The Shape Space of Discrete Orthogonal Geodesic Nets

MICHAEL RABINOVICH, ETH Zurich, Switzerland
TIM HOFFMANN, TU Munich
OLGA SORKINE-HORNUNG, ETH Zurich, Switzerland

Fig. 1. We develop a theory for shape space exploration of DOG nets, which are discrete developable surfaces parameterized by orthogonal geodesics. Our study results in a characterization of this shape space and an algorithm to discretize smooth flows on DOGs, as used in the above animation. The first three frames were created using our editing system, and the rest were generated using a *curve-constraining flow*, deforming the surface by constraining a curve that lies on it.

Discrete orthogonal geodesic nets (DOGs) are a quad mesh analogue of developable surfaces. In this work we study continuous deformations on these discrete objects. Our main theoretical contribution is the characterization of the shape space of DOGs for a given net connectivity. We show that generally, this space is locally a manifold of a fixed dimension, apart from a set of singularities, implying that DOGs are continuously deformable. Smooth flows can be constructed by a smooth choice of vectors on the manifold's tangent spaces, selected to minimize a desired objective function under a given metric. We show how to compute such vectors by solving a linear system, and we use our findings to devise a geometrically meaningful way to handle singular points. We base our shape space metric on a novel DOG Laplacian operator, which is proved to converge under sampling of an analytical orthogonal geodesic net. We further show how to extend the shape space of DOGs by supporting creases and curved folds and apply the developed tools in an editing system for developable surfaces that supports arbitrary bending, stretching, cutting, (curved) folds, as well as smoothing and subdivision operations.

CCS Concepts: • **Computing methodologies** → **Mesh models**; **Mesh geometry models**;

Additional Key Words and Phrases: developable surfaces, discrete differential geometry, geodesic nets, shape modeling, shape space

ACM Reference Format:
Michael Rabinovich, Tim Hoffmann, and Olga Sorkine-Hornung. 2018. The Shape Space of Discrete Orthogonal Geodesic Nets. *ACM Trans. Graph.* 37, 6, Article 228 (November 2018), 17 pages. https://doi.org/10.1145/3272127.3275088

Authors' addresses: Michael Rabinovich, ETH Zurich, Department of Computer Science, Universitätstrasse 6, Zurich, 8092, Switzerland; Tim Hoffmann, TU Munich, Department of Mathematics; Olga Sorkine-Hornung, ETH Zurich, Department of Computer Science, Universitätstrasse 6, Zurich, 8092, Switzerland.

0730-0301/2018/11-ART228
https://doi.org/10.1145/3272127.3275088

1 INTRODUCTION

A developable surface is easy to fabricate, but hard to design. Developable shapes are simple to construct by affordable and effective fabrication methods using a variety of materials, for example by bending flat sheets of paper or metal, or by cylindrical CNC milling [Harik et al. 2013], contributing to their prominence in product engineering and architecture. A developable surface is difficult to design since its geometry is highly constrained; at the same time, it admits a rich set of extrinsic and intrinsic deformations. Applying standard modeling tools such as freeform space deformations or elastic surface-based bending quickly violates the developability property (local isometry to the plane). Fully and freely exploring the shape space of developable surfaces without risking leaving it has proven challenging, and existing methods usually only cover a limited set of extrinsic deformations due to using ruling based representations [Liu et al. 2006; Tang et al. 2016] or do not cover intrinsic deformations because they model isometries [Burgoon et al. 2006]. In practice, developable geometry is often created by working with simple primitives like cylinders and cones. Our goal in this paper is to provide the necessary theoretical and algorithmic basis for a more complete and unhindered exploration of the developable shape space, in hopes of facilitating easier and more effective design processes.

We rely on the recent work of Rabinovich et al. [2018] for a discrete model of developable surfaces. This model represents developable shapes by discrete orthogonal geodesic nets (DOGs). The existence of orthogonal geodesic parameterization in the smooth case is equivalent to the developability property, and a DOG net models this as a quadrilateral mesh with simple local angle constraints: all angles around a vertex are equal. This discrete model avoids dependence on rulings or a fixed isometric reference shape, and is hence our foundation for modeling and exploring discrete developable shape spaces.

The focus of [Rabinovich et al. 2018] is the discrete DOG model itself rather than its deformations. In this paper, we develop the essential tools to effectively navigate the shape space of plausible DOGs. To do so, we discretize *continuous* deformations on these

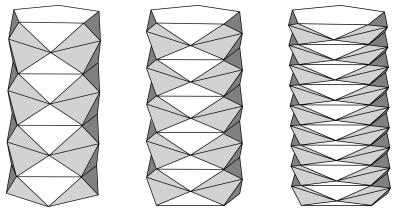


Fig. 2. Different sampling rates of the Schwarz Lantern, a classic example of pointwise convergence of the surface without convergence of its cotan Laplacian normals [Wardetzky 2007]. Our novel DOG Laplacian operator converges under sampling of any analytical orthogonal geodesic net, including a smooth Schwartz Lantern. For clarity, we display the triangular mesh, but in reality such a sampling consists solely of quadrilaterals.

discrete surfaces. More precisely, assume F^0 is a given DOG net with fixed mesh connectivity, and $\mathcal{M}$ is the set of all DOGs with the same connectivity, i.e., its *shape space*. We are interested in finding a continuous function $\mathcal{F}(t)$ such that $\mathcal{F}(0) = F^0$ and $\mathcal{F}(t) \in \mathcal{M}$ for every t; $\mathcal{F}$ is referred to as a (constrained) DOG flow. The mere existence of such flows depends on the geometry of the shape space $\mathcal{M}$, which is our primary object of study. We start with the theoretical backbone: we prove that such flows generally exist, and we count exactly how many. Our key technique is analyzing the linear dependence of the gradients of the angle constraints by characterizing a *constraints graph* of minimally dependent gradients using a few geometric observations. We show that the shape space $\mathcal{M}$ is generally a manifold of a fixed dimension k, thus implying the existence of k linearly independent DOG flows. We also show that $\mathcal{M}$ is not a manifold globally and contains a set of singular points. We then use our theory to discretize various flows on DOGs by projecting the gradients of the objective functions onto the tangent space of $\mathcal{M}$, and we devise a strategy to handle the singular points of the shape space. In addition, we address an explicit shortcoming stated in [Rabinovich et al. 2018] and extend the shape space $\mathcal{M}$ to support sharp creases and curved folds.

An essential ingredient necessary to design meaningful flows on the shape space is a Laplacian operator and a discrete mean curvature. We derive these fundamental notions specifically for DOG nets and show that they converge under sampling of an analytical net. In particular, our DOG Laplacian possesses the desired properties of a discrete Laplace operator: it is symmetric, positive semidefinite and linearly precise.

To showcase the developed toolset, we derive a set of objective functions whose constrained gradient can be used to flow towards a desired shape. We demonstrate results with Willmore flow and curve-constraining flow. We apply our theoretical results in an editing system for developable surfaces that supports bending, creasing and stretching in arbitrary directions, on various topologies, and also includes smoothing and subdivision operations, all staying within the shape space.

2 RELATED WORK

2.1 Developable surfaces and their deformations

A surface is developable if it is locally isometric to a planar patch. If the surface is C^2, developability can be equivalently defined by vanishing Gaussian curvature [do Carmo 1976], the surface being ruled with constant normals along the rulings [Pottmann and Wallner 2001], or being locally parameterizable by orthogonal geodesics [Rabinovich et al. 2018]. A non-planar developable surface is locally characterized by the directions of its rulings: it is cylindrical if the rulings are parallel, conical if they all meet at a point, and otherwise the rulings meet in a curve and the surface is a tangent developable surface [do Carmo 1976]. These are all called *torsal patches*. A general developable surface is a composition of a possibly infinite number of planar and torsal patches [Pottmann and Wallner 2001]. Deforming a C^2 developable surface by an isometry preserves its vanishing Gaussian curvature, but there is also a multitude of non-isometric deformations that keep the surface developable. It can be shown that deforming a curve on a C^2 developable surface locally determines the shape of the entire deformed surface. One can for instance deform a curvature line [Liu et al. 2006], or a curve with a prescribed geodesic curvature [Graustein 1917; Fuchs and Tabachnikov 1999] and locally construct the surface from its rulings.

One can look at a larger yet still highly structured set of developable surfaces by permitting the surface to be piecewise C^2 and C^0 along a finite number of curves. Straight "origami like" folds are C^0 creases through geodesics, and are widely studied in the field of computational origami [Demaine and O'Rourke 2007]. Any shape created by repeated application of these folds is piecewise planar [Demaine et al. 2011a]. Curved folds [Huffman 1976] are C^0 creases shaped as arbitrary curves. Unlike folds along geodesics, applying curved folds to a piece of paper does generally allow for further bending of the surface in-between, however one can show that bending a surface on one side of the fold locally determines the shape of the other side [Kilian et al. 2008]. Hence, while C^2 deformations are typically studied in differential geometry and origami folds are more combinatorial in nature, the relatively newer study of curved folds stands in a junction between both [Demaine et al. 2011b].

2.2 Modeling with developable surfaces

Several works in geometry processing model smooth developable surfaces by explicitly representing their rulings, which can be seen as discrete models of developable surfaces parameterized by conjugate nets. The works of [Liu et al. 2006; Kilian et al. 2008] model developable surfaces as collections of connected torsal patches, each modeled as a planar quad strip, while the work of [Stein et al. 2018] extends this view to triangle meshes. Tang et al. [2016] model smooth torsal surfaces as developable splines formed by connecting two Bézier curves by rulings with a constant normal. As explained in [Tang et al. 2016; Rabinovich et al. 2018], these methods can only model a subset of the deformations that keep a surface developable. Solomon et al. [2012] propose an origami based editing system for developable surfaces, where folding along multiple straight creases approximates the modeling of smooth developable shapes. Similarly to the other works mentioned above, this system is also dependent on explicit rulings.

Physically based simulation methods such as [Burgoon et al. 2006; Fröhlich and Botsch 2011; Narain et al. 2013] model developable shapes as objects made of inextensible material (such as paper), so the attainable deformations are restricted to isometries of a fixed reference surface. As mentioned, there are many more deformations that keep a surface developable besides isometries. The work of [Rabinovich et al. 2018] avoids these limitations by discretizing orthogonal geodesic nets (DOGs). Our work extends the theory of DOGs by studying continuous deformations on these surfaces, discretizing them and further deriving objective functions that are useful for editing. In terms of applications, we demonstrate superior quality and a significantly wider range of deformations compared to [Rabinovich et al. 2018].

Modeling smooth developable surfaces is complemented by works on folds and curved folding. Tachi [2009] presents a rigid origami simulator, while the works of [Kilian et al. 2008; Tang et al. 2016] model curved folds on discrete conjugate developable nets. Mitani and Igarashi [2011] generate objects with curved folds by iterated reflections. In [Kilian et al. 2017] the authors develop an algorithm to fold a crease pattern by pulling on a network of strings. These works either focus solely on isometric deformations, or are limited by the ruling based representation to only a subset of possible bending deformations. By incorporating creases and curved folds into the theory of DOGs, we are able to create the first editing system that supports bending in all directions, stretching developable deformations, and creases and curved folds, all in a unified framework.

2.3 Shape space and flows

Shape space exploration is a framework for treating shapes in the setting of Riemannian geometry, where surfaces are seen as points in a high dimensional space endowed with a metric. The work of [Kilian et al. 2007] investigated the space of triangulated surfaces, computing an as-isometric-as-possible interpolation as a geodesic between two meshes w.r.t a Riemannian metric that measures stretch of triangle edges. Heeren et al. [2014] consider the shape space of shells, showing how to shoot geodesics with prescribed initial data and providing a construction for parallel transport. This enables natural extrapolation of paths in shell space and the transfer of large nonlinear deformations from one shell to another. In [Heeren et al. 2016], the authors extend the concept of Euclidean splines to the Riemannian manifold of discrete shells, allowing for a temporally smooth interpolation of a given set of shell keyframe poses.

Most relevant for us is the work of [Yang et al. 2011], presenting a computational framework to locally characterize *constrained* shape spaces of meshes, implicitly described by a collection of nonlinear constraints. This work enables accessing the high dimension shape space through first and second order approximates, namely tangent spaces and quadratic osculating surfaces, and demonstrates applications to local shape space explorations of conjugate and circular nets for a fixed connectivity. Computing tangent directions on the space requires an expensive SVD decomposition due to possible linear dependency in the constraints. Our work instead focuses on studying the shape space of DOG net constraints. We analyze the linear dependencies of its constraints and show that in general, the constraints Jacobian has full row rank, and the shape space of a given connectivity is generically a manifold of a fixed dimension, apart from a set of singularities. We further give a simple strategy to detect and handle the singularities, avoiding costly operations to remove the linear dependency. As opposed to [Yang et al. 2011], where the Euclidean L^2 metric is used, we employ a metric based on a discrete Laplacian, inspired by the works of [Eckstein et al. 2007; Sundaramoorthi et al. 2007], as well as a bending energy as an objective, both tailored specifically for DOG nets. We show how using the Laplacian as a metric is beneficial over L^2, consistently with the observations in [Eckstein et al. 2007]. A consequence of the DOG shape space being generally a smooth manifold is the existence of smooth flows, and our work is also inspired by geometry processing literature on flows for smoothing and shape interpolation, see e.g. [Desbrun et al. 1999; Eckstein et al. 2007; Kazhdan et al. 2012; Crane et al. 2013]. Local-global solvers [Sorkine and Alexa 2007; Bouaziz et al. 2012, 2014; Peng et al. 2018] are common approaches in geometry processing to handle constrained objectives, however they do not discretize such flows, nor do they guarantee to minimize an objective and stay within the shape space due to solving the constraints in a least-squares manner. Our theory shows the existence of exact DOG flows minimizing objectives, to which our discretization converges as the time step goes to zero (Fig. 10).

2.4 Discrete Laplace operator

The theory and applications of discrete Laplacians on triangulated surfaces are well developed, and the celebrated cotan operator [Pinkall and Polthier 1993] is probably the most prominent representative. In [Wardetzky et al. 2007], the authors describe a set of natural properties for discrete Laplacians, inspired by the smooth setting, and prove an important theoretical limitation: discrete Laplacians on triangle meshes cannot satisfy all natural properties. The famous Schwarz Lantern mesh constitutes a quite general example of pointwise convergence of the surface without convergence of its cotan Laplacian normals [Wardetzky 2007].

