standard analysis of Newton's method with inexact line search. However, this is, in general, not sufficient to guarantee that the algorithm converges *quadratically*. An additional Armijo-like condition (the first termination condition in the line search in Algorithm 1; we use $\alpha = 0.1$, with a meaning similar to the Armijo condition constant) yields a more consistent quadratic behavior. The practical effect of this additional termination condition is small in most cases (most commonly, the reduction in the number of iterations on our test datasets is around 1-2). A detailed analysis of convergence of the proposed energy-free method can be found in [Zorin 2021].

Termination. The accuracy with which the target angles $\hat{\boldsymbol{\Theta}}$ can be matched of course depends (in a non-trivial manner) on the precision of the real number representation. If tolerance ε_{tol} is chosen too low relative to this, Algorithm 1 may never terminate. For practical purposes therefore additional stopping criteria can be taken into account: an upper bound on the number of Newton steps and the number of line search halvings, a lower bound on the Newton decrement $\boldsymbol{d}^\intercal\boldsymbol{g}(M, \boldsymbol{\ell}, \boldsymbol{u})$. Information about the practically achievable accuracy can be found in Section 7.3.

Additional Performance Heuristic. In particularly challenging cases, the gradient direction and in particular its magnitude can be rapidly varying. The line search loop may then have to be executed many times before a valid step size is found, causing many redundant edge flips. One additional line search heuristic that proved beneficial in this regard is a *gradient norm decrease* condition. Specifically, as a stopping condition for the line search we require that, in addition to $\boldsymbol{d}^\intercal\boldsymbol{g} < 0$, the norm of the gradient $\|\boldsymbol{g}\|$ decreases. Only if this additional condition forces the step size below a given threshold (we use 10^{-10}), the condition is lifted for one step, allowing the gradient to grow, so as to not hamper convergence.

Overlay Mesh. An embedding of the (by edge flips) modified mesh in the original mesh can be maintained by using a mesh overlay data structure. Towards the algorithm it behaves like a mesh, but internally it keeps track of the *overlay* of both meshes, updating it whenever an edge is flipped. [Fisher et al. 2007] propose to represent it explicitly by means of a polygon mesh data structure, [Gillespie et al. 2021] propose a more lightweight implicit representation by normal coordinates. We found the overhead of even the explicit structure to be benign (e.g., on average 11% added time cost on the example cases from Figure 9).

5 BOUNDARIES

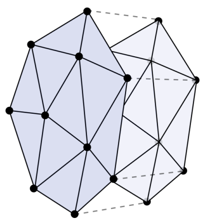
So far, we assumed that M is a closed surface. For a surface with boundary, the problem can be reduced to the case of closed surfaces by gluing a mirrored copy to the surface along the boundary, turning it into a closed surface with an obvious (reflectional) symmetry. A strategy of this kind is also used in [Sun et al. 2015] and [Gillespie et al. 2021].

However, the initial symmetry of the setting may be disturbed when applying Algorithm 1. Due to numerical inaccuracies, the values $\boldsymbol{u}$ may diverge on the two copies; application of a standard Delaunay flip algorithm is further complicated by the presence of stably cocircular configurations, as we discuss below. Therefore we describe a version of this surface double cover approach that explicitly imposes and maintains symmetry, on the numerical as well as the combinatorial level, by construction.

5.1 Double Cover

Let the input surface be N. Its double cover is constructed as follows:

(1) we attach a mirrored copy N' of the input mesh N along the boundary (merging boundary vertices and edges), as illustrated below, yielding a closed mesh M,
(2) we transfer the edge lengths $\boldsymbol{\ell}$ and the target curvatures κ_i of interior vertices v_i from N to N',
(3) we prescribe $\hat{\Theta}_i = 2\pi - 2\hat{\kappa}_i$ at each (former) boundary vertex v_i, where $\hat{\kappa}_i$ is the target discrete geodesic boundary curvature at vertex v_i.

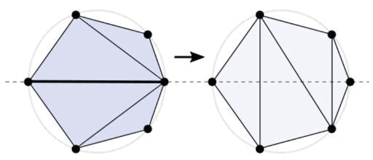


The double cover mesh M built this way exhibits an obvious reflectional symmetry, i.e., there is a map R with $R^2 = I$ that takes vertices to vertices, edges to edges, and faces to faces. It maps an element in the interior of N to its copy in N' and vice versa; on the merged (former) boundary vertices and edges, R is the identity.

Conformal Metric Symmetry. Due to symmetry (i.e., invariance with respect to R) of the mesh M, the metric $\boldsymbol{\ell}$, and the target angles $\hat{\Theta}$, the symmetrically initialized factors $\boldsymbol{u}$ will (in theory, up to numerical round-off error) remain symmetric after each iteration of the optimization process. This can be seen by observing that the function $E(\boldsymbol{u})$ is the sum of per-triangle terms $E_T(\boldsymbol{u}_T)$, where $\boldsymbol{u}_T$ is the restriction of $\boldsymbol{u}$ to vertices of the triangle T. Given the above symmetry, its gradient $\boldsymbol{g}(\boldsymbol{u}) = \nabla_{\boldsymbol{u}} E$ therefore is invariant with respect to R. Consequently, if we cut the mesh along the symmetry line in the end, so as to discard one copy, a boundary vertex v_i will have exactly half the prescribed angle, $\frac{1}{2}\hat{\Theta}_i = \pi - \hat{\kappa}_i$, and therefore exhibit a discrete geodesic boundary curvature of $\hat{\kappa}_i$, just as intended.

Tufted Double Cover. The fact that $\boldsymbol{u}$ (and thus all vertex-associated attributes) are supposed to evolve symmetrically implies that we can use a *tufted* double cover as in [Sharp and Crane 2020], with the unknown scale factors $\boldsymbol{u}$ shared between the two symmetric halves of M, to reduce the number of variables (and to impose perfect symmetry on the numerical level). This does not mean, however, that computations could trivially be restricted entirely to one half of the double cover only: edge flips may, and commonly will, create edges and faces spanning both halves of the double cover, crossing the symmetry line.

Combinatorial Symmetry. Edge flips across the symmetry line can lead to triangulations that are no longer combinatorially symmetric, as depicted in the inset. Unless special care is taken, this can increase the chance of numerical inaccuracies causing divergence from geometric symmetry. Furthermore, such cases contain co-circular vertex configurations that are *stable*, i.e., for the given triangulation, due to the symmetry of $\boldsymbol{u}$, these remain co-circular *independent of the evolution of* $\boldsymbol{u}$. An example is the diagonal edge on the right in the inset. As in this case, numerical evaluation of the Delaunay condition results in an essentially random choice of the result, in order to avoid potentially infinite flip sequences of Delaunay-critical edges, we instead perform special flips at the symmetry line, maintaining perfect combinatorial symmetry by construction, as detailed in the next section. Our method explicitly identifies these stably cocircular configurations and ensures that Delaunay flips are never applied to these, even if they appear to be slightly non-Delaunay due to numerical inaccuracies. In addition, having a symmetric Delaunay mesh for the final configuration can simplify the extraction of the resulting metric or map for the original surface with boundary.



5.2 Symmetric Meshes

Our goal is to rigorously determine which edge flip cases can occur in a symmetric mesh, in particular at the symmetry line, so as to ensure all special cases are correctly handled in our method. To that end, we begin by making precise the general notion of *combinatorially symmetric* polygon mesh. In this, rather than using edges, we use *halfedges*, each associated with a unique face (or a boundary loop, which can be treated exactly like a face). Specifically, each edge corresponds to two halfedges.

Definition 3 (Combinatorial Mesh). *A combinatorial polygon mesh is a triple $(H, \mathcal{N}, O)$ of a set of halfedges H, a bijective function $\mathcal{N} : H \to N$* (next-halfedge *function), and a bijective function O* (opposite-halfedge *function) with the property*

$$O^2(h) = h;\ O(h) \neq h \tag{8}$$

i.e., all orbits of O have size 2.

This definition is quite general which is important for maintaining intrinsic Delaunay triangulations: e.g., it allows for vertices of valence 1, polygons glued to themselves, etc., all of which are possible configurations in these triangulations.

Definition 4 (Mesh Elements). *Define the bijective circulator function $C : H \to H$ to be $\mathcal{N}^{-1}(O(h))$. Then the mesh has the following implied elements:*

- Faces *are the orbits of the next-halfedge function $\mathcal{N}$.*
- Vertices *are the orbits of the circulator function C.*
- Edges *are the orbits of the opposite-halfedge function O.*

Collectively we refer to them as (mesh) elements. *A halfedge* belongs *to an element if it is part of the respective orbit.*

A *mesh with boundary* is a mesh with a subset of its faces marked as boundary loops. The halfedges of these loops form the set H^{bnd} of *boundary halfedges*.

