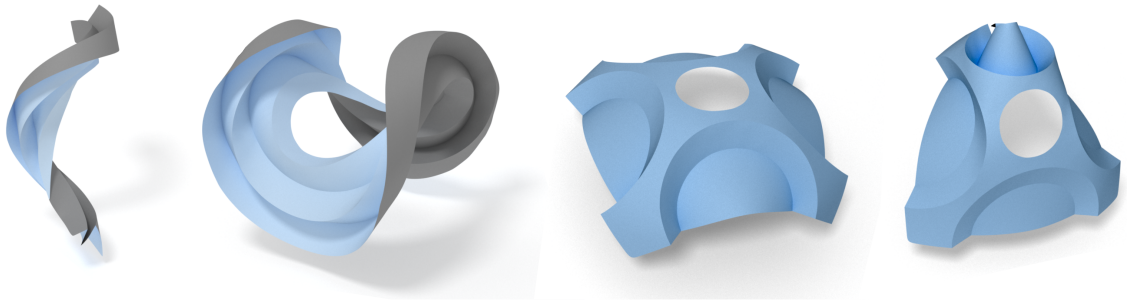
Diss. ETH No. 26488

Modeling Developable Surfaces with Discrete Orthogonal Geodesic Nets

A thesis submitted to attain the degree of
Doctor of Sciences of **ETH Zurich**
(Dr. sc. ETH Zurich)



presented by
Michael Rabinowitz
MSc in Computer Science, ETH Zurich
born on 09.09.1989
citizen of Israel

accepted on the recommendation of
Prof. Dr. Olga Sorkine-Hornung, examiner
Prof. Dr. Tim Hoffmann, co-examiner
Prof. Dr. Helmut Pottmann co-examiner

2020

Abstract

Surfaces that are locally isometric to a plane are called developable surfaces. In the physical world, these surfaces can be formed by bending thin flat sheets of material, which makes them particularly attractive in manufacturing, architecture and art. Consequently, the design of freeform developable surfaces has been an active research topic in computer graphics, computer aided design, architectural geometry and computational origami for several decades.

This thesis presents a discrete theory and a set of computational tools for modeling developable surfaces. The basis of our theory is a discrete model termed discrete orthogonal geodesic nets (DOGs). DOGs are regular quadrilateral meshes satisfying local angle constraints, extending the rich theory of nets in discrete differential geometry. Our model is simple, local, and, unlike previous works, it does not directly encode the surface rulings. Thus, DOGs can be used to model continuous deformations of developable surfaces independently of their decomposition into torsal and planar patches or the surface topology.

We start by examining the "locking" phenomena common in computational models for developable surfaces, which was the primary motivation behind our work. We then follow up with the derivation and definitions behind our solution - DOGs, while theoretically and empirically demonstrating the connection between our model and its smooth counterpart and its resilience to the locking problem. We prove that every sampling of the smooth counterpart satisfies our constraints up to second order and establish connections between DOGs and other nets in discrete differential geometry.

We then develop a theoretical and computational framework for deforming DOGs. We first derive a variety of geometric attributes on DOGs, including notions of normals, curvatures, and a novel DOG Laplacian operator. These can be used as objectives for various modeling tasks. By utilizing the regular nature of our model, our discrete quantities are simple yet precise, and we discuss their convergence.

We then study the DOG constraints, via looking at continous deformations on DOGs. We characterizae 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. The study

of the shape space leads to a better understanding of the flexibility and rigidity of DOGs, and we devote an entire chapter to examining various notions of isometries on DOGs and a novel model termed "discrete orthogonal 4Q geodesic net".

We further show how to extend the shape space of DOGs by supporting creases and curved folds. We derive a discrete binary charactarization for folds between discrete developable surfaces, accompanied by an algorithm to simultaneously fold creases and smoothly bend planar sheets. We complement our algorithm with essential building blocks for curved folding deformations: objectives to control dihedral angles and mountain-valley assignments.

We apply our theory and resulting set of tools in the first interactive editing system for developable surfaces that supports arbitrary bending, stretching, cutting, (curved) folds, as well as smoothing and subdivision operations.