Far less is known about quadrilateral meshes or the more general polygonal case. A notable polyhedral Laplacian is developed in [Alexa and Wardetzky 2011]. In this work, we derive a Laplacian operator tailored to quadrilateral DOG nets, the so-called DOG Laplacian, guided by the natural properties enumerated in [Wardetzky et al. 2007], such as symmetry, positive semidefinitiveness and linear precision. We show a strong connection between the normals induced by the DOG Laplacian and the DOG Gauss map as defined in [Rabinovich et al. 2018], and we prove the convergence of both quantities under sampling of a smooth analytical orthogonal geodesic net. In particular, when sampling along the infamous smooth Schwartz Lantern net, the DOG Laplacian converges (Fig. 2). While the polygonal Laplacian of [Alexa and Wardetzky 2011] is based on the vector area of polygons, we derive the DOG Laplacian based on a different notion of area, assuming the underlying mesh is a DOG net. This assumption is a fundamental difference between the two Laplacians, allowing us to derive an operator that is less general but converges under sampling of an analytical orthogonal geodesic net.

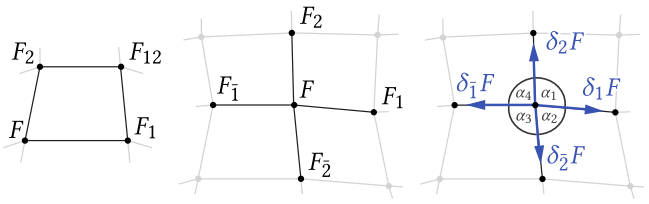


Fig. 3. The shift notation on a quad (left) and a star (center); edge directions and star angles (right), which on a DOG satisfy $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$. Illustration taken from [Rabinovich et al. 2018].

3 NOTATIONS AND SETUP

Following the notation in [Rabinovich et al. 2018], we denote continuous maps in lowercase letters and their discrete equivalents by uppercase. The notation $f(x, y) : U \to \mathbb{R}^3$, where $U \subseteq \mathbb{R}^2$, refers to a (local) regular parameterization of a smooth surface, and $n(x, y) : U \to \mathcal{S}^2$ is its normal map. A natural discrete analogy for a local parameterization f is a map $F : V \to \mathbb{R}^3$, where $V \subseteq \mathbb{Z}^2$. We refer to F as our discrete net, and likewise $N : V \to \mathcal{S}^2$ denotes our discrete Gauss map, $A : V \to \mathbb{R}$ is a notion of a vertex area. We represent our discrete nets as pure quad grid meshes, where the valence of every inner vertex is 4. We refer to an inner vertex, its four neighbors and its four emanating edges as a *star*.

We denote the number of vertices in the net by $|F|$ and the total number of vertex coordinates by $n = 3|F|$. The positions of individual vertices can be referenced by their two indices in the grid: $F(j, h) \in \mathbb{R}^3$, or simply by a single absolute vertex index: $F(1), F(2), \ldots, F(|F|) \in \mathbb{R}^3$. The vector of all vertex coordinates is denoted $\mathbf{F} = (F(1), \ldots, F(|F|) \in \mathbb{R}^n$. Vertices can also be referenced by relative indices using the shift notation, as is customary in discrete differential geometry. Slightly abusing notation, a individual vertex is then called simply F, and vertices in its vicinity are referenced using lower shift indices:

$$F = F(j, h),\ F_1 = F(j + 1, h),\ F_2 = F(j, h + 1),$$
$$F_{12} = F(j + 1, h + 1),\ F_{\bar{1}} = F(j - 1, h),\ F_{\bar{2}} = F(j, h - 1),$$

where $j, h \in \mathbb{Z}$, i.e., the lower index denotes the coordinate number to shift, and a bar above it indicates a negative shift (see Fig. 3). The unit-length directions of edges emanating from a point F are denoted as $\delta_1 F,\ \delta_2 F,\ \delta_{\bar{1}} F,\ \delta_{\bar{2}} F$, i.e.,

$$\delta_1 F = (F_1 - F)/\|F_1 - F\|, \quad \delta_{\bar{1}} F = (F_{\bar{1}} - F)/\|F_{\bar{1}} - F\|,$$
$$\delta_2 F = (F_2 - F)/\|F_2 - F\|, \quad \delta_{\bar{2}} F = (F_{\bar{2}} - F)/\|F_{\bar{2}} - F\|.$$

We denote the inner angles around a star at F as α_j, ordered consecutively clockwise (see Fig. 3). We assume our net is a discrete immersion, which means that the edge directions $\delta_i F, \delta_{\bar{i}} F,\ i = 1, 2$, are distinct. We denote the angles of the coordinate curves by $\beta_1 F, \beta_2 F$:

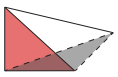
$$\cos(\beta_1 F) = \langle \delta_1 F, \delta_{\bar{1}} F \rangle, \quad \cos(\beta_2 F) = \langle \delta_2 F, \delta_{\bar{2}} F \rangle.$$

See Fig. 4 for an illustration.

4 THE SPACE OF DISCRETE ORTHOGONAL GEODESIC NETS

Throughout the paper, we assume that the connectivity of a DOG net is a subset of $\mathbb{Z}^2$. This includes topologies created by cutting holes in a disc, but no cylindrical topologies. The quads of a DOG net are not necessarily planar; however, our analysis requires a reasonable assumption on their shape and distance from planarity, summarized below.

Definition 4.1. A net F is *regular* if its quads are non-degenerate. It is *proper* if it is regular and the planes of the tetrahedrons formed by the quads are not orthogonal (see the planes formed by the pink and gray triangles in the inset).



The set of proper nets is an open set, and nets that are not proper are usually not interesting for modeling purposes. Let F^0 be a proper discrete orthogonal geodesic net, and let $\mathcal{M}(F^0)$ be the set of proper orthogonal geodesic nets with the same connectivity as F^0. Since we often discuss the shape space of a specific, fixed mesh connectivity, we usually abbreviate $\mathcal{M}(F^0)$ simply as $\mathcal{M}$. In the following, we develop a theory for exploring the shape space $\mathcal{M}$ that yields the following results:

(1) The space $\mathcal{M}$ is locally a smooth manifold of dimension $k = O(|\partial F|)$, apart from a scarce set of singular points (∂F is the set of boundary vertices of F). This implies the existence of k linearly independent smooth flows on non singular DOG nets.
(2) We show how to discretize these flows, which amounts to solving a linear system to compute tangents on the shape space manifold $\mathcal{M}$. We prove that these systems are full rank.
(3) We analyze the singular points of $\mathcal{M}$, which are not locally manifold, and suggest a strategy to compute a discrete flow on the shape space that is computationally cheap, based on leveraging close-by points on nearby manifolds.

4.1 Angle constraints of a DOG

A discrete orthogonal geodesic net F is a quad net where the angles around every vertex are equal (see Fig. 3). We can write the cosine of the angles as a function of the vertex coordinates, e.g., $\cos(\alpha_1) = \langle \delta_1 F, \delta_2 F \rangle$. For an inner vertex, the condition $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$ can be specified by three constraints of the form $\phi_i(\mathbf{F}) = 0$:

$$\begin{aligned} \phi_1(\mathbf{F}) &:= \cos(\alpha_1) - \cos(\alpha_2) = 0, \\ \phi_2(\mathbf{F}) &:= \cos(\alpha_2) - \cos(\alpha_3) = 0, \\ \phi_3(\mathbf{F}) &:= \cos(\alpha_3) - \cos(\alpha_4) = 0. \end{aligned} \tag{1}$$

Note that these three constraints also imply $\cos(\alpha_4) = \cos(\alpha_1)$. The constraints for boundary vertices with p neighbors require $p - 2$ angle equality equations. Hence, the total number of constraints is $m = 3\,I + \sum_{j \in \partial F} (|\mathcal{N}(j)| - 2)$, where I is the number of inner vertices and $\mathcal{N}(j)$ denotes the set of neighbors of net vertex j.

4.2 The shape space of a single orthogonal geodesic star

From a local point of view, the DOG constraints are quite simple. Let $\mathcal{M}_s$ be the set of single proper DOG geodesic stars, containing a central vertex and its 4 neighbors, such that all angles around the central vertex are equal.

Theorem 4.2. *$\mathcal{M}_s$ is a 12-dimensional manifold and can be parameterized by the star's edge lengths l_1, l_2, l_3, l_4, its two coordinate polygons angles β_1, β_2 and additional 6 parameters accounting for rigid motions in $\mathbb{R}^3$.*

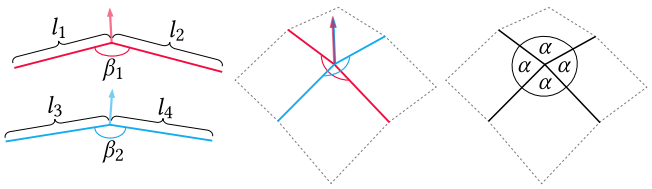


Fig. 4. Parameterization of a DOG star. By Theorem 4.2, a single proper DOG star is determined up to rigid motion by its 4 edge lengths l_j, $j = 1..4$ and 2 curve coordinate angles β_1, β_2 (left). Reconstructing the star from these parameters requires translating and rotating one of the discrete curves around the other such that their normals match and the discrete Frenet frames are overlaid (center). This guarantees that all angles around the star are equal; the value of the angle α is completely determined by β_1, β_2.

Proof. See Appendix A. □

The parameters are illustrated in Fig. 4. It follows that one can flow or smoothly deform a single DOG star by smoothly deforming these 12 parameters. As we usually work with vertex coordinates, it is useful to see that $\mathcal{M}_s$ can be represented by $3 \times 5 = 15$ vertex coordinates satisfying the 3 linearly independent constraints in Eq. (1), i.e., we can view $\mathcal{M}_s$ as a 12-dimensional manifold embedded in a 15-dimensional ambient space.

4.3 Shape space through linear dependence of constraint gradients

A general DOG net is composed of multiple connected stars, resulting in a more complicated shape space, as angle constraints of different vertices can be dependent. We analyze this shape space by studying linear dependencies between the gradients of the constraints w.r.t. vertex coordinates, $\frac{\partial \phi_i}{\partial \mathbf{F}}$. Let $C \subset \mathbb{R}^n$ be the open set of admissible variables representing vertex coordinates of proper nets ($n = 3|F|$). Given the m smooth constraint functions for F to be a DOG net, $\phi_i : C \to \mathbb{R}$, $i = 1, \ldots, m$, let $\mathcal{M}$ be the set of variables $\mathbf{F} \in C$ satisfying all the constraints: $\mathcal{M} = \{\mathbf{F} \in C \mid \phi_i(\mathbf{F}) = 0 \ \forall i = 1, \ldots, m\}$. We write $\phi(\mathbf{F}) := (\phi_1(\mathbf{F}), \ldots, \phi_m(\mathbf{F}))^\mathsf{T}$, and $J_F = \frac{\partial \phi}{\partial \mathbf{F}}$ denotes the $m \times n$ constraint Jacobian matrix. By the constant rank theorem, the following holds:

Theorem 4.3. Constraint manifold. *Let $F^0 \in \mathcal{M}$. If the rows of J_{F^0} are linearly independent, then in a neighborhood of F^0 the set $\mathcal{M}$ is a manifold of dimension $k = n - m$. At that point, the tangent space of the manifold, $T\mathcal{M}$, is orthogonal to the image space of J_{F^0}, and is exactly the set of solutions to $J_{F^0} x = 0$.*

In this case a given DOG net $F^0 \in \mathcal{M}$ can be smoothly deformed by a smooth *constrained flow*, i.e., there are smooth functions $\mathcal{F}(t)$, where $\mathcal{F}(0) = F^0$ and $\mathcal{F}(t) \in \mathcal{M}$. The amount of inherently different flows, or linearly independent flow directions, depends on the dimension of the tangent space $T\mathcal{M}$, since each smooth flow is a smooth choice of coefficients for some basis of $T\mathcal{M}$.

Let us consider the smallest DOG net containing an inner vertex as an example, namely a DOG net F with a 3×3 grid connectivity, as shown in Fig. 5. We denote the inner vertex position as F and the other eight vertices are named using the shift notation. There are 7 angle constraints, defined as $\cos(\alpha_{j+1}) - \cos(\alpha_j)$ for

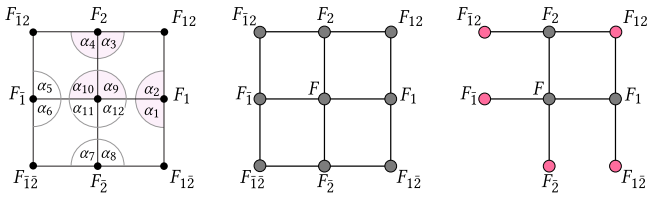


Fig. 5. A DOG net F with a 3×3 grid connectivity graph $\mathcal{G}$ and a constraints graph $\mathcal{G}_\mathcal{I}$. The notation for the angles in this example is chosen such that adjacent constrained angles have consecutive indices. On the right we show the constraints graph $\mathcal{G}_\mathcal{I}$ for $\mathcal{I} = \{1, 2, 5\}$, i.e., for the three constraints: $\phi_1 := \cos(\alpha_2) - \cos(\alpha_1)$, $\phi_2 := \cos(\alpha_4) - \cos(\alpha_3)$, $\phi_5 := \cos(\alpha_{10}) - \cos(\alpha_9)$, marked in faint pink on the left. Leaf vertices and corner vertices of $\mathcal{G}_\mathcal{I}$ are colored in pink.

$j = 1, 3, 5, 7, 9, 10, 11$, and numbered as ϕ_i, $i = 1, \ldots, 7$. Note that we do not include a constraint for the equality of α_{12} and α_9, as this is guaranteed by the other 3 constraints around that star. In summary, there are $m = 7$ constraints, $n = 3 \times 9 = 27$ variables representing the 3D coordinates of the 9 vertices, and J_F is a 7×27 matrix of the form

$$J_F = \begin{pmatrix} \frac{\partial \phi_1}{\partial \mathbf{F}} \\ \frac{\partial \phi_2}{\partial \mathbf{F}} \\ \vdots \\ \frac{\partial \phi_7}{\partial \mathbf{F}} \end{pmatrix} = \begin{pmatrix} \frac{\partial \phi_1}{\partial F_{\bar{1}\bar{2}}} & \frac{\partial \phi_1}{\partial F_{\bar{2}}} \cdots \frac{\partial \phi_1}{\partial F_{12}} \\ \frac{\partial \phi_2}{\partial F_{\bar{1}\bar{2}}} & \frac{\partial \phi_2}{\partial F_{\bar{2}}} \cdots \frac{\partial \phi_2}{\partial F_{12}} \\ & \vdots \\ \frac{\partial \phi_7}{\partial F_{\bar{1}\bar{2}}} & \frac{\partial \phi_7}{\partial F_{\bar{2}}} \cdots \frac{\partial \phi_7}{\partial F_{12}} \end{pmatrix}. \tag{2}$$

Note that $\frac{\partial \phi_i}{\partial F_{\bar{1}\bar{2}}}$, $\frac{\partial \phi_i}{\partial F_{\bar{2}}}$ etc. are row vectors of dimension 3. It is convenient to write them as 3-vectors since the constraints are of the form $\phi_i(\mathbf{F}) = \cos(\alpha_{j_1}) - \cos(\alpha_{j_2})$ and the cosine angle gradients have a simple geometric meaning:

Lemma 4.4. *We use the angle notation of Fig. 3. If the net is proper then $\frac{\partial \cos(\alpha_9)}{\partial F_1} = w \neq 0$, where w is a vector in the plane spanned by F, F_1, F_2 and $w \perp (F - F_1)$.*

Proof. See Appendix B. □

By symmetry, this is also true for other angles and neighboring vertices in other directions, i.e., one can replace the angle α_9 and the vertices F_1, F_2 in the claim by e.g. $\alpha_{11}, F_{\bar{1}}, F_{\bar{2}}$.