Definition 5 (Reflection Map). *A reflection map $R : H \to H$ for a mesh $(H, \mathcal{N}, O)$ is an involution ($R^2 = I$) defined on the set of halfedges: each halfedge is mapped either to itself, or forms a reflection pair with a distinct halfedge. It is required to satisfy the following conditions:*

(1) *preservation of O relation: $O(R(h)) = R(O(h))$,*
(2) *inversion of $\mathcal{N}$ relation: $\mathcal{N}(R(h)) = R(\mathcal{N}^{-1}(h))$,*
(3) *preservation of boundary: $h \in H^{bnd} \iff R(h) \in H^{bnd}$.*

Note that conditions (1) and (2) correspond to the properties of continuity and orientation-reversal of continuous reflection maps [Panozzo et al. 2012]. They imply that R maps orbits of $\mathcal{N}$, of O, and, as a consequence, of C, to orbits of these functions, i.e., it is well-defined for faces, edges, and vertices (via $R(x) = x' \iff R(h) \in x'$ for any $h \in x$). Furthermore, because $R^2 = I$, all orbits of R have length 1 or 2, whether it acts on halfedges, faces, edges, or vertices. This implies the following partitioning.

Proposition 2 (Halfedge Sets). *H can be partitioned into disjoint sets H^1, H^2, H^s so that the following conditions are satisfied:*

- *$h \in H^s \iff R(h) = h$;*
- *$h \in H^1 \iff R(h) \in H^2$;*
- *for any face or edge x, either all belonging halfedges are in H^1, or all in H^2, or x is fixed by R (i.e. $R(x) = x$).*

This leads to the following partitioning of the sets of edges and faces, where $e = (h, h')$, $f = (h_0, \ldots h_{m-1})$ denote the orbits of belonging halfedges:

- $e \in E_i \iff h, h' \in H^i$, $i = 1, 2$
- $e \in E^\perp \iff h, h' \in H^s$
- $e \in E^\parallel \iff h = R(h')$
- $f \in F_i \iff h_0 \in H^i$, $i = 1, 2$
- $f \in F^s \iff R(h_0) \in f$

The set $E^\perp$ is the set of edges (perpendicularly) crossing the symmetry line between two halves of a symmetric mesh mapped to each other (see Figure 5 right); the set $E^{\|}$ is the set of edges on the symmetry line; F^s is the set of faces that cross the symmetry line, and are mapped by the symmetry map to themselves. For additional details, see Appendix A.

Using this terminology, our double cover construction from Section 5.1 can be described formally in terms of combinatorial structure of the mesh (see Appendix B). Initially we have $E^{\perp} = \varnothing$ and $F^s = \varnothing$, i.e., no element crosses the symmetry line (the former boundary). $E^{\|}$ contains the edges lying *on* the symmetry line, i.e., those for whose halfedges the O relation was adjusted to glue the two copies. This initially simple situation can change, however, when edge flips are performed on the double cover mesh.

5.3 Symmetric Flips

When an edge e in a symmetric mesh $M = (H, \mathcal{N}, O, R)$ shall be flipped, the edge $R(e)$ needs to be flipped as well (unless $R(e) = e$), so as to be able to maintain a symmetric mesh. The simultaneous flip of e and $R(e)$ (as well as the single flip of e if $R(e) = e$) is referred to as *symmetric flip*. As discussed in Section 5.1, in the algorithm from Section 4 the metric evolves symmetrically. This implies that whenever the algorithm intends to flip an edge e, it simultaneously intends to flip $R(e)$ as well. The algorithm is therefore compatible with the restriction to symmetric flips.

While for an edge $e \in E_i$ with incident faces $f, g \in F_i$ the process is obvious, special care needs to be taken when elements from $E^{\|}$, $E^{\perp}$, or F^s are involved. We will exhaustively distinguish different types of symmetric flips based on the membership of the involved edges and faces in these sets.

Flip Types. For a triple (f_a, e, f_b) of an edge e with incident faces f_a, f_b, the triple of labels denoting their set memberships, e.g., $(1, \|, 2)$, is called flip *type* of the edge e.

Consistent Flip Types. We say that an edge has a *consistent* flip type, if this particular triple may occur in a symmetric mesh. For instance, $(1, \perp, 1)$ is not a consistent type, as edges from $E^{\perp}$ necessarily have incident faces from F^s by definition.

Proposition 3 (Appendix A) helps to reduce the possible set to the following six possibilities, up to a $1 \leftrightarrow 2$ exchange. It is easy to construct examples proving that all of them are consistent, i.e., may occur in a symmetric mesh:

• Edge in E^1:	Set 1a: $(1, 1, 1)$, $(1, 1, s)$	Set 1b: $(s, 1, s)$
• Edge in $E^{\|}$:	Set 2a: $(1, \|, 2)$	Set 2b: $(s, \|, s)$
• Edge in $E^{\perp}$:	Set 3: $(s, \perp, s)$	

Relevant Flip Types. Among these types, only four are also *relevant*; Following Proposition 4 (Appendix A), types of the form $(s, \|, s)$ and $(s, 1, s)$ in the sets 1b and 2b are necessarily associated with edges that satisfy the Delaunay condition Eq. (4) irrespective of the choice of lengths of edges involved. These are not relevant for the purpose of the algorithm from Section 4, which exclusively flips non-Delaunay edges. This leaves only sets 1a, 2a, and 3 for further consideration.

Triangles and Quadrilaterals. A flip of type $(1, 1, s)$ leads to a pair of triangles in F^s that together form a quadrilateral which is inscribed, i.e., the four vertices are intrinsically co-circular (Figure 5 center). Remarkably, this statement holds regardless of metric, as long as it is symmetric, i.e., invariant with respect to R. Instead of randomly choosing a diagonal splitting this quadrilateral into two triangles, we explicitly represent it as a quadrilateral face. This avoids violating the symmetry by the diagonal, which, e.g., would complicate recovering the surface with boundary after the conformal metric is computed, and avoids potential issues such as sequences of flips caused by numerically nearly co-circular points.

Faces in F^s can therefore be triangular or quadrilateral. We accordingly partition $F^s = F^t \cup F^q$, and based on this distinguish t-versions and q-versions of flip types involving the label s. This yields a total of seven types that are consistent and relevant.

Six of these seven flip types form three pairs of mutually inverse flips, while one is self-inverse. We can thus succinctly summarize :

$$\begin{array}{lrcl}
(1) & (1,1,1) + (2,2,2) & \leftrightarrow & (1,1,1) + (2,2,2); \\
(2) & (1,\|,2) & \leftrightarrow & (t,\perp,t); \\
(3) & (1,1,t) + (2,2,t) & \leftrightarrow & (t,\perp,q); \\
(4) & (1,1,q) + (2,2,q) & \leftrightarrow & (q,\perp,q).
\end{array}$$

Case (1) is the standard case of flipping a configuration not involving the symmetry line. (2), (3), and (4) are the special cases crossing the symmetry line; they are illustrated in Figure 5. Table 1 details the combinatorial changes to be performed on the symmetric mesh so as to execute these symmetric flips. In terms of implementation, it thus simply comes down to initially labeling the edges and faces of the double cover, updating the labels when flipping edges, and using one of these special case rules whenever a label other than 1 or 2 is involved in a flip.

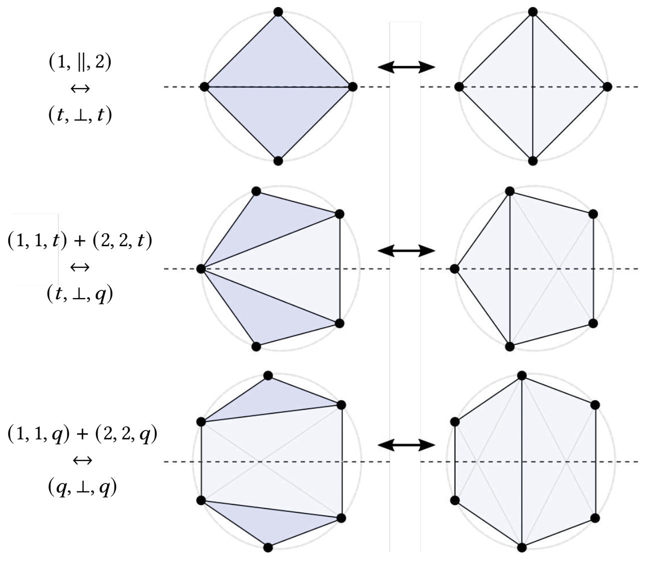


Fig. 5. Symmetric edge flips involving faces from F^s (light blue), crossing the symmetry line (dashed). Faces from F^1 and F^2 are colored dark blue. The configurations are shown with co-circular vertices, though combinatorially flips can be performed in any state. Note that the light blue quads' vertices, however, are necessarily co-circular by symmetry, regardless of metric.

5.4 Symmetric Metric

To be able to apply Algorithm 1 to such symmetric meshes to compute a symmetric conformal metric, what is left to clarify is how to deal with quadrilateral faces.

Delaunay Criterion. For edges with two incident triangles, the Delaunay check needed for the algorithm is standard, via Eq. (4). If one of the incident faces is a quad, due to symmetry it, regardless of the metric, is an inscribed trapezoid. As a consequence, whichever way we (virtually) split it into triangles we get the same angles opposite any of its edges. Hence, we may perform the Delaunay check assuming arbitrary virtual diagonals in the quads.

Gradient and Hessian. For the same reason, the computation of gradient $g(\boldsymbol{u})$ and Hessian $H(\boldsymbol{u})$ can be performed based on arbitrary diagonals; the choice does not affect the result [Springborn 2020].

Ptolemy Formula. Note that each of the edges created by symmetric flips involving quads (Figure 5) can also be obtained by a sequence of edge flips involving triangles (and split quads) only. In this way the length of such edges can be computed using (multiple instances of) the standard Ptolemy formula Eq. (6). As there are only four types of flips involving quads, one can conveniently derive closed form expressions for these cases in advance, rather than actually performing these sequences for each flip. Note that each quad needs to store its diagonal length to enable these computations.

5.5 Restriction to Single Cover

Once Algorithm 1 has terminated and the desired conformal metric has been computed, we finally need to discard half of the double cover: we need to cut the symmetric surface along the line of symmetry. While initially the entire symmetry line is formed by a sequence of mesh edges, this may no longer be the case due to flips (unless an overlay is used), namely whenever F^s and $E^\perp$ are not empty in the end. One simply needs to split all edges from $E^\perp$ at their midpoint, and split the triangles and quads from F^s by connecting these inserted split vertices.

6 CONTINUOUS MAPS FROM DISCRETE METRICS

The algorithms described in previous sections deal exclusively with discrete metric definitions, i.e., assignments of edge lengths to edges of a mesh. If mesh connectivity does not change, an affine map from the initial mesh triangles T to the final mesh triangles $\tilde{T}$ can be easily inferred from the lengths. However, as pointed out in [Springborn et al. 2008], a natural map is actually a projective map between triangles, which, in addition to mapping the original lengths to conformally deformed ones, also maps the circumcircle of T to the circumcircle of $\tilde{T}$. While for fixed connectivity this yields only a moderate improvement in, e.g., texture quality, for changing connectivity the map definition is more relevant.

While for the discrete algorithm itself we only needed a simple-to-formulate (although surprising) fact that Weeks flip algorithm can be used to obtain an intrinsically Delaunay mesh even if the triangle inequality is violated at intermediate steps, defining maps between the original mesh points and the final (e.g. flat) mesh points requires a more in-depth exposition of the underlying theory.

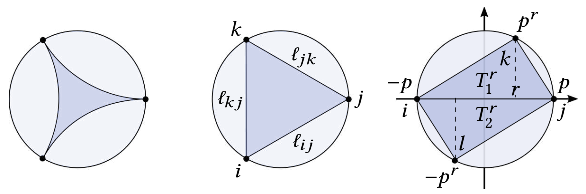


Fig. 6. Left: Poincaré model. Center: Beltrami-Klein model, both with an ideal triangle. Note that in the Beltrami-Klein model it forms a Euclidean triangle. Right: Two-triangle chart.

Our goal in this section is to define a map $f : |M| \to |M'|$, from the original to the final (e.g. conformally flattened) mesh, more specifically, mapping formulas of the form $(w'_l, w'_m, w'_m; T') = f(w_i, w_j, w_k; T)$, where (w_i, w_j, w_k) are barycentric coordinates of a point on the input triangle T_{ijk} in M and (w'_l, w'_m, w'_n) is the corresponding point on a triangle T'_{lmn} in mesh M'.

6.1 Cusped Hyperbolic Metric on Meshes

The central idea of the theory in [Gu et al. 2018b] and several other papers dealing with related problems is a construction of a hyperbolic metric corresponding to a Euclidean metric $\boldsymbol{\ell}$ which is invariant to conformal scale factors $\boldsymbol{u}$; in this context the lengths $\boldsymbol{\ell}$ are referred to as *Penner coordinates* of the hyperbolic metric.

Conformal deformations of $\boldsymbol{\ell}$ do not change this hyperbolic metric, and flips define just different triangulations of a fixed surface. The update of Penner coordinates for an edge flip using the Ptolemy formula Eq. (6) happens to produce a mesh that is isometric in the hyperbolic metric to the mesh before the flip. Next, we discuss the hyperbolic metric definition and isometric retriangulation in this metric in more detail.

Beltrami-Klein Model. We use the *Beltrami-Klein* hyperbolic plane model. The model represents the hyperbolic plane H^2 as the interior of a unit disk, with points of the boundary of the disk being points at infinity in the hyperbolic metric. These points (which are not a part of the hyperbolic plane, but play an important role in the model) are called *ideal points*. The model has the following properties.

- Lines are segments connecting points on the boundary.
- Given two distinct points p and q in the disk, the unique Euclidean straight line connecting them intersects the disk's boundary at two ideal points, a and b; label them so that the points are, in order, a, p, q, b along the line. The hyperbolic distance between p and q then is:
$$d_H(p,q) = \frac{1}{2} \log \frac{|aq|\ |pb|}{|ap|\ |qb|}$$
- *Isometries of the hyperbolic plane correspond to projective transformations preserving the unit disk.*
- An isometry is defined uniquely by specifying images of three points on the boundary of the disk (ideal points). There is an isometry mapping any three ideal points to any other three ideal points. We denote such projective maps $P[T \to T']$ where T and T' are triples of points on the unit disk (Figure 6 center).
- While angles are not preserved, if a line is a diameter, perpendicular lines are also perpendicular to it in the model.

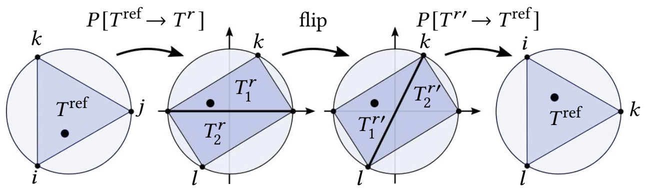


Fig. 7. Mapping a point through a single flip via a two-triangle chart.

Defining the Hyperbolic Metric. For a mesh M with vertices excluded, the hyperbolic metric is defined by mapping each mesh triangle, with edge lengths given by $\boldsymbol{\ell}$, to a similar Euclidean triangle inscribed in a unit disk, and using the Beltrami-Klein model to define the hyperbolic distances inside the triangle. Under this hyperbolic metric the triangles are ideal, with vertices at infinity (referred to as *cusped*, for reasons more obvious in the Poincaré model, see Figure 6 left). Furthermore they are all congruent, because there is a hyperbolic isometry, a projective circumcircle-preserving map, mapping one triangle to the other.

Note however, that unlike the case of finite triangles, the identification of sides of ideal triangles that are adjacent in M is not unique: because the sides are infinitely long, one can slide them along each other isometrically. The natural gluing defined by identifying points that correspond in the disk model picks one such isometric identification. One can show that if Penner coordinates $\boldsymbol{\ell}$ and $\tilde{\boldsymbol{\ell}}$ are related by a set of conformal scale factors $\boldsymbol{u}$, the resulting gluing between adjacent ideal triangles is the same, i.e., they define the same metric.

This allows a convenient definition of *two-triangle isometric charts* (Figure 6, right) for this metric, which provide most of what we need for defining our maps f across edge flips.

Two-Triangle Charts. Consider two adjacent triangles T_{ijk} and T_{jil} sharing edge e_{ij}, and five Penner coordinates $\ell_{ij}, \ell_{jk}, \ell_{ki}, \ell_{il}, \ell_{lj}$. For a single triangle, Penner coordinates can be changed arbitrarily using conformal deformations. Note however, that there are only four conformal scale factors u_i, u_j, u_k, u_l involved when mapping two adjacent triangles, so the five lengths cannot be chosen completely arbitrarily. The invariant that is preserved under these remappings is the *cross-ratio* $c_{ij} = (\ell_{jk}\ell_{il})/(\ell_{jl}\ell_{ki})$. Cross-ratio assignments to edges (*shear coordinates*) actually are in one-to-one correspondence with choices of cusped hyperbolic metrics on a fixed mesh.

We are thus free to choose the conformal scale factors u_i, u_j, u_k, u_l so that the following conditions are satisfied: (1) edge e_{ij} is mapped to the diameter $(-p, p)$, with $p = (1, 0)$, on the horizontal coordinate axis; (2) vertices k and l are mapped to antipodal points $p^r = (r, \sqrt{1 - r^2})$ and $-p^r$ on the circle. It is easy to check that the four scale factors are uniquely defined by these conditions, with r equal to $(1 - c_{ij})/(1 + c_{ij})$. Notice that $c_{ij} > 0$, thus $r \in (-1, 1)$, regardless of any triangle inequality condition. We denote these two chart triangles T_1^r and T_2^r.

Thus, an isometric atlas can be constructed for the whole mesh, by mapping each triangle pair to a chart as described above. This gives us the necessary tools to define the map f.

Mapping Across a Flip. Let M_k be a mesh obtained after applying a sequence of k flips to M, and M_{k+1} a mesh obtained by flipping a single further edge e_{ij} shared by triangles $T_1 = T_{ijk}$ and $T_2 = T_{jil}$ as above. Each mesh has length assignments $\boldsymbol{\ell}_k$, but as these are guaranteed to satisfy triangle inequalities only at certain steps k where the Delaunay condition is satisfied, these are best viewed as Penner coordinates for a hyperbolic metric.