As a result of Theorem 4.2, Theorem 4.3 and Lemma 4.4 we get the following:

Theorem 4.5. *The constraints in Eq.* (1) *around a proper DOG star are linearly independent.*

Proof. A DOG star contains 5 vertices in $\mathbb{R}^3$. Let $\mathcal{M}^i \subset \mathbb{R}^{15}$ be the set of proper star vertex coordinates $\mathbf{F}$ satisfying $\phi_i(\mathbf{F}) = 0$. Each constraint gradient $\frac{\partial \phi_i}{\partial \mathbf{F}}$ is non-vanishing, meaning $\frac{\partial \phi_i}{\partial \mathbf{F}} = \frac{\partial(\cos(\alpha_{j_1}) - \cos(\alpha_{j_2}))}{\partial \mathbf{F}} \neq \vec{0}$ as a result of Lemma 4.4 and the fact that each cosine term involves a vertex that the other does not. Therefore by Theorem 4.3, $\mathcal{M}^i$ are 14-dimensional hypersurfaces composed of coordinates of stars satisfying one of the three angle constraints, and $\mathcal{M}^s = \bigcap_{i=1}^3 \mathcal{M}^i$. By Theorem 4.2, $\mathcal{M}^s$ is a 12-dimensional manifold,

and so its tangent space at each point is the intersections of the three tangent spaces of M^i at that point. By counting dimensions, the gradients, which are the normals to the tangent spaces of M^i at the given point cannot be linearly dependent. □

In the next section we set up the necessary tools to show that in general, the Jacobian J_F has full row-rank. The key technique involves representing "small" subsets of linearly dependent rows in J_F in a graph and using the geometry of the constraints as described by Lemma 4.4.

4.4 Linearly dependent DOG constraint gradients

In the following, the matrix J is a Jacobian matrix for the DOG constraints of a proper net. Let $\mathcal{I} \subseteq \{1, 2, ..., m\}$ be a set of indices; $\mathcal{I}$ essentially corresponds to a selection of a subset of DOG constraints on our net. We denote by $C_\mathcal{I}$ the $|\mathcal{I}| \times n$ *constraint matrix* formed by taking the rows of J corresponding to those indices.

Definition 4.6. We call a *minimal* linearly dependent set of rows of J a *circuit*. That is, a circuit is a dependent set of rows such that all its proper subsets are independent. We denote a circuit by the set of indices of its rows, $\mathcal{I} \subseteq \{1, 2, ..., m\}$.

If J is row-rank deficient, then the set of its circuits is not empty. Let $\mathcal{I}$ be a circuit and $C_\mathcal{I}$ its constraint matrix. Since a circuit is a minimal dependent set, there exist coefficients $a_i \neq 0$ such that $\sum_{i=1}^{|\mathcal{I}|} a_i C_i = \vec{0}$, where C_i is the i-th row of $C_\mathcal{I}$.

The rows of $C_\mathcal{I}$ constitute the constraint gradients $\frac{\partial \phi_i}{\partial \mathbf{F}}, i \in \mathcal{I}$, and the columns correspond to vertex coordinates derivatives of each constraint ϕ_i. Since the constraints are local, each involving only a vertex and its neighbors, these rows of $C_\mathcal{I}$ (and J) are sparse. Hence, the connectivity of the net F plays a significant role in the linear combination $\sum_i a_i C_i = \vec{0}$. This is explained by the following trivial but important lemma.

Lemma 4.7. *Let $\frac{\partial \phi_i}{\partial \mathbf{F}}, \frac{\partial \phi_j}{\partial \mathbf{F}}$ be gradients of constraints on angles around vertices v and u, respectively. If the gradient vectors have a common non-zero coordinate (i.e., the derivatives of both ϕ_i and ϕ_j w.r.t. a coordinate do not vanish) then either $v = u$ or v and u are neighbors connected by an edge.*

Definition 4.8. Let $\mathcal{G}$ be the vertex adjacency graph of a net F. Let $\mathcal{I} \subseteq \{1, 2, ..m\}$. The *constraints graph* of $\mathcal{I}$ is a graph $\mathcal{G}_\mathcal{I} \subseteq \mathcal{G}$ defined as follows:

(1) The vertices of $\mathcal{G}_\mathcal{I}$ are those vertices of F whose three corresponding columns in $C_\mathcal{I}$ (for the x, y, z coordinates) are not all zeros. By Lemma 4.7, this means that there are angles around the vertices or their neighbors that contribute to at least one constraint in the subset $\mathcal{I}$.
(2) There is an edge $e = (v, u)$ in $\mathcal{G}_\mathcal{I}$ between two vertices $v, u \in \mathcal{G}_\mathcal{I}$ if this edge exists in $\mathcal{G}$ and there is a constraint $\phi_i,\ i \in \mathcal{I}$, whose partial derivative w.r.t. both the coordinates of v and u does not vanish. This implies in particular that ϕ_i is constraining some angles around v or u.

See Fig. 5 for an example. We now turn to characterizing the constraints graphs of circuits by formulating a few simple lemmas. We study the connectivity structure of circuit's constraints graphs and show that such graphs are connected and contain leaves, i.e., vertices that are connected to only one other vertex. These combinatorial results help us deduce an important geometric fact about the DOG embedding: the leaves are connected by an edge to straight coordinate lines, an observation that is paramount to our algorithm as it allows us to prove that the shape space is locally a manifold and to detect and handle singularities.

Lemma 4.9. *The constraints graph of a circuit is connected.*

Proof. Let $\mathcal{I}$ be the set of indices of a minimal dependent set of rows in J and let $C_\mathcal{I}$ be its constraint matrix. We have $\sum_{i=1}^{|\mathcal{I}|} a_i C_i = \vec{0}$ for some $a_i \neq 0$. Hence we can perform Gauss elimination on the rows of $C_\mathcal{I}$ to cancel a given row, w.l.o.g. the first row. By Lemma 4.4 every step in the elimination is an operation on rows corresponding to angles of the same vertex or on two neighboring vertices. Therefore a minimal set of row operations used to cancel the first row is performed on gradients of constraints corresponding to angles in a connected path to the vertex associated with the angles of the first row's constraint. □

Definition 4.10. A vertex F in a constraints graph of a circuit is a *leaf* if it has only one neighbor. If this neighbor is to the left (resp. right, up or down) from F, it is said to be connected to the left (resp. right, up or down).

Definition 4.11. A vertex F in a constraints graph of a circuit is a *corner* if it has exactly two neighbors, each in a different lattice direction. For example the top right corner in the rightmost graph in Fig. 5.

Lemma 4.12. *(See Fig. 6, left.) Let $\mathcal{G}_\mathcal{I}$ be a constraints graph of a circuit. If $F_{\bar{1}}$ is a leaf connected to the right, then vertex F and all its neighbors, namely $F_{\bar{1}}, F, F_2, F_{\bar{2}}, F_1$ are all in $\mathcal{G}_\mathcal{I}$, and $F_{\bar{1}}, F, F_2, F_{\bar{2}}$ lie on a plane.*

Proof. There exist coefficients $a_i \neq 0$ such that $\sum_i a_i C_i = \vec{0}$. The vertex $F_{\bar{1}}$ is connected only to one other vertex in $\mathcal{G}_\mathcal{I}$. We can assume w.l.o.g. that there are either 1 or 2 constraints where the values corresponding to the columns of $F_{\bar{1}}$ are nonzero. The nonzero values are

$$\frac{\partial \left(\cos(\alpha_{11}) - \cos(\alpha_{10})\right)}{\partial F_{\bar{1}}}, \quad \frac{\partial \left(\cos(\alpha_{12}) - \cos(\alpha_{11})\right)}{\partial F_{\bar{1}}}.$$

There exist a_1, a_2 that are not both zero, such that

$$a_1 \frac{\partial \left(\cos(\alpha_{11}) - \cos(\alpha_{10})\right)}{\partial F_{\bar{1}}} + a_2 \frac{\partial \left(\cos(\alpha_{12}) - \cos(\alpha_{11})\right)}{\partial F_{\bar{1}}} = 0.$$

Naturally, $\frac{\partial \cos(\alpha_{12})}{\partial F_{\bar{1}}} = \vec{0}$ since the angle α_{12} is not affected by the vertex $F_{\bar{1}}$. This means there exist a_1, a_2 that are not both zero such that

$$a_1 \frac{\partial \left(\cos(\alpha_{11}) - \cos(\alpha_{10})\right)}{\partial F_{\bar{1}}} = a_2 \frac{\partial \cos(\alpha_{11})}{\partial F_{\bar{1}}}. \tag{3}$$

By Lemma 4.4, both $\frac{\partial \cos(\alpha_{10})}{\partial F_{\bar{1}}} \neq \vec{0}$ and $\frac{\partial \cos(\alpha_{11})}{\partial F_{\bar{1}}} \neq \vec{0}$ for a non-degenerate net. Therefore there are 2 constraints around F, which implies that all neighbors of F are in $\mathcal{G}_\mathcal{I}$. Eq. (3) also implies $\frac{\partial \cos(\alpha_{10})}{\partial F_{\bar{1}}}$

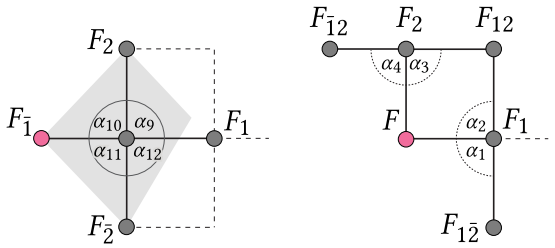


Fig. 6. Illustrations for lemmas about leaves and corners in a constraints graph $\mathcal{G}_I$ of a circuit. If $F_{\bar{1}}$ is a leaf vertex connected to F, then all neighbors of F are in $\mathcal{G}_I$, and $F_{\bar{1}}$, F, F_2, $F_{\bar{2}}$ lie on a plane. If F is a corner vertex with neighbors F_1, F_2, then F_{12}, $F_{\bar{1}2}$, $F_{1\bar{2}}$ are also in $\mathcal{G}_I$.

is parallel to $\frac{\partial \cos(\alpha_{11})}{\partial F_{\bar{1}}}$. By Lemma 4.4, both these vectors are obtained by rotating the same edge by $\frac{\pi}{2}$ in two planes. The vectors can thus be parallel if and only if these two planes are equal. □

Lemma 4.13. *Let F be a corner in a circuit's constraints graph $\mathcal{G}_I$ of a proper net. The neighbors of F are not corners in $\mathcal{G}_I$. In particular, if the neighbors of F are F_1, F_2, then $F_{12}, F_{\bar{1}2}, F_{1\bar{2}}$ are in $\mathcal{G}_I$.*

Proof. See Fig. 6 (right) for notation. As F is a corner in $\mathcal{G}_I$ there are rows in C_I whose partial derivative w.r.t. to the coordinates of F do not vanish. By Lemma 4.7 these are linear combination of gradients constraining angles of nearby vertices. As $\mathcal{G}_I$ is a circuit's constraint graph there are w.l.o.g. 4 coefficients a_i such that $\sum_i a_i \frac{\partial \cos(\alpha_i)}{\partial F} = 0$. This is equivalent to

$$a_1 \frac{\partial \cos(\alpha_1)}{\partial F} + a_2 \frac{\partial \cos(\alpha_2)}{\partial F} = -\left(a_3 \frac{\partial \cos(\alpha_3)}{\partial F} + a_4 \frac{\partial \cos(\alpha_4)}{\partial F}\right).$$

By Lemma 4.4, the left-hand side is orthogonal to the edge $F_1 - F$, while the right-hand side is orthogonal to $F_2 - F$, hence both sides of the equation are vectors that are orthogonal to both edges. Assume that $F_{\bar{1}2}$ is not in $\mathcal{G}_I$, then $a_4 = 0$, and in particular $\frac{\partial \cos(\alpha_3)}{\partial F}$ is orthogonal to $F - F_1$ and $F - F_2$, implying that the dihedral angle between the planes of F, F_2, F_1 and F, F_2, F_{12} is $\frac{\pi}{2}$ and thus contradicting the fact that the net is proper. By similar arguments $F_{12}, F_{1\bar{2}}$ are also in $\mathcal{G}_I$. □

Lemma 4.14. *Let $\mathcal{G}_I$ be a circuit's constraints graph of a proper net, then $\mathcal{G}_I$ contains leaves connected to the right, left, up and down.*

Proof. We show that there exists a leaf connected to the right. The rest of the proof is analogous. Let V_L be the set of "leftmost" vertices in $\mathcal{G}_I$, $V_L = \min_j\{F(j,h) \in \mathcal{G}_I\}$. Let $F_{\bar{1}} \in V_L$ be the "lowest" vertex in V_L; so there are no neighbors to the left or down of $F_{\bar{1}}$. This means that $F_{\bar{1}}$ is a corner or a leaf. The rest of the proof is detailed in Fig. 7. □

Until now, we analyzed the constraint Jacobian J, but we did not assume the net F is a DOG net. The next lemma relates the existence of leaves in a constraints graph $\mathcal{G}_I$ to the geometry of DOG nets.