As barycentric coordinates are not invariant with respect to projective maps, we need to choose a reference triangle for barycentric representation (w_i, w_j, w_k). We use an equilateral reference triangle T^{ref}, with vertices q_0, q_1, q_2, with $q_s = (\cos 2s\pi/3, \sin 2s\pi/3)$ for any triangle T_1 of M_k, see Figure 7 left.

In the two-triangle chart, T_1 and T_2 are mapped to T_1^r and T_2^r. After the flip in the chart, the new chart triangles, corresponding to triangles $T_1' = T_{jkl}$ and $T_2' = T_{ilk}$ are $T_1^{r\prime} = (p, p^r, -p^r)$ and $T_2^{r\prime} = (-p, -p^r, p^r)$, see Figure 7 center. If the image of the point (w_i, w_j, w_k) in the chart belongs to triangle T_1' then the map $(w_i, w_j, w_k) \to (w_i', w_j', w_k')$ is obtained as the composition of circumcircle-preserving projective maps:

$$\begin{aligned}(w_j', w_k', w_l') &= f(w_i, w_j, w_k) = \\ &\left(P[T_1^{r\prime} \to T^{\text{ref}}] \circ B \circ P[T^{\text{ref}} \to T_1^r]\right)(w_i, w_j, w_k)\end{aligned} \tag{9}$$

where B is the matrix converting barycentric coordinates on T_1^r to barycentric coordinates on $T_2^{r\prime}$. The expression is similar in the case when the image of the point in the chart lands in $T_2^{r\prime}$. The circumcircle-preserving projective maps P can be computed in barycentric coordinates using the following formula:

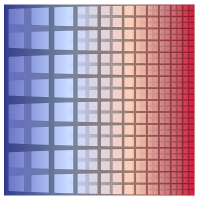
$$P(w_i, w_j, w_k) = (w_i S_i, w_j S_j, w_k S_k)/(w_i S_i + w_j S_j + w_k S_k)$$

with $S_i = \frac{\ell_{ij}\ell_{ki}\tilde{\ell}_{jk}}{\tilde{\ell}_{ij}\tilde{\ell}_{ki}\ell_{jk}}$, where $\boldsymbol{\ell}$ are lengths of the source, and $\tilde{\boldsymbol{\ell}}$ are lengths of the target triangle.

7 EVALUATION

We have implemented Algorithm 1 (with support for boundaries following Section 5) in C++. Our goal is to assess how well this theoretically sound method performs practically. While by default we use standard double precision floating point numbers, the optional use of extended precision arithmetics in our implementation allows us to assess to what extent potential convergence issues are related to finite precision or other problems, as detailed in Sections 7.3 and 7.4. We find that, as conformal maps can easily involve a very large range of scales across a mesh, for certain challenging settings the use of extended precision arithmetics can be essential to yield results of adequate quality.

In cases where a (mostly) flat metric is computed, the result can be visualized by turning the metric into a map (using a layout of the flat mesh in the plane [Springborn et al. 2008]) and mapping a texture (e.g. a grid or checkerboard) to the surface using this map. For a clear visualization in cases with high scale distortion, we use a procedurally generated hierarchical grid texture, as illustrated here. Its density is chosen adaptively based on the pointwise magnitude of the scale distortion on the surface mesh, halving the spacing between texture lines when the scale factor of the conformal map is halved.



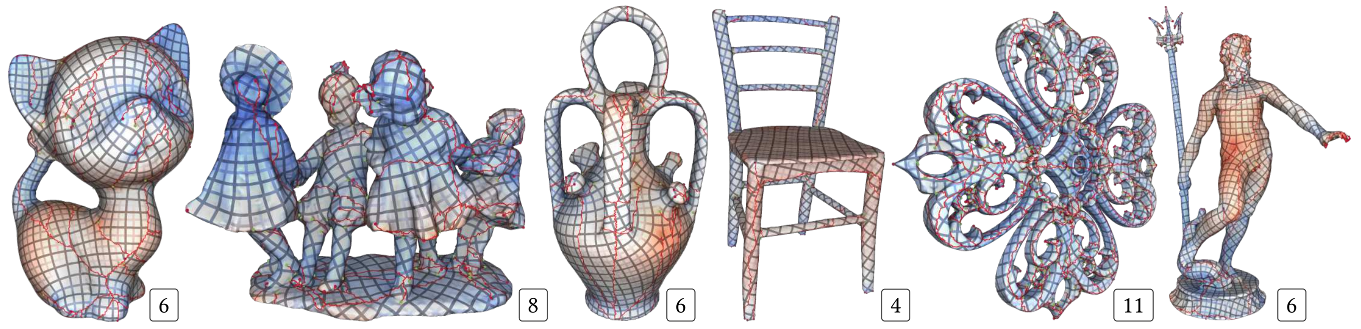


Fig. 8. Visualization of conformal maps, implied by conformal cone metrics, on some of the closed models with angle prescriptions from the dataset of [Myles et al. 2014]. The numbers indicate the scale range (difference of maximal and minimal conformal (natural) logarithmic scale factor $\boldsymbol{u}$) for each model. Cones are marked by red and green dots; texture jumps due to cones are marked red. The textured map and scale visualization follow the description from Section 7.

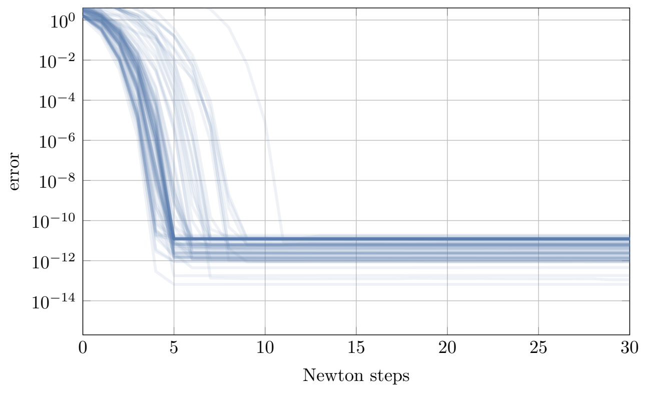


Fig. 9. Decay of maximum angle error $\|\hat{\Theta} - \Theta\|_\infty$ over the iterations of the Newton algorithm. Each graph represents one of the closed-surface instances from the dataset of [Myles et al. 2014].

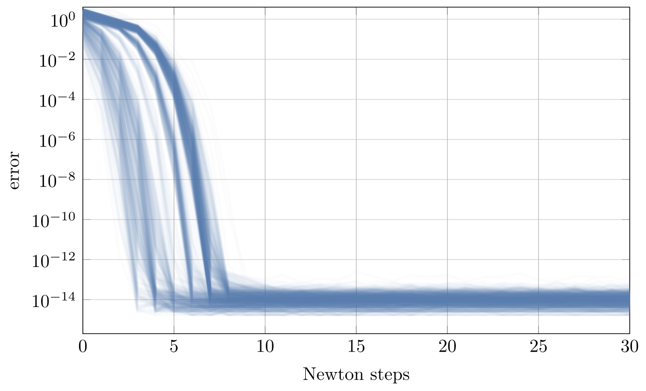


Fig. 10. Like Figure 9, but each graph represents one of 1000 random test instances (again without boundary)

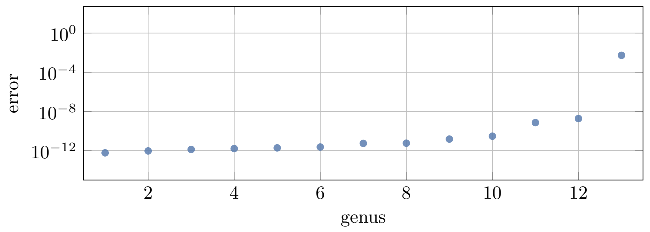


Fig. 11. Final residual angle error for the extreme case of concentrating all curvature in a single cone on an g-torus surface (genus g). For the genus 12 case, where the residual error is still benign, the conformal scale factor spans 232 orders of magnitude. For the problematic genus 13 case it surpasses 262. By increasing numerical precision (Section 7.3), this can be remedied; for instance, with 200-bit precision, the $g = 150$ case converges to below 10^{-29}, with 400-bit precision, the $g = 400$ case to below 10^{-65} (with the scale factors spanning 611 orders of magnitude). (To reduce numerical issues in this extreme experiment, the initial step size λ was halved until the range of the coefficients of $\lambda\boldsymbol{d}$ was less than 10.)

7.1 Validation

Closed Surfaces. A dataset of mesh models together with angle prescriptions $\hat{\Theta} > 0$ (based on cones of cross fields) has been released with [Myles et al. 2014]. We applied our implementation to the closed models from this dataset. The angle error decay in the course of the algorithm on these cases is visualized in Figure 9. Some of the models with the resulting conformal map are visualized in Figure 8. We observe that the models reach angle accuracy of 10^{-10} in less than 15 Newton iterations. The final achievable accuracy varies and is correlated with the range of scale factors in the final mesh (cf. Figure 20), as a large variation of scale factors leads to a moderate loss of precision in the gradient computation.

As further test instances, we use 1000 different random target angle prescriptions $\hat{\Theta}$ (with $\hat{\Theta}_i \in (\pi, 3\pi)$ for all vertices v_i) on a sphere mesh (1K vertices). The error decay is visualized in Figure 10. Note that the overall behavior is very similar, whether the prescribed angles are random or geometrically meaningful (as in Figure 9).

We consider the extreme scenario of concentrating the target metric's entire curvature in one point (i.e., prescribing a single cone of angle $2\pi(2g-1)$ in an otherwise flat metric). Errors for surfaces of increasing genus g (procedurally generated g-tori) are shown in Figure 11. A blow-up of the configuration around the single prescribed cone vertex on a genus 6 example is shown in Figure 13.

Surfaces with Boundary. The above mentioned dataset from [Myles et al. 2014] also contains meshes with one or more boundary loops, together with angle prescriptions $\hat{\Theta} > 0$ for interior and boundary

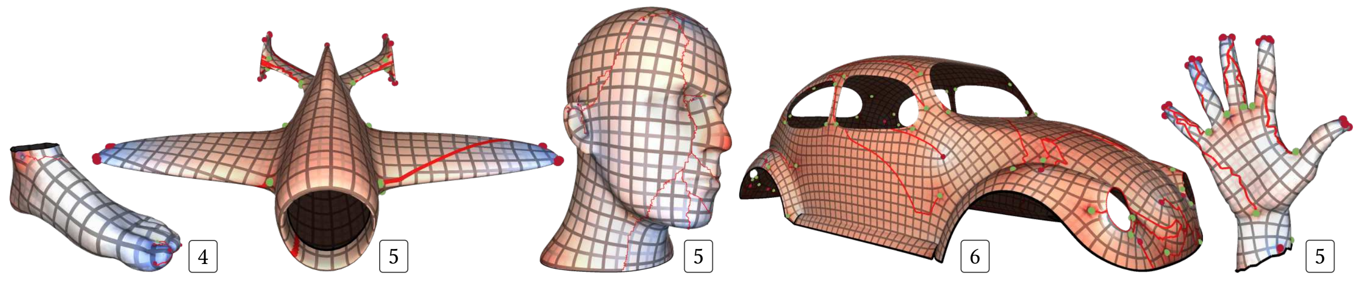


Fig. 12. Visualization of conformal maps, analogous to Figure 8, on some of the models *with boundary* from the dataset of [Myles et al. 2014]. The boundary geodesic curvature is prescribed to be zero, therefore the angle between texture grid lines and the boundary is constant per boundary loop.

vertices. The error decay on these cases is shown in Figure 14. Some of the models are visualized in Figure 12. The behavior is overall similar to the closed surface case.

As a synthetic test, we generate 1000 different random target angle prescriptions $\hat{\Theta}$ for a surface with boundary (a disk with 5K vertices). In the interior we prescribe a flat metric, at the boundary we prescribe a geodesic curvature, maximally in the range $\pm\pi$, i.e., $\hat{\Theta}_i \in (0, 2\pi)$ for all boundary vertices v_i. Figure 15 shows the number of the different types of symmetric flips that are performed in the course of the algorithm on these cases. As expected, the number of flips is larger for cases with a prescribed curvature spanning a larger range.

Another relevant scenario is that of prescribed geodesic curvature along a cut graph. We take the closed models of the dataset from [Myles et al. 2014] and mimic the setting employed by [Campen et al. 2019]: we compute a cut graph on each of these surfaces, and prescribe $\hat{\Theta}_i = \pi$ along this cut graph's segments' from both sides (effectively asking them to be straight under the conformal

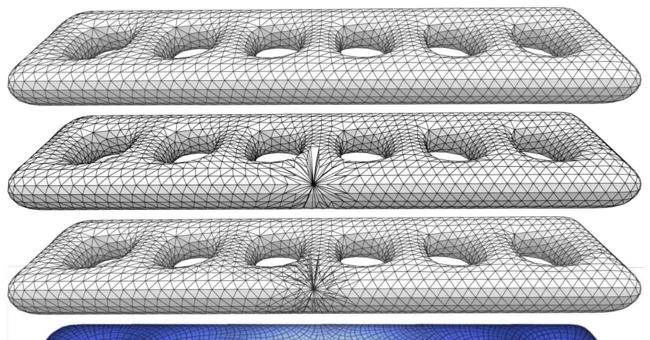


Fig. 13. Top: Input triangulation. Second row: Resulting intrinsic retriangulation, when concentrating all curvature on a single vertex ($\Theta = 22\pi$); it is Delaunay under the computed conformal metric (with curvature -20π at the central vertex). Third row: overlay triangulation [Fisher et al. 2007], allowing for a simple representation of the implied conformal map, linear or projective per triangle. Bottom: Visualization of implied conformal map using a hierarchical grid texture (spanning 25 levels in this extreme case).

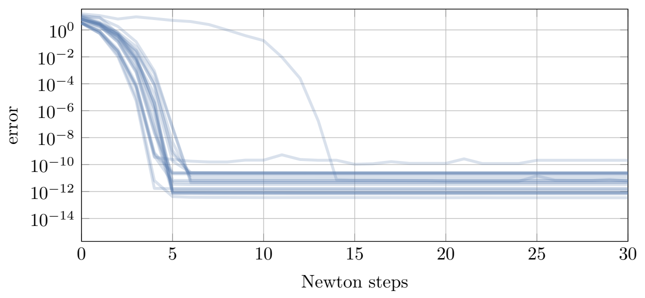


Fig. 14. Decay of maximum angle error $\|\hat{\Theta} - \Theta\|_\infty$ over the iterations of the Newton algorithm. Each graph represents one of the instances *with boundary* from the dataset of [Myles et al. 2014].

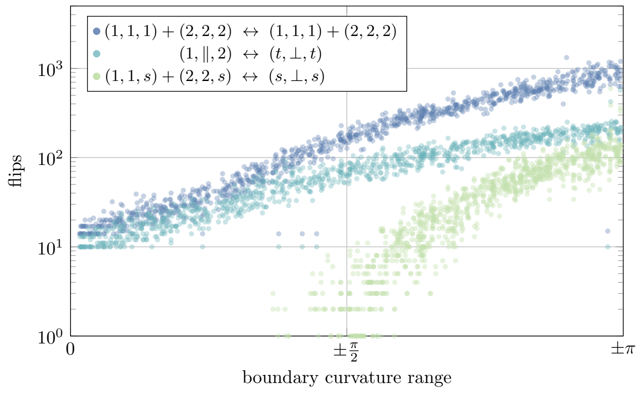


Fig. 15. Scatter plot showing the numbers of different types of symmetric flips during the algorithm relative to the range of prescribed random boundary curvatures. Each dot represents one type of flips for one of 1000 test instances.

metric). The cut graph is composed of g short handle loops computed as in [Diaz-Gutierrez et al. 2009], connected by additional shortest paths. The resulting cut forms a graph on the surface with nodes of valence 3; at each node, an angle of π is prescribed for the largest sector, and angles $\pi/2$ for the remaining two. Some of the models are visualized in Figure 16, with the cut graph marked in black. Depending on the shape of the cut graph, this scenario turns out to be the most challenging numerically: As can be seen in Figure 17 left, in a few cases the final maximum error is over 10^{-10}, i.e., higher than in the previously discussed scenarios. This

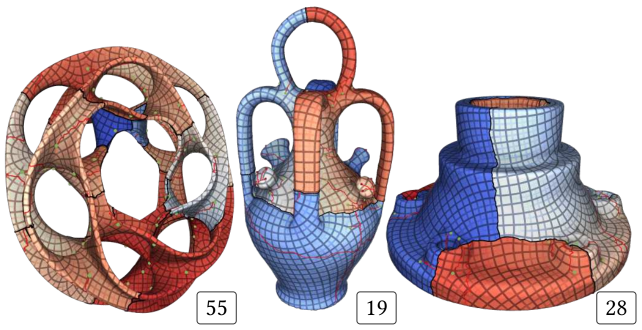


Fig. 16. Visualization of conformal maps with cones, analogous to Figure 8, on models cut to disk topology using a cut graph (black). Due to the prescribed geodesic curvature along the cut boundary, the cut is axis-aligned under the map. Notice that such enforced alignment can easily imply a broad range of scales, which is challenging numerically.

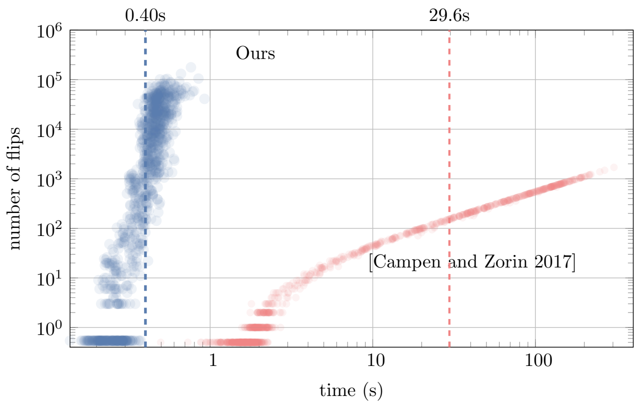


Fig. 18. Scatter plot showing the number of flips and the run time (to reach $\varepsilon_{\text{tol}} = 10^{-10}$), for the described Delaunay-flip method (blue) and the degeneration flip method (red). Each dot represents one of 1000 test instances. Dashed lines mark the average run time, 0.4s and 29.6s, respectively.

is related to the scale distortion of the implied conformal metric spanning a range of up to 73 orders of magnitude in these cases. With higher-precision arithmetic, these residuals can be reduced, as discussed in Section 7.3.