Lemma 4.15. *Let $F_{\bar{1}}$ be a leaf connected to the right in a circuit's constraints graph $\mathcal{G}_I$ of a DOG net, then $F, F_2, F_{\bar{2}}$ are collinear.*

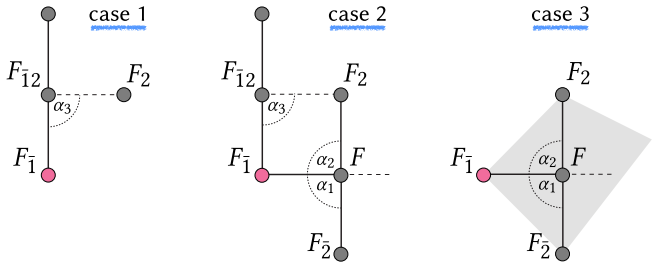


Fig. 7. Lemma 4.14. If $F_{\bar{1}}$ is the lowest vertex among all "leftmost" vertices, there are three cases for the neighborhood of $F_{\bar{1}}$ in the constraints graph $\mathcal{G}_I$, as depicted above. Case 1 is not possible for a circuit constraints graph, since by Lemma 4.4, $\frac{\partial \cos(\alpha_3)}{\partial F_{\bar{1}}} \neq 0$. Case 2 is not possible due to Lemma 4.13, because $F_{\bar{1}2}$ cannot have a neighbor to the left since $F_{\bar{1}}$ is the "leftmost" vertex. So we must have case 3, and then by Lemma 4.12, $F_{\bar{1}}$, F, F_2, $F_{\bar{2}}$ are all in $\mathcal{G}_I$ and lie on the same plane.

Proof. By Lemma 4.12, the neighbors of F are in $\mathcal{G}_I$ and the points $F_{\bar{1}}, F, F_2, F_{\bar{2}}$ are coplanar. If $F, F_2, F_{\bar{2}}$ are not collinear, then by the direction propagation lemma in [Rabinovich et al. 2018] (Lemma 8.2), F_1 lies on the ray of $F_{\bar{1}} - F$ and hence the star of F is "folded-over", so the net is not regular. □

4.5 The shape space of DOG nets

Definition 4.16. A DOG net is *x-ruled* if there is a vertex F such that $F, F_1, F_{\bar{1}}$ are collinear. It is *y-ruled* if $F, F_2, F_{\bar{2}}$ are collinear.

Definition 4.17. A DOG net $F \in \mathcal{M}$ is generic if it is *not* both x-ruled and y-ruled. We denote the set of such nets by $\mathcal{M}_G$.

Note that a cylinder in curvature line parameterization is generic, but a planar patch is not.

Theorem 4.18. *For a net $F \in \mathcal{M}_G$, the rows of its Jacobian J are linearly independent, and the shape space around F is locally a manifold. The tangent space is of dimension $k = n - m = O(|\partial F|)$.*

Proof. By Lemma 4.14 and Lemma 4.15, if the rows of J are linearly dependent then the net is both x-ruled and y-ruled, so it is not generic. Therefore the rows of J must be linearly independent, and the rest follows from Theorem 4.3. Note that the tangent space dimension $k = n - m$ is $O(|\partial F|)$, the number of boundary vertices, because n is 3 times the total number of vertices and m is dominated by the number of inner vertices times 3 angle constraints per each such vertex. □

If the net connectivity is an $l \times r$ rectangular grid patch, and we consider a geometric realization F that is generic, then around F the shape space $\mathcal{M}$ is locally a manifold of dimension $k = 4(l + r - 1)$. Sadly, $\mathcal{M}$ is not a manifold globally, because there are singular points on $\mathcal{M}$.

Theorem 4.19. *$\mathcal{M}$ is not a manifold. In particular, a flat disc topology patch has $O(n)$ linearly independent flows.*

Proof. Let F be a planar disc topology patch whose z-coordinates are zero. One can choose any set of vertices $\{F(i)\}$ that are at least two edges apart from each other and push them in the z-direction at variable speeds v_i. See Fig. 8. □

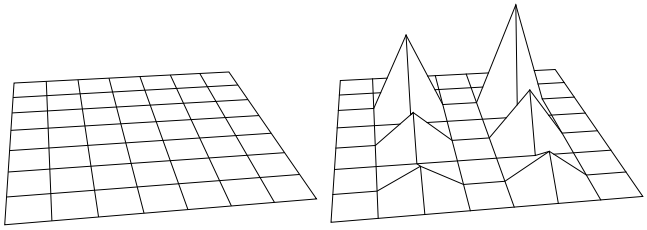


Fig. 8. A planar DOG net has more flows than a general typical DOG net, and it admits non-smooth flows, since one can push inner vertices in the z-direction independently from other vertices. This freedom is a discrete artifact that exists only on such singular nets, since typically DOG nets do not admit local deformations.

Direct calculation also shows that in the special case of a planar net, the gradients of the 12 constraints around an inner quad, i.e. a quad surrounded by 4 inner vertices, are linearly dependent. Nevertheless, a general smooth DOG deformation on a planar net moves it away from being a singularity and into $\mathcal{M}_G$.

The following lemma shows how we can handle singularities by slightly offsetting the mesh vertices and flowing on a nearby manifold.

Theorem 4.20. *Let $F \in \mathcal{M}$ be a singular point in $\mathcal{M}$, then for every $\epsilon > 0$ there exists a net F^* (not necessarily a DOG) with the same connectivity as F in distance smaller than ϵ such that the constraints Jacobian J has full row rank. Denote the constraint values on F^* as $\phi_i(\mathbf{F}^*) = \epsilon_i$; then F^* lives on a $k = n - m$ dimensional manifold of nets satisfying $\phi_i = \epsilon_i$.*

Proof. Lemma 4.14 does not depend on a net being a DOG. Let $\epsilon > 0$, construct F^* by slightly offsetting vertices such that nearby triangles are not coplanar as in Lemma 4.12, then the Jacobian of F^* has full row rank. □

5 DISCRETIZING DOG FLOWS

A corollary of Theorem 4.18 is the existence of a multitude of continuous deformations of generic DOGs, or flows on the shape space $\mathcal{M}$. In this section we show how to discretize these flows and how to deform singularities based on the theory developed in Sec. 4. The resulting algorithm is summarized in Algorithm 1.

5.1 Constrained gradient flows

Following the work of [Eckstein et al. 2007], we would like to define a variety of *gradient flows*. Let F be a given DOG net and let $E(\mathbf{F})$ be an objective function we are interested in minimizing. Our goal is to devise an evolution of the mesh geometry $\mathbf{F}$ that decreases the objective with an infinitesimal change of the surface. We can readily use various established objective energies $E(\mathbf{F})$, but we need to define the infinitesimal change of a surface precisely. This requires a measure of steepness, i.e., a metric M on the space of DOGs $\mathcal{M}$, defined on tangent vectors of $\mathcal{M}$, or on $T\mathcal{M}$. An often employed metric is the metric induced by the canonical L^2 inner product on $\mathcal{M}$, and the associated gradient operator is denoted by ∇, e.g., $\nabla E(\mathbf{F})$. However, as demonstrated in [Eckstein et al. 2007], it is often beneficial to use other metrics on tangent spaces. One can define a variety of gradient operators ∇_M induced by metrics M by using more general inner products of the form $\langle \mathbf{F}, \tilde{\mathbf{F}} \rangle_M = \langle M\mathbf{F}, \tilde{\mathbf{F}} \rangle_{L^2} = \langle \mathbf{F}, M\tilde{\mathbf{F}} \rangle_{L^2}$. The metric here is any symmetric positive definite matrix $M \in \mathbb{R}^{n \times n}$. The gradient of a function can then be computed by solving a linear system whose right-hand side is the L^2 gradient: $\nabla_M E(\mathbf{F}) = M^{-1} \nabla \mathbf{F}$.

All our results use a Laplacian matrix $\mathbb{L}$ as the metric M, which we derive specifically for DOG nets in Sec. 6. A comparison between using this metric vs. the standard L_2 metric is shown in Fig. 9 (left). As a notable difference to the setting in [Eckstein et al. 2007], we are interested in *constrained flows* that do not deviate from our DOG constraints, i.e., stay in the shape space $\mathcal{M}$. This amounts to calculating the projection of the gradient of $E(\mathbf{F})$ onto the tangent space $T\mathcal{M}$ at point $\mathbf{F}$, using the chosen metric M. We denote this projected gradient by $\overline{\nabla}_M E(\mathbf{F})$:

$$\overline{\nabla}_M E(\mathbf{F}) := \arg\min_{\mathbf{t}} \|\nabla_{\mathbf{M}} \mathbf{E}(\mathbf{F}) - \mathbf{t}\|_M, \quad \mathbf{t} \in T\mathcal{M}. \tag{4}$$

A comparison between a Willmore flow with and without projecting the gradients onto $T\mathcal{M}$ is shown in Fig. 9 (right). Let F be a DOG net, $E(\mathbf{F})$ some objective function, $\nabla E(\mathbf{F}) \in \mathbb{R}^n$ the standard L_2 gradient of E at F, and $J \in \mathbb{R}^{m \times n}$ the constraint Jacobian matrix of F, defined as in Sec. 4. The projected gradient $\overline{\nabla}_M E(\mathbf{F}) \in \mathbb{R}^n$ can be computed by solving the following KKT system, where $\lambda \in \mathbb{R}^m$ are the Lagrange multipliers:

$$\begin{gathered} K \begin{pmatrix} \overline{\nabla}_M E(\mathbf{F}) \\ \lambda \end{pmatrix} = \mathbf{b} \\ K = \begin{pmatrix} M & J^\mathsf{T} \\ J & 0 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \nabla E(\mathbf{F}) \\ \mathbf{0} \end{pmatrix}. \end{gathered} \tag{5}$$

By Theorem 4.18, J has full row rank for a generic net. This implies that the system in Eq. (5) is non-singular if the metric M is positive definite [Nocedal and Wright 2006]. The system is also precise, as it computes a vector on $T\mathcal{M}$ corresponding to the gradient of $E(\mathbf{F})$ under the metric M. Thus we can compute a flow by iteratively advancing by a variable step size t in the direction computed in Eq. (5), where our flow approaches the continuous flow as $t \to 0$. In practice, to allow for fluid interaction with a steady frame rate, we often have a limited time budget that precludes full convergence, so every iteration is computed starting with a net that is not precisely a DOG net, but rather satisfies $\phi(\mathbf{F}) = \boldsymbol{\epsilon}$, with a small norm $\|\boldsymbol{\epsilon}\| < \epsilon$. We therefore solve the following system:

$$\begin{gathered} K \begin{pmatrix} \overline{\nabla}_M E(\mathbf{F}) \\ \lambda \end{pmatrix} = \mathbf{b} \\ K = \begin{pmatrix} M & J^\mathsf{T} \\ J & 0 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \nabla E(\mathbf{F}) \\ \boldsymbol{\epsilon} \end{pmatrix}. \end{gathered} \tag{6}$$

5.2 Handling singularities

The shape space $\mathcal{M}$ is not always a manifold locally, as shown in Theorem 4.19. Non-generic nets can be singular points on the DOG shape space $\mathcal{M}$. There are in fact singular nets that are quite important for modeling, in particular planar nets, which possess more linearly independent flows than the dimension of $T\mathcal{M}$ for generic points (see the proof of Theorem 4.19). By Theorem 4.3, this also implies that the system in Eq. (5) is singular for planar $\mathbf{F}$.

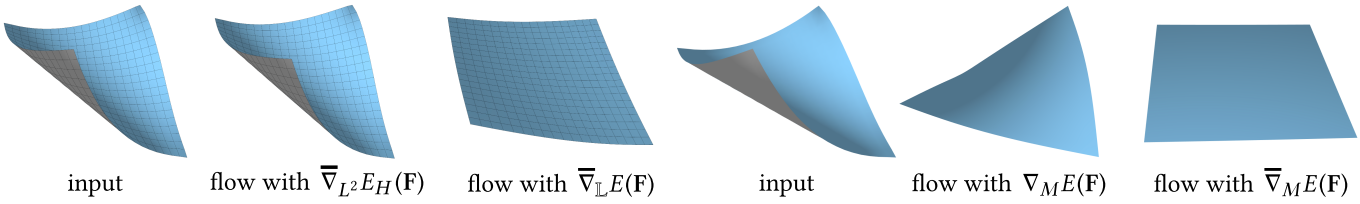


Fig. 9. The effect of shape space metric M (left) and gradient projection (right) on a flow minimizing squared mean curvature (as defined in Def. 6.8). Left: we show the input model and the results of 200 iterations of the flow (taking 1.4 seconds on a 20×20 grid), using the standard L^2 as metric M vs. the DOG Laplacian metric $\mathbb{L}$ for the projected gradient $\overline{\nabla}_M E(\mathbf{F})$. When using the L^2 metric, the iterations hardly advance. Right: Flowing by iteratively computing the gradient $\nabla_M E(\mathbf{F})$ without projection leads to a non DOG-net. Specifically, the flow objective pushes both families of coordinate lines to be straight rulings along a hyperbolic surface. In contrast, the flow with the projected gradient $\overline{\nabla}_M E(\mathbf{F})$ advances along the DOG shape space towards a planar net. Both examples were computed by only flowing along the gradients, without the additional DOG constraints optimization.

We could remove redundant linearly dependent rows from J, as often done when solving KKT systems using sparse QR or LU decomposition [Nocedal and Wright 2006]. This would compute a flow direction while keeping the DOG constraints up to first order, but is significantly slower. We therefore use the observation of Theorem 4.20 to devise a simple and computationally cheap strategy that introduces a minimal change to the solved system. In every flow iteration, we find the sets of x-ruled and y-ruled vertices. If both sets are not empty, the net is not generic. In that case, we choose the smaller of the two sets of vertices and randomly perturb the vertex positions by an ϵ to obtain a new net F^*. We then solve the system in Eq. (5) on the net F^*, which, by Theorem 4.20, has full rank. In essence, if the values of the constraints on $F*$ are $\phi_i(\mathbf{F}^*) = \epsilon_i$, then the computed flow direction corresponds to a tangent vector on the tangent space of another manifold $\mathcal{M}^*$, which is close to $\mathcal{M}$ and is the manifold of nets satisfying $\phi_i(\mathbf{F}^*) = \epsilon_i$.

Linear dependencies between the constraint gradients of a singular mesh imply the existence of a larger set of deformations, where the net's angle constraints are kept up to first order (see Fig. 8), and these are avoided by perturbing the vertices, since the nearby manifold has the dimension of the manifold of generic DOGs. The vertex perturbation strategy also results in a significant speedup; we solve the resulting linear system using PARDISO [Kourounis et al. 2018; Verbosio et al. 2017; De Coninck et al. 2016]. On a 30×30 planar mesh this solve takes 0.0197 seconds, whereas removing linear dependencies with SuiteSparse [Davis 2011] using sparse QR increases the run time by a factor of $\times 8.4$. Solving the entire system with MOSEK [MOSEK ApS 2017], whose solver uses LU factorization to remove linear dependencies, is 10.3 times slower. See supplementary video for various examples of editing starting from a singularity, as well as Fig. 10.

5.3 Algorithm

We use a line search to minimize the objective $E(\mathbf{F})$ at each flow step. Since the flow direction keeps the DOG constraints only up to first order, after each flow step we project the resulting mesh to the DOG shape space using a penalty based method with LBFGS [Nocedal 1980], as done in [Rabinovich et al. 2018] section 6.3. In our case however, the geometry we project is already very close to the shape space, so this step is more efficient and also results in a smoother deformation (see Fig. 16 and supplementary video). To sum up, a step in our flow amounts to solving a sparse, symmetric indefinite system, followed by LBFGS. We list the steps in Algorithm 1 below; throughout the paper we use the parameter values $t_0 = 1, \epsilon = 1e{-}8$.

Algorithm 1: Discrete orthogonal geodesic flow

Input:
A discrete orthogonal geodesic net F^k and an objective function $E(\mathbf{F})$.

Output:
A discrete orthogonal geodesic net F^{k+1}.

1: Find the sets of x-ruled and y-ruled vertices, $\mathcal{I}^x$ and $\mathcal{I}^y$.
2: Set $\mathcal{I}$ as the smaller of the two sets $\mathcal{I}^x, \mathcal{I}^y$.
3: Perturb the position of each vertex in $\mathcal{I}$ in a random direction $r_i \in \mathbb{R}^3$ at distance $\epsilon > 0$ to obtain F^*.
4: Compute the constraint Jacobian $J = J(F^*)$.
5: Solve Eq. (5) to compute $\overline{\nabla}_M E(\mathbf{F})$.
6: Perform a line search on the objective $E(\mathbf{F})$ in direction $\overline{\nabla}_M E(\mathbf{F})$ starting from step size $t = t_0$ to compute F'.
7: Obtain F^{k+1} by minimizing the DOG angle constraints on F' as in [Rabinovich et al. 2018] section 6.3.

On generic nets our flow discretization converges to the existing smooth flow minimizing the objective as t_0 goes to zero. Though we do not have guarantees for singular nets, our measurements indicate convergence on various examples, albeit slower (Fig. 10).

6 DOG LAPLACIAN AND MEAN CURVATURE

Here we supply the basic tools that can be directly plugged in Algorithm 1 to compute various flows that use a DOG Laplacian as a metric and a mean curvature as a basis for objective functions. Rabinovich et al. [2018] used the uniform Laplacian in their optimization. It works well for very regular nets where all edge lengths are nearly equal, but is insufficient when modeling arbitrary stretching deformations on a DOG net, since the uniform Laplacian even does not vanish on planar nets. Our DOG Laplacian derivation is guided by the salient properties of smooth Laplacians: locality, symmetry, positive semidefiniteness and linear precision, as well as convergence of the operator; for details see [Alexa and Wardetzky 2011; Wardetzky et al. 2007]. Inspired by the works of [Pinkall and Polthier 1993; Alexa and Wardetzky 2011] we define a Laplacian operator for DOG nets by deriving the gradient of the surface area of a DOG net. While

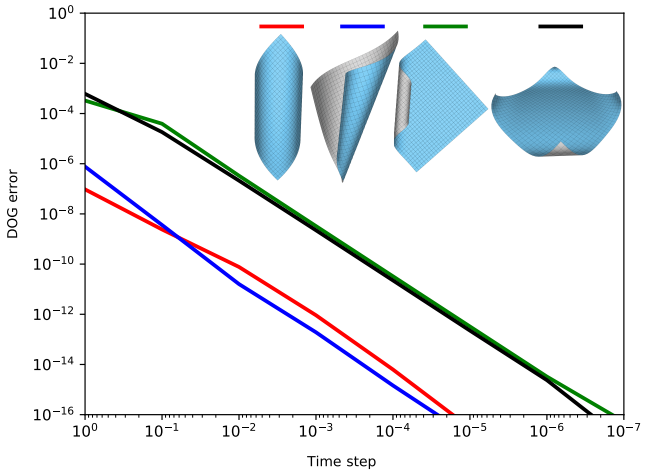


Fig. 10. Convergence plot of generic nets (red and blue) and singular nets (green and black) under Willmore flow as the flow time step goes to zero. We run Willmore flow until each mesh becomes planar. The vertical axis ("DOG error") represents the maximal value of the sum of squared errors of the DOG constraints, taken over the entire flow.

Alexa and Wardetzky [2011] propose a Laplacian for general polygonal meshes, we take advantage of the highly structured nature of our specific nets to derive the area, resulting in a symmetric positive definite Laplacian that converges under sampling of an analytical orthogonal geodesic net.

6.1 DOG vertex area

The quads of a DOG net are generally non-planar. We therefore start with the most basic definitions: the areas of a DOG quad and a DOG vertex. We base these on flattening a DOG net into a planar net. The simplest case is when F is already a planar DOG net, which means it is a planar orthogonal grid. The (planar) quad area is then unambiguously defined, and the area associated with a vertex can be defined as the area of its dual face, formed by connecting the centroids of its adjacent faces (see Fig. 11). Therefore, for a vertex with index i in F, the area is $A_i = \frac{1}{4}\sum_{j\in \mathrm{Quads}(i)} Q_j$, where Q_j is the planar quad area of face j.

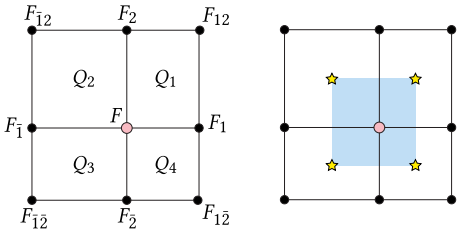


Fig. 11. Left: A planar DOG net around a vertex F (colored light red) and its four surrounding quads, whose areas are denoted by Q_i. Right: The area of the vertex is the area of its dual face (in blue), formed by connecting the centroids of its neighboring faces (marked by stars).

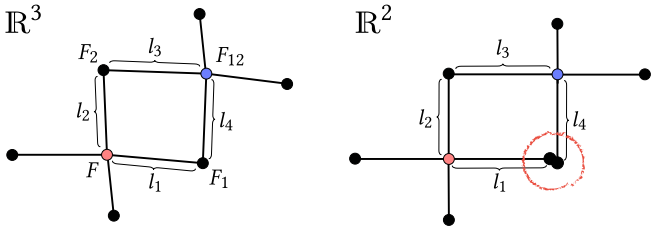


Fig. 12. Planar DOG nets are orthogonal grids, but for general DOG nets, one can locally flatten each star while exactly preserving the edge lengths, but generally not the entire net. Left: a DOG net in $\mathbb{R}^3$ containing two stars (red and blue). Right: individually flattened red and blue stars cannot be "connected" to form a coherent mesh since the lengths l_1, l_3 and l_2, l_4 are generally different, although they are similar, and equal at the smooth limit. We therefore define the area of a quad in a DOG net as $\frac{1}{4}(l_1 + l_3)(l_2 + l_4)$.

The problem with directly generalizing this vertex area definition to non-planar DOG nets is that a DOG star can be isometrically flattened, but a DOG net generally cannot be, see Fig. 12. As shown in the figure, the reason is that opposite edges do not have exactly the same length. Discrete nets whose opposite edges do have the same length are called discrete Chebyshev nets [Bobenko and Suris 2008]. They can be seen as the analogous of smooth Chebyshev nets: nets f satisfying $\|f_x\|_y = \|f_y\|_x = 0$. A smooth orthogonal geodesic net is also a Chebyshev net (Theorem 6.1), hence a DOG net is approximately a discrete Chebyshev net and approaches being exactly one in the (smooth) limit.

Theorem 6.1. *A smooth orthogonal geodesic net f is also a Chebyshev net, i.e., it satisfies $\|f_x\|_y = \|f_y\|_x = 0$.*

Proof. See Appendix C. □

Hence we define the area of a quad in a DOG net by a symmetric formula:

Definition 6.2. Let the lengths of consecutive edges of a quad j in a DOG net be l_1, l_2, l_3, l_4, such that $l_1 \approx l_3$ and $l_2 \approx l_4$. We define the area of quad j as $Q_j = \frac{1}{4}(l_1 + l_3)(l_2 + l_4)$. See Fig. 12.

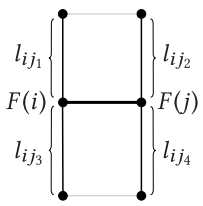
Definition 6.3. We define the total surface area of a DOG net as the sum of the quad areas, $\mathcal{A} = \sum Q_j$. The area associated with a vertex F is $A = \frac{1}{4}(Q_1 + Q_2 + Q_3 + Q_4)$ in the notation of Fig. 11.

6.2 DOG Laplacian

The area gradient of a DOG net results in a weak (integrated) Laplacian operator $\mathbb{L}$:

$$\mathbb{L}F(i) = \frac{\partial \mathcal{A}}{\partial F(i)} = \sum_{j\in\mathcal{N}(i)} \omega_{ij}\,(F(i) - F(j))\,,$$
$$\omega_{ij} = \frac{1}{4}\cdot\frac{l_{ij_1} + l_{ij_2} + l_{ij_3} + l_{ij_4}}{\|F(i) - F(j)\|}, \tag{7}$$

where l_{ij_k} are the respective lengths of the four edges that are adjacent to edge (i, j) and also incident to the two faces that share the edge (i, j) (see inset). The derivation of Eq. (7) can be found in Appendix D.1.



The weights are symmetric and positive: $\omega_{ij} = \omega_{ji} > 0$, which are two desired properties noted in [Wardetzky et al. 2007]. As a result, $\mathbb{L}$ is symmetric and positive semidefinite. In the special case where a vertex is in the center of a planar star, these weights coincide with the cotangent weights of any triangulation of the planar mesh (see Appendix D.2). The latter also implies that $\mathbb{L}$ vanishes on inner vertices in planar nets, i.e., it satisfies the linear precision property for a Laplacian.

6.3 Laplacian mean curvature normal

The Laplacian Δ on a smooth surface f satisfies

$$\Delta f = -2Hn, \tag{8}$$

where H is the mean curvature and n is the unit-length normal to the surface. If f is a smooth orthogonal geodesic net, then the following holds:

Theorem 6.4. *Let f be an orthogonal geodesic net around point $f(x_0, y_0)$ and let $f(x_0+t,\ y_0)$, $f(x_0,\ y_0+t)$ be its coordinate curves at $f(x_0, y_0)$, with respective curvatures k^x, k^y and unit-length principal normals n^x, n^y. Then:*

$$-\Delta f = 2Hn = k^x n^x + k^y n^y. \tag{9}$$

Proof. Let $I\!I$ be the second fundamental form on f. The coordinate curves are geodesics, hence $n \parallel n^x \parallel n^y$ and their curvatures are the surface normal curvatures in directions $f_x,\ f_y$, respectively, meaning $k^x = |I\!I(f_x, f_x)|,\ k^y = |I\!I(f_y, f_y)|$. Let e_1, e_2 be the principal directions of the developable surface f such that $I\!I(e_1, e_1) = 2H,\ I\!I(e_2, e_2) = 0$, and let θ be the angle between f_x and e_1 in the orientation of the tangent plane spanned by f_x, f_y. By plugging in the the orthogonality of the coordinate curves $f_x \perp f_y$ into Euler's formula, we get $k^x + k^y = |I\!I(f_x, f_x)| + |I\!I(f_y, f_y)| = 2H\cos^2(\theta) + 2H\sin^2(\theta) = 2H$. □

Hence in the smooth case the normal and the mean curvature of the net can be deduced by looking at the principal normals and curvatures of coordinate curves , $k^x n,\ k^y n$. This is also the case with the discrete Gauss map of a DOG, N, defined in [Rabinovich et al. 2018] as the intersection of the osculating planes of the discrete curves:

Definition 6.5. The discrete Gauss map of a geodesic net F is

$$N = \frac{T_1 \times T_2}{\|T_1 \times T_2\|},$$

where $T_j = \frac{\delta_j F - \delta_{\bar{j}} F}{\|\delta_j F - \delta_{\bar{j}} F\|},\ j = 1, 2$. It also satisfies

$$N \parallel (\delta_1 F + \delta_{\bar{1}} F) \parallel (\delta_2 F + \delta_{\bar{2}} F), \tag{10}$$

meaning N is the bisector of the discrete curves in their osculating planes, which can be seen as the discrete principal normal of the polylines $(F_{\bar{1}}, F, F_1)$ and $(F_{\bar{2}}, F, F_2)$.

The following lemma is the key in understanding how the Laplacian conforms to the normal map N and how to define a discrete mean curvature on DOGs based on a discrete notion of k^x, k^y.

Lemma 6.6. *Let F be an inner vertex on a Chebyshev DOG net. The DOG Laplacian $\mathbb{L}$ satisfies the following:*

$$\begin{aligned} \mathbb{L}F &= \frac{\partial \mathcal{A}}{\partial F} = -A\,(K^1 N + K^2 N), \\ K^j &= \frac{2\,\|\delta_j F + \delta_{\bar{j}} F\|}{\|F_j - F\| + \|F_{\bar{j}} - F\|} = \frac{4\cos\frac{\beta_j F}{2}}{\|F_j - F\| + \|F_{\bar{j}} - F\|},\ \ j = 1, 2. \end{aligned} \tag{11}$$

Proof. See Appendix D.3. □

As a corollary, we see that the Laplacian of a vertex $\mathbb{L}F(i)$ is parallel to the vertex normal $N(i)$ in case the DOG is a Chebyshev net.

The Laplacian $\mathbb{L}$ is a *weak* Laplacian representing an integrated quantity around a vertex with area A in the form of $\int_A (k^x + k^y)\,dA$, which can be discretized using Lemma 6.6. In that sense, Lemma 6.6 shows how the Laplacian mean curvature is analogous to Theorem 6.4, where K^1, K^2 are the curvatures of the discrete coordinate curves.

Definition 6.7. The normal curvatures of a DOG net, K^1, K^2, are defined as:

$$K^j = \frac{2\,\|\delta_j F + \delta_{\bar{j}} F\|}{\|F_j - F\| + \|F_{\bar{j}} - F\|} = \frac{4\cos\frac{\beta_j F}{2}}{\|F_j - F\| + \|F_{\bar{j}} - F\|},\ \ j = 1, 2. \tag{12}$$

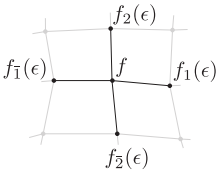
Definition 6.8. The mean curvature of a DOG net is $H = \frac{K^1 + K^2}{2}$.

One could devise a Laplacian that recovers the discrete Gauss map precisely for every DOG net, not necessarily Chebyshev, by defining an area of a star using weights induced only by its edge lengths and avoiding the averaging in Def. 6.2. However, doing so forfeits the important property of symmetry ($\omega_{ij} = \omega_{ji}$) of the Laplacian $\mathbb{L}$. Since a smooth DOG is approximately Chebyshev, our definition yields a precisely symmetric Laplacian, but approximate normals and mean curvature. In the next section we show that this small discrepancy is solved at the limit because a smooth orthogonal geodesic net f is a Chebyshev net.