7.2 Comparison

We demonstrate the advantages of the Delaunay flip approach over the degeneration flip approach (Section 3.2) in terms of efficiency as well as numerical robustness. To this end, we apply an implementation of the described method and an implementation of the algorithm described by [Campen and Zorin 2017b] (both using standard double precision floating point numbers) to the same set of inputs.

Efficiency. The main differences between the two methods lie in the number of linear system solves (to compute the descent direction $\boldsymbol{d}$) and the number of intrinsic flips. In the proposed method, the number of flips is often significantly higher (see the discussion in Section 3.2), while the number of system solves is lower. As a flip is a cheap local operation, while a system solve is an expensive global operation, a run time benefit can be conjectured.

The scatter plot in Figure 18 shows that this is the case on average. As test instances we use 1000 different random target angle prescriptions $\hat{\Theta}$ (with $\hat{\Theta}_i \in (\pi, 3\pi)$ for all vertices v_i) on a sphere mesh (10K vertices). Only for relatively simple cases, where the target curvature can be matched without degeneration flips, the number of system solves may be similar. On average, run time is 73× lower with the Delaunay-based method on these examples.

Robustness. Differences in robustness can best be observed by considering extreme cases. In Figure 19 we show the residual error of the two methods when prescribing one very small or very large target angle (while distributing the remaining curvature). For small angles it becomes apparent that the degeneration flip algorithm is numerically more fragile.

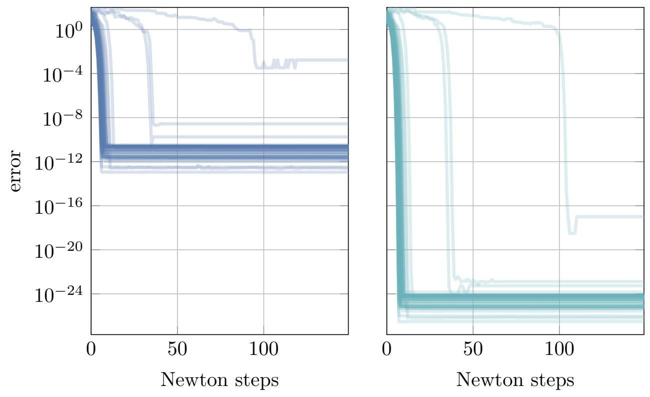


Fig. 17. Decay of maximum angle error $\|\hat{\Theta} - \Theta\|_\infty$ over the iterations of the Newton algorithm. Each graph represents one of the closed instances from the dataset of [Myles et al. 2014], with *prescribed curvature along a cut graph.* Left: double precision. Right: extended precision (100 bits mantissa).

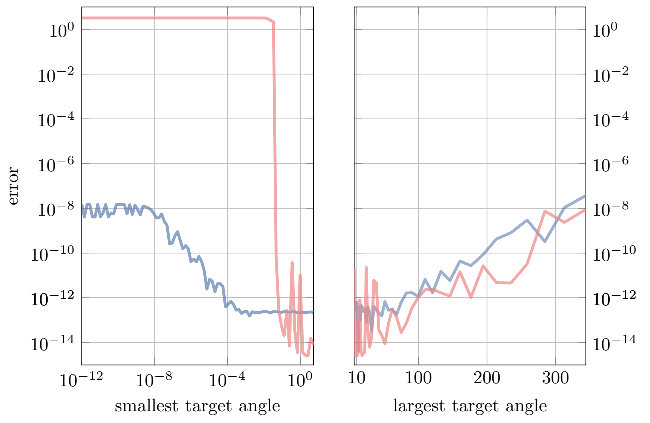


Fig. 19. Final residual angle error $\|\hat{\Theta} - \Theta\|_\infty$ for extreme cases (one very small or very large target angle, on a sphere with 1K vertices), comparing the Delaunay-based algorithm (blue) and the degeneration flip algorithm. [Campen and Zorin 2017b] (red).

7.3 Accuracy

While the method is theoretically guaranteed to yield the desired result, in practice numerical inaccuracies limit how closely the target curvature will be matched. As the method involves exponential and trigonometric functions (Eqs. (1) and (2)), it cannot be implemented in a numerically exact manner using adaptive precision rational or integer number types. Using extended precision floating point number types (such as MPFR), the method's accuracy can, however, be increased arbitrarily. We evaluate the effect of this choice on result accuracy in Figure 20. As test instances we use 1000 different random target angle prescriptions $\hat{\Theta}$ (with $\hat{\Theta}_i \in (\pi, 3\pi)$ for all vertices v_i) on a sphere mesh (1K vertices).

As can be observed, the remaining error decreases consistently as the number of bits used for the floating point computations is increased. Due to dependence on many factors (input mesh and edge lengths, target angles, choice of linear system solver for the Newton direction) a simple bound on the error cannot be given, but Figure 20 gives an empirical idea of the behavior. Note that some correlation can be observed to the conformal scale distortion (the range $[e^{\min \boldsymbol{u}}, e^{\max \boldsymbol{u}}]$) that is required to match the target curvature.

In Figure 17 right the effect of increased precision on test cases from Section 7.1 can be observed. In particular, for the models that have maximum error over 10^{-10} when using standard double precision arithmetic, the error is reduced to below 10^{-16} when using a 100 bits mantissa instead.

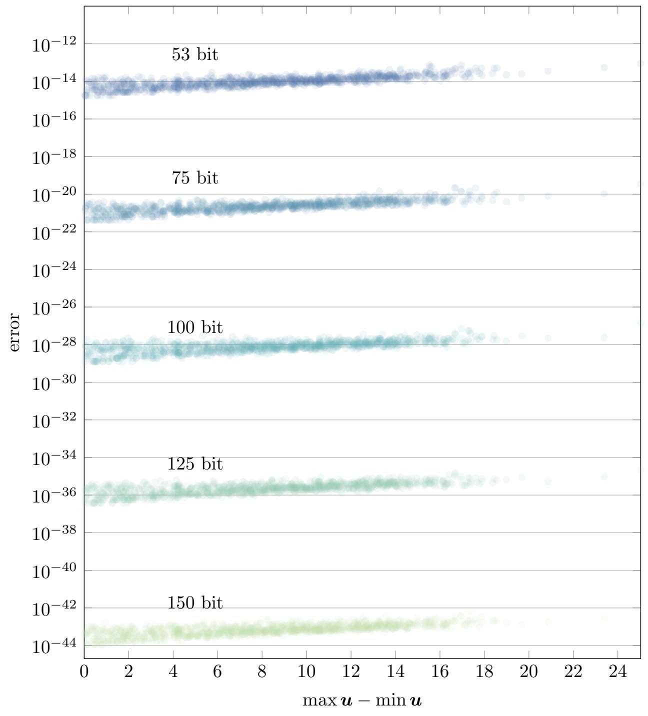


Fig. 20. Scatter plot showing residual angle error $\|\hat{\Theta} - \Theta\|_\infty$ (after at most 50 Newton steps) relative to the range of logarithmic conformal scale factors $\boldsymbol{u}$. Each dot represents one test instance, run using floating point numbers with a mantissa of 53 bits (`double`), 75 bits, 100 bits, 125 bits, 150 bits (`MPFR`).

7.4 Failure Modes

We can distinguish two types of potential issues (detailed below): high residual error or a high number of optimization steps. The former can be caused by limited arithmetic precision (and can therefore be remedied by using extended precision, as demonstrated above). The latter can be caused by an unfavorable energy landscape, and is therefore more fundamental, regardless of numerics.

Precision-related failures. Depending on the target curvature, a high amount of metric distortion may be required, with negative numerical effects on result accuracy. It can be observed that this is correlated with local concentrations of positive or negative target (Gaussian or geodesic) curvature. Figure 21 left shows an experiment in which an increasingly large cluster of vertices have a target angle below 2π and the rest above 2π. When using standard double precision, a large fraction of these synthetic test cases essentially fails to reach a reasonably accurate state. Performing these computations with higher precision (Figure 21 right) resolves these problems. Analogously, we notice that cut graphs with more complex shape than the ones used in Figure 17 (e.g., the more constrained "hole-chain" in [Campen et al. 2019]) cause a similar behavior.

Near-degeneracy failures. This second issue is more fundamental. While the method may, in principle, converge eventually, the step size can decrease to the point that the number of iterations needed becomes impractical. When using the above mentioned hole-chain choice of cuts on the dataset of [Myles et al. 2014], we can identify four high-genus models with complex singularities which fail to converge in a reasonable number of steps even with high-precision arithmetic. The underlying reason is illustrated in Figure 22, showing the plot of the projected gradient $\boldsymbol{d}^\top \boldsymbol{g}(\boldsymbol{u} + \lambda \boldsymbol{d})$ for a line search direction. One can see that while theoretically the gradient is C^1, it may experience very significant jumps, when a large number of triangle flips happen nearly simultaneously as λ changes (in this particular case 58). We observe that this occurs in particular in the presence of highly distorted near-degenerate triangles.

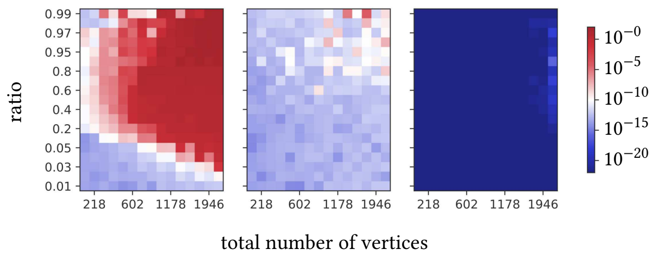


Fig. 21. Heatmap showing the final error $\|\hat{\Theta} - \Theta\|_\infty$ for spheres of varying resolution (x-axis) with some ratio (y-axis) of the vertices set to target angle 3 and the rest to a constant target angle $< 2\pi$ such that the Gauss-Bonnet theorem is satisfied. Left: double precision results when the two angle values are distributed in two clusters. Center: double precision results when the two angle values are distributed randomly over the sphere. Right: extended precision (150 bits mantissa) results with the same distribution as left. (For this experiment, the threshold for the gradient norm decrease was set to 0 and, to reduce the run time in this particular case, λ was chosen adaptively, initially halved until the range of coefficients of $\lambda \boldsymbol{d}$ was less than 10.)

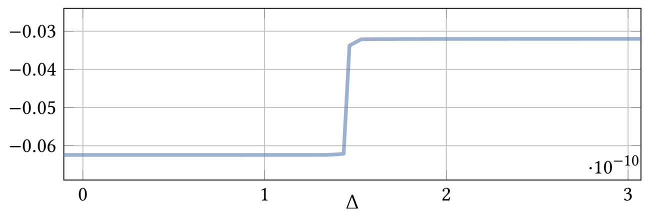


Fig. 22. Projected gradient $\boldsymbol{d}^\intercal \boldsymbol{g}(\boldsymbol{u} + \lambda \boldsymbol{d})$ along the normalized Newton descent direction with step length $\lambda = 0.0217745227 + \Delta$.

8 CONCLUSIONS AND FUTURE WORK

We presented a practical realization of the method for computing discrete conformal maps based on the ideas of [Gu et al. 2018b; Springborn 2020], elaborating how it can be applied safely to meshes with boundary, the most practically relevant scenario for conformal mapping. Our improvements include a straightforward to implement algorithm for maintaining symmetric Delaunay triangulations and several improvements increasing the robustness of Newton's optimization method in the context of our application. We explored its behavior on a standard dataset, and for a number of challenging synthetic examples, demonstrating its robustness for a broad range of cases involving high distortion. We also observe that common failure cases can be addressed by using extended precision arithmetic, albeit at a significant cost in run time.

However, in our extensive tests we did identify a small number of cases for which the method does not produce a conformal map in reasonable time, which indicates potential for further algorithmic improvements. It would also be desirable to find ways to minimize the use of extended precision arithmetic to the minimum necessary in a *filtered* approach, so as to increase accuracy while maintaining performance. Finally, extension of the method from Euclidean to spherical and hyperbolic discrete metrics would be not only of theoretical interest [Schmidt et al. 2020].

ACKNOWLEDGMENTS

Supported by NSF CAREER award 1652515, DMS-1821334, OAC-1835712, OIA-1937043, CHS-1908767, CHS-1901091, a gift from Adobe Research, a gift from nTopology, and a gift from Advanced Micro Devices, Inc.

REFERENCES

Julien Basch, Leonidas J Guibas, and John Hershberger. 1999. Data structures for mobile data. *Journal of Algorithms* 31, 1 (1999), 1–28.

Mirela Ben-Chen, Craig Gotsman, and Guy Bunin. 2008. Conformal Flattening by Curvature Prescription and Metric Scaling. *Computer Graphics Forum* 27, 2 (2008).

Alexander I Bobenko and Boris A Springborn. 2007. A discrete Laplace–Beltrami operator for simplicial surfaces. *Discrete & Computational Geometry* 38, 4 (2007), 740–756.

Marcel Campen, Hanxiao Shen, Jiaran Zhou, and Denis Zorin. 2019. Seamless Parametrization with Arbitrary Cones for Arbitrary Genus. *ACM Trans. Graph.* 39, 1 (2019).

Marcel Campen and Denis Zorin. 2017a. *On Discrete Conformal Seamless Similarity Maps.* arXiv:1705.02422 [cs.GR]

Marcel Campen and Denis Zorin. 2017b. Similarity Maps and Field-Guided T-Splines: a Perfect Couple. *ACM Trans. Graph.* 36, 4 (2017).

Keenan Crane. 2020. Discrete Conformal Geometry. In *Proceedings of Symposia in Applied Mathematics.* American Mathematical Society.

Mathieu Desbrun, Mark Meyer, and Pierre Alliez. 2002. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum* 21, 3 (2002), 209–218.

Pablo Diaz-Gutierrez, David Eppstein, and Meenakshisundaram Gopi. 2009. Curvature aware fundamental cycles. *Computer Graphics Forum* 28, 7 (2009), 2015–2024.

Matthew Fisher, Boris Springborn, Peter Schröder, and Alexander I Bobenko. 2007. An algorithm for the construction of intrinsic Delaunay triangulations with applications to digital geometry processing. *Computing* 81, 2-3 (2007), 199–213.

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

Mark Gillespie, Boris Springborn, and Keenan Crane. 2021. Discrete Conformal Equivalence of Polyhedral Surfaces. *ACM Trans. Graph.* 40, 4 (2021).

Xianfeng Gu, Ren Guo, Feng Luo, Jian Sun, and Tianqi Wu. 2018a. A discrete uniformization theorem for polyhedral surfaces II. *Journal of Differential Geometry* 109, 3 (2018), 431–466.

Xianfeng Gu, Feng Luo, Jian Sun, and Tianqi Wu. 2018b. A discrete uniformization theorem for polyhedral surfaces. *Journal of Differential Geometry* 109, 2 (2018), 223–256.

Xianfeng Gu and Shing-Tung Yau. 2003. Global conformal surface parameterization. In *Proc. Symp. Geometry Processing 2003.* 127–137.

Miao Jin, Junho Kim, and Xianfeng David Gu. 2007. Discrete surface Ricci flow: Theory and applications. In *IMA International Conference on Mathematics of Surfaces.* Springer, 209–232.

Liliya Kharevych, Boris Springborn, and Peter Schröder. 2006. Discrete conformal mappings via circle patterns. *ACM Trans. Graph.* 25, 2 (2006), 412–438.

Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics* 21, 3 (2002), 362–371.

Feng Luo. 2004. Combinatorial Yamabe flow on surfaces. *Communications in Contemporary Mathematics* 6, 05 (2004), 765–780.

Ashish Myles, Nico Pietroni, and Denis Zorin. 2014. Robust Field-aligned Global Parametrization. *ACM Trans. Graph.* 33, 4 (2014), 135:1–135:14.

Ashish Myles and Denis Zorin. 2012. Global parametrization by incremental flattening. *ACM Trans. Graph.* 31, 4 (2012), 109.

Daniele Panozzo, Yaron Lipman, Enrico Puppo, and Denis Zorin. 2012. Fields on Symmetric Surfaces. *ACM Trans. Graph.* 31, 4 (2012).

Igor Rivin. 1994. Euclidean structures on simplicial surfaces and hyperbolic volume. *Annals of mathematics* 139, 3 (1994), 553–580.

Rohan Sawhney and Keenan Crane. 2017. Boundary First Flattening. *ACM Trans. Graph.* 37, 1 (2017).

Patrick Schmidt, Marcel Campen, Janis Born, and Leif Kobbelt. 2020. Inter-Surface Maps via Constant-Curvature Metrics. *ACM Transactions on Graphics* 39, 4 (2020).

Nicholas Sharp and Keenan Crane. 2020. A Laplacian for nonmanifold Triangle Meshes. *Computer Graphics Forum* 39, 5 (2020), 69–80.

Nicholas Sharp, Yousuf Soliman, and Keenan Crane. 2019. Navigating intrinsic triangulations. *ACM Transactions on Graphics* 38, 4 (2019), 1–16.

Yousuf Soliman, Dejan Slepčev, and Keenan Crane. 2018. Optimal cone singularities for conformal flattening. *ACM Transactions on Graphics* 37, 4 (2018), 1–17.

Boris Springborn. 2020. Ideal Hyperbolic Polyhedra and Discrete Uniformization. *Discrete & Computational Geometry* 64, 1 (2020), 63–108.

Boris Springborn, Peter Schröder, and Ulrich Pinkall. 2008. Conformal Equivalence of Triangle Meshes. *ACM Transactions on Graphics* 27, 3 (2008), 1–11.

Jian Sun, Tianqi Wu, Xianfeng Gu, and Feng Luo. 2015. Discrete conformal deformation: algorithm and experiments. *SIAM Journal on Imaging Sciences* 8, 3 (2015), 1421–1456.