6.4 Convergence under sampling

We show that the mean curvature normal induced by the discrete Gauss map N, the discrete H as well as the discrete mean curvature of the Laplacian $\mathbb{L}$ all converge under sampling to the correct smooth counterparts for an analytical orthogonal geodesic net f.

Let f be an arbitrary analytical smooth orthogonal geodesic net and $p = f(x, y)$ a point on the surface. One can sample nearby points around p to generate a discrete star. Let $\epsilon > 0$ and define the shorthands $f(\epsilon) = f(x, y),\ f_1(\epsilon) = f(x+\epsilon, y),\ f_{\bar{1}}(\epsilon) = f(x-\epsilon, y),\ f_2(\epsilon) = f(x, y+\epsilon),\ f_{\bar{2}}(\epsilon) = f(x, y-\epsilon)$. For a given $\epsilon > 0$ the star around f is not necessarily a DOG star, but it was shown to converge to one at the limit as $\epsilon \to 0$ [Rabinovich et al. 2018]. Let $N_\epsilon, H_\epsilon, \mathbb{L}_\epsilon$ be the Gauss map, discrete mean curvature and DOG Laplacian as defined above. These all converge under sampling by the following theorem.



Theorem 6.9. *If f is an analytical smooth orthogonal geodesic net, then the following holds:*

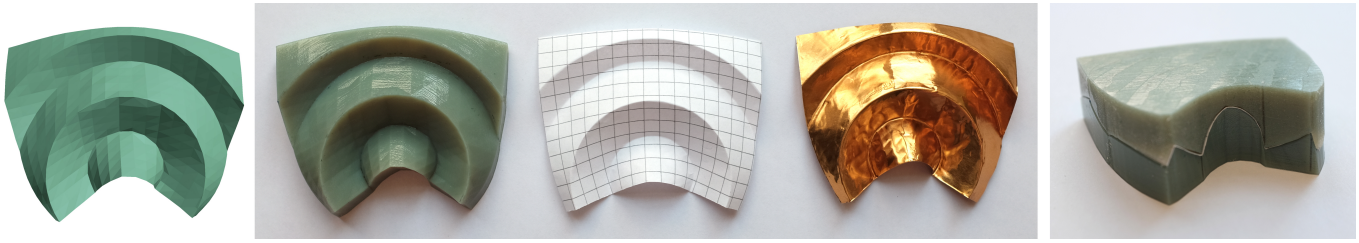


Fig. 13. Design and fabrication of a surface with curved folds (four concentric circles). From left to right: the DOG mesh, triangulated for rendering and 3D printing purposes; 3D printed "mold"; rectangular sheets of paper and foil, manually formed by the mold; fixating the shape of the paper in a 3D printed "sandwich". Note that the rectangular paper sheet perfectly fits into the sandwich without wrinkling, indicating that our DOG surface is reasonably developable in the physical sense. The foil sheet is highly pliable and easily deforms when handling, even showing the impressions of the coarse triangles of the mold.

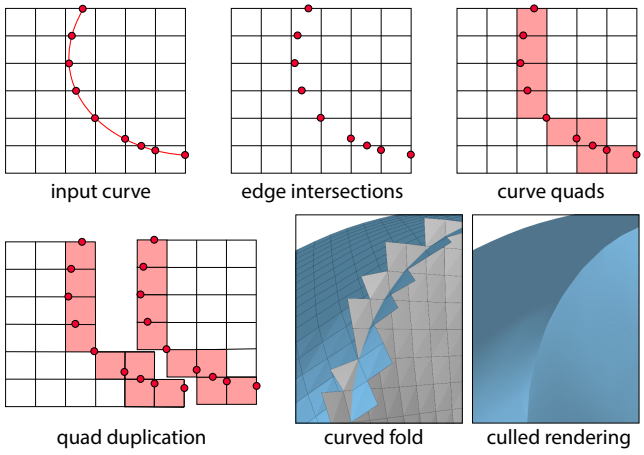


Fig. 14. Crease curves on DOGs. A curve on the surface is represented by its intersections with mesh edges. To model a sharp crease, we duplicate the faces intersected by the curve and split the mesh, adding C^0 continuity constraints. For curved folds, we also incorporate flattenability conditions along the curve. For rendering purposes, extraneous parts of the duplicated faces are culled.

$$(1)\ \lim_{\epsilon\to 0} K^1_\epsilon N_\epsilon = k^x n \ \ and \ \ \lim_{\epsilon\to 0} K^2_\epsilon N_\epsilon = k^y n$$

$$(2)\ \lim_{\epsilon\to 0} H_\epsilon N_\epsilon = \lim_{\epsilon\to 0} \mathbb{L}_\epsilon f(\epsilon) = Hn$$

The proof is detailed in Appendix E.

7 CREASES AND CURVED FOLDS

Up to this point our discussion has been limited to discretizing smooth deformations on developable surfaces, since the Laplacian metric and the mean curvature based objective (alluded to in Sec. 6 and formulated below in Sec. 8.1) are smooth functions. Applying such deformations to a planar sheet solely bends the surface but does not generate sharp creases, which are curves on the surface with tangents discontinuities. We would like to extend the set of deformations to model *piecewise* smooth developable surfaces, allowing tangent discontinuities along arbitrary curves on the surface. We look for a simple solution supporting arbitrary crease directions. We therefore represent a curve on our surface by its intersection points with edges on our net: $c = \{c(1), ..., c(s)\}$ (see Fig. 14, left).

Each point $c(i)$ is a linear combination of two vertices on our net: $c(i) = tF(i_1) + (1-t)F(i_2)$, where $0 \le t \le 1$ and i_1, i_2 are the indices of the edge vertices. To allow discontinuities along this curve, we further split our mesh along it, duplicating faces touching the edges of the curve, and enforce C^0 continuity of the net by constraining the duplicated points $c_1(i), c_2(i)$ on the edges on both meshes to have equal coordinates (see Fig. 14). This can be formulated as linear constraints in the form of $\psi_i = c_1(i) - c_2(i) = 0$, since $c_1(i), c_2(i)$ are linear combinations of vertices, and added to the constraint Jacobian J in Algorithm 1. Note that in our rendering we cull the extraneous parts of the faces intersecting a crease curve past the cuts (see Fig. 14).

7.1 Local flattenability condition

A piecewise smooth developable surface f is composed of multiple smooth developable surfaces f_i, intersecting along curves c_j where the surface is not C^1. In general, each patch f_i can be flattened onto the plane but two patches sharing such a discontinuity curve cannot be flattened together, unless the curve has the same geodesic curvature on both patches [Kilian et al. 2008]. If all patches around a curve c_j can be locally flattened, c_j is a curved fold or a straight crease. The latter is a special case where c_j is a geodesic. Modeling curved folds in our framework can be done by penalizing the quadratic difference in edge lengths of duplicated quads (see Fig. 14, right) or by using E_{iso} as defined in Sec. 8.

8 APPLICATIONS

Algorithm 1 enables us to locally explore a given DOG shape space using smooth transformations, and it can be directly integrated into a modeling framework by choosing a suitable objective function $E(\mathbf{F})$ and a metric M on the shape space. We use the DOG Laplacian $\mathbb{L}$ as a base for the metric, and we choose $E(\mathbf{F})$ as a weighted combination of extrinsic, intrinsic and tangential (parameterization controlling) terms.

8.1 Objective functions

Our first objective function is a bending minimization term defined as

$$E_H(\mathbf{F}) = \sum_{i \text{ vertex in } F} A_i H_i^2. \tag{13}$$

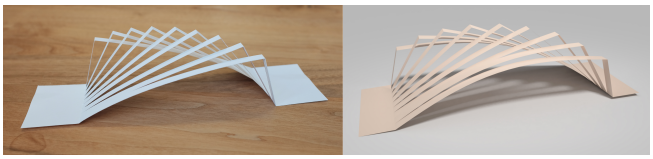


Fig. 15. A kirigami surface folded from a planar sheet after applying multiple cuts and straight creases. On the left we show our physical mockup made of thick paper, and on the right is a physically based rendering of the corresponding DOG mesh designed in our system.

The above term is a direct discretization of Willmore flow [Bobenko and Schröder 2005] constrained to DOG nets.

The use of the Laplacian as a metric often prevents large stretching deformations (see Fig. 9), but for more elaborate editing the user might wish to add an explicit term to preserve the length of curves. Let e be an edge on the mesh, l_e its length and l_e^0 the length in a reference mesh. We define the following intrinsic isometry term:

$$E_{\text{iso}}(\mathbf{F}) = \sum_{e \text{ edge in } F} (l_e - l_e^0)^2. \tag{14}$$

While one might prefer to limit stretching when modeling, allowing for stretching a developable surface can in fact be beneficial for editing tasks, since this does not limit the model to a fixed reference isometric sheet, which is often unknown during the design phase. We show in Fig. 16 and in the accompanying video that our algorithm naturally handles stretching, as opposed to the work of [Rabinovich et al. 2018], which uses ARAP [Sorkine and Alexa 2007] as an initial guess before optimizing the DOG constraints. Our approach of iteratively computing a surface that only slightly deviates from the DOG constraints before optimizing the constraints directly is useful in generating a smooth deformation sequence, especially one that allows for large stretching deformations. As shown in Fig. 16, stretching a DOG can also produce large tangential deformations, leading to a non-uniform grid. We therefore employ the following terms to control the parameterization regularity and minimize these

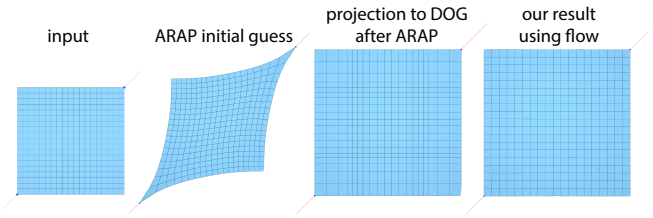


Fig. 16. Stretching a planar orthogonal grid along the diagonal by moving point handles at two diagonally opposing corners. The algorithm of [Rabinovich et al. 2018] computes an initial guess with ARAP, and then projects the guess to a nearby DOG. Strongly stretching the surface causes a distorted ARAP initial guess lying far away from the DOG shape space, leading to a jittery editing experience (see the accompanying video), and also an non-uniform grid. In contrast, when using our flow algorithm, the editing is temporally smooth, and the mesh grid stays uniform thanks to the grid regularity term in the flow objective. In this simple example, following our flow with a projection to the DOG space does not alter the mesh. We used $w_H = 1$, $w_{\text{uni-all}} = 0$, $w_{\text{iso}} = w_{\text{uni-opp}} = 0$, $w_{\text{p}} = 0.5$ in the flow objective.

tangential deformations:

$$E_{\text{uni-all}}(\mathbf{F}) = \sum_{\text{star in } F} (l_1 - l_2)^2 + (l_3 - l_4)^2 + (l_1 - l_3)^2 + (l_2 - l_4)^2,$$
$$E_{\text{uni-opp}}(\mathbf{F}) = \sum_{\text{star in } F} (l_1 - l_3)^2 + (l_2 - l_4)^2, \tag{15}$$

where l_1, l_2, l_3, l_4 are the lengths of consecutive edges around a star. The term $E_{\text{uni-all}}(\mathbf{F})$ penalizes non-uniform grids, but it is impossible to satisfy if one models a rectangular sheet, for instance. The term $E_{\text{uni-opp}}(\mathbf{F})$ vanishes when the lengths of the coordinate curves in the x and y directions are distributed uniformly across the edges.

Soft constraints. Smoothing operations or freeform editing necessitate incorporating positional constraints into our flows, which we add analogously to the flows in [Desbrun et al. 1999]. Denote $\mathbf{C}$ as the set of constrained vertices indices, $F(i)$ as a given constrained vertex position and $P(i)$ as the desired constrained position for $i \in \mathbf{C}$. Adding soft positional constraints amounts to adding a weight parameter $w_{\text{p}} > 0$ and an objective term of the form

$$w_{\text{p}} E_{\text{p}}(\mathbf{F}) = w_{\text{p}} \sum_{\text{i in C}} \|F(i) - P(i)\|^2$$

, as well as setting the metric to $M = \mathbb{L} + D_{\text{p}}$, where $\mathbb{L}$ is the DOG Laplacian and D_{p} is a diagonal matrix with w_{p} on the diagonal entries corresponding to the constrained vertex coordinates and 0 otherwise.

The overall objective $E(\mathbf{F})$ is composed as a weighted sum:

$$E(\mathbf{F}) = w_H E_H + w_{\text{iso}} E_{\text{iso}} + w_{\text{uni-all}} E_{\text{uni-all}} + w_{\text{uni-opp}} E_{\text{uni-opp}} + w_{\text{p}} E_{\text{p}},$$

where $w_H, w_{\text{iso}}, w_{\text{uni-all}}, w_{\text{uni-opp}}, w_{\text{p}}$ are user defined weights.

8.2 Smoothing and subdivision

The definition of a DOG net does not encode smoothness, hence a DOG net can be jaggy, as in Fig. 8. This is common in discrete differential geometry [Bobenko and Suris 2008], as the space of discrete nets is in fact richer than that of smooth nets. We are interested in a subset of DOG nets that is smoother, and possibly containing a set of crease curves. Fig. 18 shows how our Willmore flow can be used for smoothing with suitable positional boundary constraints. In the same figure we show a subdivision operation in the spirit of the subdivision in [Liu et al. 2006]. Since standard subdivision schemes applied to a DOG mesh generate a non-DOG mesh, here we implement a DOG subdivision operator as a combination of Catmull-Clark followed by the minimization of DOG constraints and smoothing. An example of smoothing a singular net can be found in the accompanying video. We note that smoothing operations are not possible in the optimization framework of [Rabinovich et al. 2018], since the smoothness term in their objective function measures deviation of the Laplacian mean curvature normals from a given reference: when the reference surface is jaggy but nonetheless satisfies the DOG constraints, their optimization keeps it as is.