Jeffrey R Weeks. 1993. Convex hulls and isometries of cusped hyperbolic 3-manifolds. *Topology and its Applications* 52, 2 (1993), 127–149.

Tianqi Wu. 2014. *Finiteness of Switches in discrete Yamabe flow.* Master's thesis. Tsinghua University.

Denis Zorin. 2021. *Convergence Analysis of the Algorithm in "Efficient and Robust Discrete Conformal Equivalence with Boundary".* arXiv:2109.03436 [math.NA]

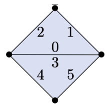
A PROOFS AND ADDITIONAL LEMMAS

Proof of Proposition 2

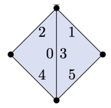
Proof. If x is not fixed, by the well-definedness of R on mesh elements, for each $h \in x$ we have $R(h) \notin x$. Therefore for a non-fixed individual face or edge x all its halfedges can be assigned to H^1 (or to H^2) without contradicting the conditions. It needs to be shown that this can be done for all such elements consistently.

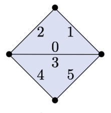
Let H^e the set of halfedges whose edges are not fixed and H^f the set of halfedges whose faces are not fixed. Let Q the relation that is the union of $O|_{H^e}$ and $\mathcal{N}|_{H^f}$ on $H \setminus H^s$. Consider the connected components H_i of Q (intuitively: the mesh's connected components separated by fixed edges and fixed faces). Due to the properties of R

Table 1. Combinatorial updates required to perform symmetric flips of all relevant consistent types. The change to $\mathcal{N}$ is given by listing the orbits (halfedge cycles forming faces) of $\mathcal{N}$ created by the flip. The employed indexing is depicted in the figures left and right. Similarly, we define changes to R viewing it as a permutation with orbits of length 1 or 2, and listing the sets of orbits being replaced. Finally, rather than deleting and adding new halfedges on demand, for implementational efficiency we can associate a superfluous pair of halfedges, eliminated by a quad-creating flip, with the quad (listed behind the bar).

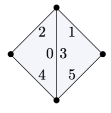


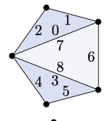
$(1,1,1)+(2,2,2) \quad \leftrightarrow \quad (1,1,1)+(2,2,2)$	
$\mathcal{N}: (h^i_0, h^i_1, h^i_2),\ (h^i_3, h^i_4, h^i_5),\ i=1,2$	$\mathcal{N}: (h^i_0, h^i_2, h^i_4),\ (h^i_1, h^i_3, h^i_5),\ i=1,2$
R : unchanged	R : unchanged



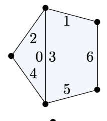


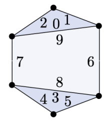
$(1,\parallel,2) \quad \leftrightarrow \quad (t,\perp,t)$	
$\mathcal{N}: (h_0,h_1,h_2),\ (h_3,h_4,h_5)$	$\mathcal{N}: (h_0,h_2,h_4),\ (h_1,h_3,h_5)$
$R: (h_0,h_3)$	$R: (h_0),\ (h_3)$



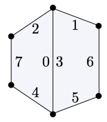


$(1,1,t)+(2,2,t) \quad \leftrightarrow \quad (t,\perp,q)$	
$\mathcal{N}: (h_0,h_1,h_2),\ (h_3,h_4,h_5), (h_6,h_7,h_8)$	$\mathcal{N}: (h_0,h_2,h_4),\ (h_1,h_3,h_5,h_6)\ \mid\ h_7,h_8$
$R: (h_0,h_3),\ (h_7,h_8)$	$R: (h_0),\ (h_3)$





$(1,1,q)+(2,2,q) \quad \leftrightarrow \quad (q,\perp,q)$	
$\mathcal{N}: (h_0,h_1,h_2),\ (h_3,h_4,h_5), (h_6,h_9,h_7,h_8)$	$\mathcal{N}: (h_0,h_2,h_7,h_4),\ (h_1,h_3,h_5,h_6)\ \mid\ h_8,h_9$
$R: (h_0,h_3),\ (h_8,h_9)$	$R: (h_0),\ (h_3)$



(preserving/inverting O and $\mathcal{N}$) it is well-defined on these connected components via $R(H_i) = H_j \Leftrightarrow R(h) \in H_j$ for any $h \in H_i$. Using arguments analogous to [Panozzo et al. 2012, Prop. 2] one verifies that the set of fixed elements necessarily forms a cycle; therefore there are at least two such connected components.

As R on $H \setminus H^s$ has orbits of length 2 only, it allows a bipartition of the connected components, i.e., they can be assigned to two sets H^1 and H^2 in accordance with the above conditions. □

Label Compatibility

PROPOSITION 3.

(a) $e \in E^\perp \Rightarrow f_a, f_b \in F^s$.
(b) $e \in E^\parallel \Rightarrow f_a \in F^1, f_b \in F^2$ *or* $f_a = f_b \in F^s$.
(c) $e \in E^1 \Rightarrow f \notin F^2,\ e \in E^2 \Rightarrow f \notin F^1$.
(d) $e \in E^i, f_a, f_b \in F^s \Rightarrow R(e) \in f_a, f_b$.

PROOF. Part (a) follows immediately from the definition of F^i, as faces from F^i cannot have edges from $E^\perp$.

Suppose a face f_a is incident at an edge e from $E^\parallel$. For these edges $R(e) = e$. Suppose $f_a \in F^1$, then $R(f_a)$ is incident to $R(e) = e$, therefore $f_b = R(f_a)$. As $R(f_a) \in F^2$ by definition of F^2, this proves the first part of (b). Suppose $f_a \in F^s$, and let h a halfedge $h \in e$, $h \in f_a$. Then $R(h) \in f_a$ by the definition of F^s; but, by definition of $E^\parallel$, $R(h) \in e$, so $f_a = f_b$, i.e., a face is adjacent to itself along e.

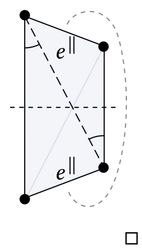
Part (c) directly follows from the definitions of E^i and F^i.

In part (d), suppose f_a and f_b are incident at $e \in E^1$, $f_a, f_b \in F^s$, and $e = (h_a, h_b)$. Then $R(h_a) \in R(f_a) = f_a$, $R(h_b) \in f_b$, and $O(R(h_a)) = R(h_b)$ by the properties of R, i.e., $(R(h_a), R(h_b))$ is an edge. By definition of E^i, it has to be in E^2, i.e., faces f_a and f_b share a second edge, and this edge is from E^2. □

Irrelevance of Flip Types $(s,\parallel,s)$ and $(s,1,s)$

PROPOSITION 4. *Types $(t,\parallel,t)$, $(q,\parallel,q)$, $(t,1,t)$, $(t,1,q)$, and $(q,1,q)$ are associated with edges that are Delaunay regardless of metric.*

PROOF. Consider $(t,\parallel,t)$. By Prop. 3(b), it corresponds to a configuration with a single face: $(f^t, e^\parallel, f^t)$. As the triangle f^t is isosceles, and both side edges of the triangle coincide with $e^\parallel$, angles opposite $e^\parallel$ are $\pi/2 - \alpha/2$ if the apex angle is α, i.e., their sum is guaranteed to be less than π and the edge is Delaunay. For $(q,\parallel,q)$, to evaluate the Delaunay criterion, we split f^q into triangles. As f^q is inscribed the choice of diagonal does not affect the angles; we can choose the diagonal that connects a vertex of $e^\parallel$ with a vertex with trapezoid angles $\leq \pi/2$ (see inset), from which we can see that both angles opposite $e^\parallel$ are less than $\pi/2$. For cases $(t,1,t)$, $(t,1,q)$, and $(q,1,q)$ the same logic applies to each face incident at the shared edge e^1.



□

B DOUBLE COVER: FORMAL DEFINITION

Given a mesh $N = (H^0, \mathcal{N}^0, O^0)$, with boundary and interior halfedges $H^{bnd} \cup H^{int} = H^0$, we discard H^{bnd} and set $H = H^1 \cup H^2$ where $H^1 = H^{int}$ and $H^2 = \bar{H}^{int}$, where $\bar{\cdot}$ denotes a copy. The reflection map R is defined via $R(h) := h'$ if $h' \in H^2$ is the copy of $h \in H^1$. O^0 is adopted on both copies to define O, except that $O(h) := R(h)$ if $O^0(h) \in H^{bnd}$; this latter adjustment constitutes the *gluing* of the two copies along their boundaries. Finally

$$\mathcal{N}(h) := \begin{cases} \mathcal{N}^0(h) & \text{if } h \in H^1, \\ R((\mathcal{N}^0)^{-1}(R(h))) & \text{if } h \in H^2. \end{cases}$$

This forms the symmetric double cover mesh $M = (H, \mathcal{N}, O, R)$ with triangle faces and map R. Note that R is a reflection map: it satisfies the conditions of Def. 5 (where condition (3) is void as M has no boundary). It is easy to see that this construction implies $E^\perp = \varnothing$ and $F^s = \varnothing$, i.e., no element crosses the symmetry line (the former boundary). $E^\parallel$ contains the edges lying *on* the symmetry line, i.e., those for whose halfedges the O relation was adjusted to glue the two copies.