8.3 Free form editing

Our flows can be directly plugged into an editing system. Unless stated otherwise, the examples created in this paper were generated using the objective weights $w_H = w_{\text{uni-opp}} = 1$, $w_{\text{iso}} = 0.1$, $w_{\text{uni-all}} = 0$, $w_{\text{p}} = 0.5$. Similarly to [Rabinovich et al. 2018], we

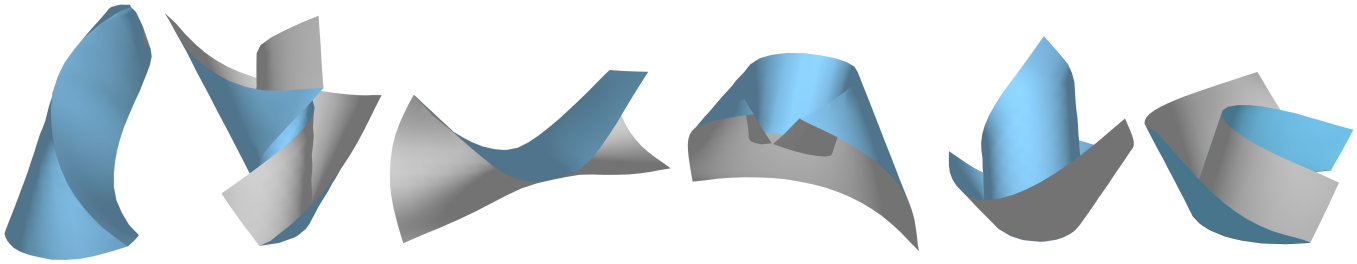


Fig. 17. Discrete developable surfaces created using our point-handle based editing system.

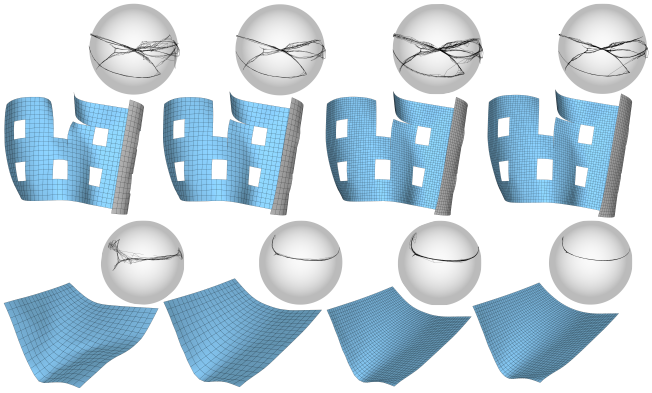


Fig. 18. DOG nets and their discrete Gauss maps. The image of the Gauss map of a smooth orthogonal geodesic net is locally a point or a curve, and Gauss maps of smoother DOG nets are closer to being one dimensional, although this is not directly optimized for. From left to right: An input DOG net; smoothing the DOG by applying soft constraints on its lower horizontal boundary curve and minimizing E_H; the mesh after applying Catmull-Clark subdivision; subdivided mesh after another smoothing operation, which results in a thinner Gauss map.

support a wide range of topologies, unconstrained by ruling directions or any other global mesh information. Unlike the work of [Rabinovich et al. 2018], our system handles stretching deformations smoothly and intuitively (Fig. 16), supports straight creases and curved folds (Fig. 1, 13, 17, 15), and offers smoothing and subdivision operations (Fig. 18).

8.4 Curve-constraining flow

Our framework offers an interesting variant on deformations with positional constraints: flows induced by constraining whole curves on the surface. As in Sec. 7, we represent a curve intrinsically by a sequence of points on edges specified by a linear combination of vertices. We then choose target positions for the curve and compute an interpolation between the curve's initial and target coordinates. As the curve changes, the surface must change globally as well to satisfy the DOG constraints. We interpolate the curve geometrically by using a parameterization that is invariant to rigid motions: edge lengths l, curvature on inner vertices κ, and torsion τ on every inner edge $F_1 - F$:

$$\kappa = \frac{2\sin(\beta_1)}{\|F_1 - F_{\bar{1}}\|}, \quad \tau = \frac{\sin(\theta)}{\|F_1 - F\|}, \tag{16}$$

where θ is the turning angle between the osculating planes of the edge vertices [Hoffmann 2009]. We linearly interpolate the values of l, κ and τ between source and target. To translate the rotation invariant representation to Euclidean coordinates, we must specify a rigid motion, which we find using Procrustes to the coordinates of the previous curve on the mesh. Our curve interpolation method could be seen as an extension of the intrinsic 2D curve morphing of [Sederberg et al. 1993] to non-planar, 3D curves. As a final step, we feed the reconstructed positions of the curve as soft positional constraints to our flow algorithm, minimizing bending and grid regularity while interpolating the constraints (see Fig. 19). Frames 4-8 in Fig. 1 were also created using a curve-constraining flow.

9 CONCLUSIONS AND FUTURE WORK

This paper presents a discrete theory and an algorithm to continuously deform DOG nets, building upon the DOG model introduced in [Rabinovich et al. 2018].

Our technique for analyzing the linear dependency of the constraint gradients by studying circuit constraints graphs is quite general, and potentially can be extended to study other shape spaces, such as the space of conjugate or circular nets. Although our theory applies to connectivities containing holes, our shape space analysis does not apply to cylindrical shapes. This limitation stems from the assumption that no circular geodesics exist, used in Lemma 4.14.

The highly structured nature of DOG nets allows us to derive a specific Laplacian operator that converges under sampling; an interesting avenue for future research is the study of other Laplace operators for various classes of nets in discrete differential geometry.

A clear current limitation of our work is the speed for interactive editing, which is restricted to rather coarse models of around 1000 vertices, similarly to [Rabinovich et al. 2018]. On such meshes, the iteration speed is dominated by the LBFGS projection routine. We note however that one can use our subdivision operator concurrently while editing to achieve better mesh resolution.

Another limitation is the support for intricate crease patterns: currently we assume that each crease curve is simple and starts and ends at a mesh boundary, and we do not model intersecting creases and curved folds. Supporting such patterns requires special handling, since the areas around the crease intersections should most likely be discretized by triangles, as done in the ruling based triangular

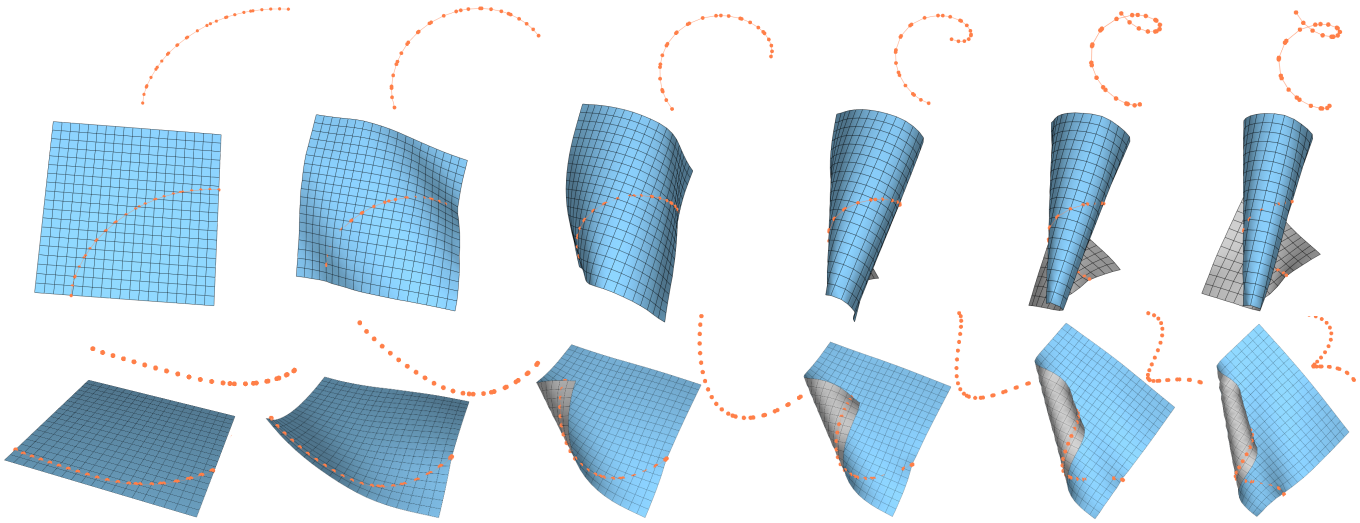


Fig. 19. Curve-constraining flows induced by fixing whole curves on the surface using positional constraints. We first represent a curve intrinsically as a set of points on edges, then design target curve positions and compute an interpolation between the curve's initial and target state. Finally, we feed the curve positions in each interpolation frame as soft positional constraints to our flow algorithm and optimize a bending and grid regularity objective.

model of [Stein et al. 2018]. Inspired by the same work, we leave it as future research to apply the theory of DOGs for approximation of arbitrary geometry using piecewise developable surfaces.

ACKNOWLEDGMENTS

The authors would like to thank Oliver Glauser, Chen Greif, Katja Wolff, Christian Schüller, Roi Poranne, Jan Wezel, Moritz Bächer and Christian Schumacher for illuminating discussions and help with results and figures production. The work was supported in part by the Deutsche Forschungsgemeinschaft-Collaborative Research Center, TRR 109, "Discretization in Geometry and Dynamics."

REFERENCES

M. Alexa and M. Wardetzky. 2011. Discrete Laplacians on general polygonal meshes. *ACM Trans. Graph.* 30, 4 (2011).

A. I. Bobenko and P. Schröder. 2005. Discrete Willmore Flow. In *Proc. Symposium on Geometry Processing*. http://dl.acm.org/citation.cfm?id=1281920.1281937

A. I. Bobenko and Y. B. Suris. 2008. *Discrete differential geometry: integrable structure*. Graduate studies in mathematics, Vol. 98. American Mathematical Society, Providence (R.I.).

S. Bouaziz, M. Deuss, Y. Schwartzburg, T. Weise, and M. Pauly. 2012. Shape-up: Shaping discrete geometry with projections. *Comput. Graph. Forum* 31, 5 (2012), 1657–1667.

S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly. 2014. Projective dynamics: fusing constraint projections for fast simulation. *ACM Trans. Graph.* 33, 4 (2014).

R. Burgoon, Z. J. Wood, and E. Grinspun. 2006. Discrete Shells Origami. In *Computers and Their Applications*.

K. Crane, U. Pinkall, and P. Schröder. 2013. Robust fairing via conformal curvature flow. *ACM Trans. Graph.* 32, 4 (2013).

T. A. Davis. 2011. Algorithm 915, SuiteSparseQR: Multifrontal multithreaded rank-revealing sparse QR factorization. *ACM Transactions on Mathematical Software (TOMS)* 38, 1 (2011), 8.

A. De Coninck, B. De Baets, D. Kourounis, F. Verbosio, O. Schenk, S. Maenhout, and J. Fostier. 2016. Needles: Toward Large-Scale Genomic Prediction with Marker-by-Environment Interaction. *Genetics* 203, 1 (2016), 543–555. https://doi.org/10.1534/genetics.115.179887 arXiv:http://www.genetics.org/content/203/1/543.full.pdf

E. D. Demaine, M. L. Demaine, V. Hart, G. N. Price, and T. Tachi. 2011a. (Non) existence of pleated folds: how paper folds between creases. *Graphs and Combinatorics* 27, 3 (2011), 377–397.

E. D. Demaine, M. L. Demaine, D. Koschitz, and T. Tachi. 2011b. Curved crease folding: a review on art, design and mathematics. In *Proceedings of the IABSE-IASS Symposium: Taller, Longer, Lighter*. 20–23.

E. D. Demaine and J. O'Rourke. 2007. *Geometric Folding Algorithms: Linkages, Origami, Polyhedra*. Cambridge University Press.

M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proc. ACM SIGGRAPH*. 317–324.

M. P. do Carmo. 1976. *Differential Geometry of Curves and Surfaces*. Prentice-Hall.

I. Eckstein, J.-P. Pons, Y. Tong, C.-C. Kuo, and M. Desbrun. 2007. Generalized surface flows for mesh processing. In *Proc. Symposium on Geometry Processing*. 183–192.

S. Fröhlich and M. Botsch. 2011. Example-Driven Deformations Based on Discrete Shells. *Comput. Graph. Forum* 30, 8 (2011), 2246–2257.

D. Fuchs and S. Tabachnikov. 1999. More on paperfolding. *The American Mathematical Monthly* 106, 1 (1999), 27–35.

W. C. Graustein. 1917. On the geodesics and geodesic circles on a developable surface. *Annals of Mathematics* 18, 3 (1917), 132–138.

R. F. Harik, H. Gong, and A. Bernard. 2013. 5-axis flank milling: A state-of-the-art review. *Computer-Aided Design* 45, 3 (2013), 796–808.

B. Heeren, M. Rumpf, P. Schröder, M. Wardetzky, and B. Wirth. 2014. Exploring the geometry of the space of shells. *Comput. Graph. Forum* 33, 5 (2014), 247–256.

B. Heeren, M. Rumpf, P. Schröder, M. Wardetzky, and B. Wirth. 2016. Splines in the space of shells. *Comput. Graph. Forum* 35, 5 (2016), 111–120.

T. Hoffmann. 2009. Discrete differential geometry of curves and surfaces. *COE Lecture Notes* 18 (2009).

D. A. Huffman. 1976. Curvature and creases: a primer on paper. *IEEE Trans. Computers* 25, 10 (1976), 1010–1019.

M. Kazhdan, J. Solomon, and M. Ben-Chen. 2012. Can mean-curvature flow be modified to be non-singular? *Comput. Graph. Forum* 31, 5 (2012), 1745–1754.

M. Kilian, S. Flöry, Z. Chen, N. J. Mitra, A. Sheffer, and H. Pottmann. 2008. Curved folding. *ACM Trans. Graph.* 27, 3 (2008), 75:1–75:9.

M. Kilian, N. J. Mitra, and H. Pottmann. 2007. Geometric modeling in shape space. *ACM Trans. Graph.* 26, 3 (2007).

M. Kilian, A. Monszpart, and N. J. Mitra. 2017. String actuated curved folded surfaces. *ACM Trans. Graph.* 36, 3 (2017), 25:1–25:13.

D. Kourounis, A. Fuchs, and O. Schenk. 2018. Towards the Next Generation of Multiperiod Optimal Power Flow Solvers. *IEEE Transactions on Power Systems* PP, 99 (2018), 1–10. https://doi.org/10.1109/TPWRS.2017.2789187

Y. Liu, H. Pottmann, J. Wallner, Y.-L. Yang, and W. Wang. 2006. Geometric modeling with conical meshes and developable surfaces. *ACM Trans. Graph.* 25, 3 (2006).

J. Mitani and T. Igarashi. 2011. Interactive design of planar curved folding by reflection. In *Proc. Pacific Graphics, Short Papers*.

MOSEK ApS. 2017. *The MOSEK optimization toolbox for MATLAB manual. Version 8.1.* http://docs.mosek.com/8.1/toolbox/index.html

R. Narain, T. Pfaff, and J. F. O'Brien. 2013. Folding and Crumpling Adaptive Sheets. *ACM Trans. Graph.* 32, 4 (2013).

J. Nocedal. 1980. Updating quasi-Newton matrices with limited storage. *Math. Comp.* 35, 151 (1980), 773–782.

J. Nocedal and S. J. Wright. 2006. *Sequential quadratic programming.* Springer.
Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, and L. Liu. 2018. Anderson Acceleration for Geometry Optimization and Physics Simulation. *ACM Trans. Graph.* 37, 4, Article 42 (July 2018), 14 pages. https://doi.org/10.1145/3197517.3201290
U. Pinkall and K. Polthier. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics* 2, 1 (1993), 15–36.
H. Pottmann and J. Wallner. 2001. *Computational line geometry.* Springer, Berlin, Heidelberg, New York.
M. Rabinovich, T. Hoffmann, and O. Sorkine-Hornung. 2018. Discrete Geodesic Nets for Modeling Developable Surfaces. *ACM Trans. Graph.* 37, 2 (2018), 16.
T. W. Sederberg, P. Gao, G. Wang, and H. Mu. 1993. 2-D shape blending: an intrinsic solution to the vertex path problem. In *Proc. ACM SIGGRAPH.* 15–18.
J. Solomon, E. Vouga, M. Wardetzky, and E. Grinspun. 2012. Flexible developable surfaces. *Comput. Graph. Forum* 31, 5 (2012), 1567–1576.
O. Sorkine and M. Alexa. 2007. As-rigid-as-possible surface modeling. In *Proc. Symposium on Geometry Processing.* 109–116.
O. Stein, E. Grinspun, and K. Crane. 2018. Developability of triangle meshes. *ACM Trans. Graph.* 37, 4 (2018).
G. Sundaramoorthi, A. Yezzi, and A. C. Mennucci. 2007. Sobolev active contours. *International Journal of Computer Vision* 73, 3 (2007), 345–366.
T. Tachi. 2009. Simulation of rigid origami. *Origami* 4 (2009), 175–187.
C. Tang, P. Bo, J. Wallner, and H. Pottmann. 2016. Interactive design of developable surfaces. *ACM Trans. Graph.* 35, 2, Article 12 (2016), 12 pages.
F. Verbosio, A. D. Coninck, D. Kourounis, and O. Schenk. 2017. Enhancing the scalability of selected inversion factorization algorithms in genomic prediction. *Journal of Computational Science* 22, Supplement C (2017), 99 – 108. https://doi.org/10.1016/j.jocs.2017.08.013
M. Wardetzky. 2007. *Discrete differential operators on polyhedral surfaces – convergence and approximation.* Ph.D. Dissertation. Freie Universität Berlin.
M. Wardetzky, S. Mathur, F. Kälberer, and E. Grinspun. 2007. Discrete Laplace operators: no free lunch. In *Proc. Symposium on Geometry Processing.* 33–37.
Y.-L. Yang, Y.-J. Yang, H. Pottmann, and N. J. Mitra. 2011. Shape space exploration of constrained meshes. *ACM Trans. Graph.* 30, 6 (2011).

A PROOF OF THEOREM 4.2

The quantities $l_1, l_2, l_3, l_4, \beta_1, \beta_2$ uniquely define a star with all angles equal, up to rigid motion. Indeed, take two discrete curves (i.e., two chains of 2 edges each) with edge lengths l_1, l_3 and l_2, l_4, respectively, and angles β_1, β_2, respectively. Place both center vertices at the origin, and observe the discrete Frenet frames of both curves, as defined in [Rabinovich et al. 2018]. Rotate one of the curves such that its discrete principal normal matches the other and the tangents and binormals coincide. This guarantees that all angles around the star are equal, and this rotation is unique unless the curve is straight, in which case there is a rotational freedom, but the geometry of the resulting star remains the same. □

B PROOF OF LEMMA 4.4

The derivative $\frac{\partial \cos(\alpha_9)}{\partial F_1}$ is clearly not vanishing, since there are directions to move F_1 so that α_9 changes. The angle and its cosine are defined by the triangle formed by points F, F_1, F_2, and by the chain rule, the derivative of any function on this triangle w.r.t. one of its vertices has to lie in the plane of the triangle. The angle α_9 remains constant if F_1 is moved along the direction $F_1 - F$, hence the gradient $\frac{\partial \cos(\alpha_9)}{\partial F_1}$ must be perpendicular to this direction. □

C PROOF OF THEOREM 6.1

Since f is a geodesic net, the principal normals of f_x, f_y are parallel to the surface normal n. Hence $f_{xx} = af_x + bn$, $f_{yy} = cf_y + dn$ for some $a, b, c, d \in \mathbb{R}$ and from $f_x \perp f_y$ we get that $\langle f_{xx}, f_y \rangle = \langle f_{yy}, f_x \rangle = 0$. Using the orthogonality $f_x \perp f_y$ again, we get that $0 = \langle f_x, f_y \rangle_x = \langle f_{xx}, f_y \rangle + \langle f_x, f_{xy} \rangle$ and so $f_{xy} \perp f_x$. Therefore $\langle f_x, f_x \rangle_y = 2\langle f_x, f_{xy} \rangle = 0$ and similarly $2\langle f_y, f_{xy} \rangle = 0$ and so $\|f_x\|_y = \|f_y\|_x = 0$. □

D LAPLACIAN DERIVATIONS AND PROPERTIES

D.1 Derivation of the Laplacian

We first compute the gradient of the area of a single DOG quad, defined in Sec. 6.1 as $Q = \frac{1}{4}(l_1 + l_3)(l_2 + l_4)$ using the notation of Fig. 12. We note that:

$$\frac{\partial \|F - F_1\|}{\partial F} = \frac{F - F_1}{\|F - F_1\|}. \tag{17}$$

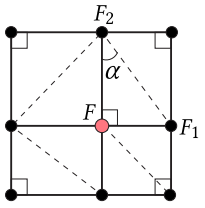
Plugging this in and using the chain rule leads to

$$\begin{aligned}\frac{\partial Q}{\partial F} =& \frac{1}{4}\frac{\partial(\|F - F_1\| + \|F_{12} - F_2\|)(\|F - F_2\| + \|F_{12} - F_1\|)}{\partial F} = \\ & \frac{1}{4}\left(\frac{\|F - F_2\| + \|F_{12} - F_1\|}{\|F - F_1\|}(F - F_1) + \right. \\ & \left. \frac{\|F - F_1\| + \|F_{12} - F_2\|}{\|F - F_2\|}(F - F_2)\right).\end{aligned} \tag{18}$$

Eq. (7) now follows by summing up the contribution of all faces incident on F and rearranging the terms per edge.

D.2 Equivalence to cotan weights on planar nets

If a DOG inner vertex and its 8 surrounding neighbors lie on a plane, the angles around the star are $\frac{\pi}{2}$. Observe the triangulated quads in the inset. Note that $\cot(\alpha) = \frac{\|F_2 - F\|}{\|F_1 - F\|}$ and $\cot(\frac{\pi}{2}) = 0$, therefore the edge weights of the DOG Laplacian correspond to the edge weights of the cotan Laplacian [Pinkall and Polthier 1993] for any triangulation of the planar orthogonal grid. The cotan Laplacian applied to a planar mesh vanishes, hence so does the DOG Laplacian $\mathbb{L}$, meaning that it satisfies the linear precision property.



D.3 Proof of Lemma 6.6

Let Q_1, Q_2, Q_3, Q_4 be the quad areas around vertex F, as denoted in Fig. 11. Assuming the DOG is Chebyshev, the following holds:

$$\begin{aligned}\|F_{12} - F_1\| &= \|F_2 - F\| = \|F_{\bar{1}\bar{2}} - F_{\bar{1}}\|, \\ \|F_1 - F_{1\bar{2}}\| &= \|F - F_{\bar{2}}\| = \|F_{\bar{1}} - F_{\bar{1}\bar{2}}\|, \\ \|F_{12} - F_2\| &= \|F_1 - F\| = \|F_{1\bar{2}} - F_{\bar{2}}\|, \\ \|F_2 - F_{\bar{1}2}\| &= \|F - F_{\bar{1}}\| = \|F_{\bar{2}} - F_{\bar{1}\bar{2}}\|.\end{aligned} \tag{19}$$

Plugging this into Eq. (18), we get:

$$\begin{aligned}\frac{\partial Q_1}{\partial F} &= -\frac{\|F - F_2\|}{2}\delta_1 F - \frac{\|F - F_1\|}{2}\delta_2 F, \\ \frac{\partial Q_2}{\partial F} &= -\frac{\|F - F_{\bar{1}}\|}{2}\delta_2 F - \frac{\|F - F_2\|}{2}\delta_{\bar{1}} F, \\ \frac{\partial Q_3}{\partial F} &= -\frac{\|F - F_{\bar{2}}\|}{2}\delta_{\bar{1}} F - \frac{\|F - F_{\bar{1}}\|}{2}\delta_{\bar{2}} F, \\ \frac{\partial Q_4}{\partial F} &= -\frac{\|F - F_1\|}{2}\delta_{\bar{2}} F - \frac{\|F - F_{\bar{2}}\|}{2}\delta_1 F.\end{aligned} \tag{20}$$

And therefore:

$$\frac{\partial \mathcal{A}}{\partial F} = \sum_{i=1}^{4}\frac{\partial Q_i}{\partial F} = -\left(s^2\left(\delta_1 + \delta_{\bar{1}}\right) + s^1\left(\delta_2 + \delta_{\bar{2}}\right)\right), \tag{21}$$

with $s^2 = \frac{1}{2}\left(\|F - F_2\| + \|F - F_{\bar{2}}\|\right)$ and $s^1 = \frac{1}{2}\left(\|F - F_1\| + \|F - F_{\bar{1}}\|\right)$. By Eq. (10), the above is a linear combination of two vectors parallel to N. Plugging in the vertex area $A = s^1 s^2$, we get:

$$\frac{\partial \mathcal{A}}{\partial F} = -s^1 s^2 \left(\frac{\delta_1 F + \delta_{\bar{1}} F}{s^1} + \frac{\delta_2 F + \delta_{\bar{2}} F}{s^2}\right) = \tag{22}$$
$$= -A\left(\frac{2\,\|\delta_1 F + \delta_{\bar{1}} F\|}{\|F - F_1\| + \|F - F_{\bar{1}}\|} + \frac{2\,\|\delta_2 F + \delta_{\bar{2}} F\|}{\|F - F_2\| + \|F - F_{\bar{2}}\|}\right) N.$$

□

E PROOF OF THEOREM 6.9

To shorten notation, we remove the explicit ϵ, denoting $K^1 = K^1_\epsilon$, $f_1 = f_1(\epsilon)$, etc. We first prove that the normals and mean curvature converge by showing that the curves' curvature normals $K^1 N,\ K^2 N$ converge under sampling. For $K^1 N$ this amounts to computing the limit:

$$\lim_{\epsilon\to 0} K^1 N = \lim_{\epsilon\to 0} 2\frac{\delta_1 f + \delta_{\bar{1}} f}{\|f_1 - f\| + \|f_{\bar{1}} - f\|}. \tag{23}$$

We denote the normalized curve tangent as $t = \frac{f_x}{\|f_x\|}$, the curve's curvature by κ and its principal normal by n. We use the Taylor expansion of f to write its nearby points. By Appendix A in [Rabinovich et al. 2018]:

$$\begin{aligned}
\delta_1 f &= \frac{f_x}{\|f_x\|} + \epsilon\left(-\frac{\langle f_{xx}, f_x\rangle}{2\langle f_x, f_x\rangle^{3/2}} f_x + \frac{f_{xx}}{2\|f_x\|}\right) + o(\epsilon^2),\\
\delta_{\bar{1}} f &= -\frac{f_x}{\|f_x\|} + \epsilon\left(-\frac{\langle f_{xx}, f_x\rangle}{2\langle f_x, f_x\rangle^{3/2}} f_x + \frac{f_{xx}}{2\|f_x\|}\right) + o(\epsilon^2),\\
\delta_1 f + \delta_{\bar{1}} f &= 2\epsilon\left(-\frac{\langle f_{xx}, f_x\rangle}{2\langle f_x, f_x\rangle^{3/2}} f_x + \frac{f_{xx}}{2\|f_x\|}\right) + o(\epsilon^2).
\end{aligned} \tag{24}$$

Let $\lambda = \|f_x\|$. We can write the derivatives of f in terms of the tangent and principal normal of f:

$$f_x = \lambda t,\ f_{xx} = \lambda_x t + \lambda^2 \kappa n,\ \lambda_x = \langle f_x, f_x\rangle_x = \frac{\langle f_{xx}, f_x\rangle}{\|f_x\|}. \tag{25}$$

Plugging this in Eq. (24), we get

$$\begin{aligned}
\delta_1 f + \delta_{\bar{1}} f &= 2\epsilon\left(-\frac{\lambda\langle f_{xx}, f_x\rangle}{2\langle f_x, f_x\rangle^{3/2}} t + \frac{\lambda_x}{2\|f_x\|} t + \frac{\lambda^2\kappa}{2\|f_x\|} n\right) + o(\epsilon^2) =\\
&= 2\epsilon\left(-\frac{\|f_x\|\langle f_{xx}, f_x\rangle}{2(\|f_x\|^2)^{3/2}} t + \frac{\langle f_{xx}, f_x\rangle}{2\|f_x\|^2} t + \frac{\lambda\kappa}{2} n\right) + o(\epsilon^2) =\\
&= \epsilon\lambda\kappa n + o(\epsilon^2).
\end{aligned}$$

Plugging that into Eq. (23) results in

$$\lim_{\epsilon\to 0} K^1 N = \lim_{\epsilon\to 0} 2\frac{\epsilon\lambda\kappa n}{\|f_1 - f\| + \|f_{\bar{1}} - f\|} = \lim_{\epsilon\to 0} \frac{2\epsilon\lambda\kappa n}{2\lambda\epsilon + o(\epsilon^2)} = \kappa n.$$

This proves the convergence of $K^1 N$, and the proof for $K^2 N$ is analogous. The convergence of the mean curvature normal HN follows by linearity. Since f is an orthogonal geodesic net, by Theorem 6.1 it is also a Chebyshev net satisfying $\|f_x\|_y = \|f_y\|_x = 0$, and therefore Eq. (19) is satisfied up to second order, i.e. $\|f_1 - f\| = \|f_{12} - f_2\| + o(\epsilon^2)$, etc. Hence the principal normal and the curvature vector for the f_x direction given by the Laplacian is

$$K^{1*} N^* = 2\left(\frac{\delta_1 f}{\|f_1 - f\| + \|f_{\bar{1}} - f\| + o(\epsilon^2)} + \frac{\delta_{\bar{1}} f}{\|f_1 - f\| + \|f_{\bar{1}} - f\| + o(\epsilon^2)}\right),$$

and by a similar limit calculation we get that the mean curvature normal of the DOG Laplacian converges. □