Zusammenfassung

Flächen, die lokal zu einer Ebene isometrisch sind, werden als abwickelbare Flächen bezeichnet. Physikalisch lassen sich diese Flächen durch das Biegen dünner Materialschichten realisieren, wodurch sie insbesondere für die Industrie, Architektur und Kunst attraktiv werden. Aufgrund der vielfältigen Verwendungsgebiete wurde in den letzten Jahrzehnten intensiv in den Bereichen Computergrafik sowie rechnergestütztes Konstruieren, Architektur und Origami zu diesem Thema geforscht.

Diese Dissertation präsentiert eine diskrete Theorie sowie konkrete Algorithmen zur Modellierung diskreter abwickelbarer Flächen. Die Grundlage dieser Theorie bildet ein diskretes Modell, die sogenannten diskreten orthogonalen geodätischen Netze (DOGs). DOGs sind reguläre Vierecksnetze, die lokale Winkelbedingungen erfüllen und somit eine Erweiterung der Theorie diskreter differentialgeometrischer Netze darstellen. Unser Modell ist einfach, lokal und beschreibt, im Gegensatz zu anderen Arbeiten, die sogenannten Rulings nicht explizit. Daher können DOGs kontinuierliche Deformationen, unabhängig von ihrer Zerlegung in torsale und planare Stücke oder ihrer Oberflächentopologie, modellieren.

Wir beginnen mit einer detaillierten Analyse des Locking Phänomens, das häufig im Zusammenhang mit digitalen Modellen abwickelbarer Flächen auftritt. Diese Problematik war eine der Hauptmotivationen für diese Arbeit. Anschliessend beschreiben wir die Definition und Herleitung von DOGs und demonstrieren den Zusammenhang des diskreten Modells mit kontinuierlichen abwickelbaren Flächen sowohl theoretisch als auch praktisch. Darüber hinaus zeigen wir, dass unser Modell nicht besonders anfällig für Locking ist. Wir beweisen, dass jede Abtastung einer kontinuierlichen abwickelbaren Fläche unsere Bedingung bis zur zweiten Ordnung erfüllt und beschreiben den Zusammenhang von DOGs und weiteren Netzen aus dem Bereich der diskreten Differentialgeometrie.

Als nächstes entwickeln wir einen theoretischen und algorithmischen Rahmen für die Verformung von DOGs. Wir leiten zunächst eine Vielzahl von geometrischen Attributen für DOGs ab, einschliesslich der Begriffe Normale und Krümmung sowie die Definition eines DOG Laplace Operators. Diese Objekte können als Zielfunktionen für verschiedene Modellierungsaufgaben verwendet werden. Durch das Ausnutzen der regulären Struktur unseres Modells, lassen sich diese Objekte einfach und präzise

ausdrücken. Weiterhin diskutieren wir die Konvergenz dieser diskreten Objekte in Bezug auf kontinuierliche Flächen.

Anschliessend untersuchen wir mithilfe kontinuierlicher Deformationen Nebenbedingungen für DOGs. Wir charakterisieren den geometrischen Konfigurationsraum von DOGs bei gegebener Konnektivität und zeigen, dass sich dieser Raum überall, abgesehen von einer Menge von Singularitäten, lokal durch Mannigfaltigkeiten einer bestimmten Dimension beschreiben lässt. Daraus folgt, dass sich DOGs kontinuierlich deformieren lassen. Glatte Flüsse können durch eine glatte Menge von Vektoren in den Tangentialräumen der Mannigfaltigkeit konstruiert werden, die so ausgewählt werden, dass eine gewünschte Zielfunktion unter einer gegebenen Metrik minimiert wird. Die Untersuchung des Konfigurationsraums führt zu einem besseren Verständnis der Flexibilität und Rigidität von DOGs. Ein ganzes Kapitel widmen wir der Untersuchung verschiedener Begriffe von Isometrie an DOGs und einem neuartigen Modell, das als "diskretes orthogonales geodätisches 4Q-Netz"bezeichnet wird.

Wir zeigen ausserdem, wie der Konfigurationsraum von DOGs mithilfe von Falten und gekrümmten Falten erweitert werden kann. Wir leiten eine diskrete binäre Charakterisierung für Falten zwischen diskreten abwickelbaren Oberflächen her und entwickeln einen Algorithmus zum gleichzeitigen Falten und glatten Biegen ebener Flächen. Wir ergänzen unseren Algorithmus um wesentliche Bausteine für die Optimierung gekrümmter Faltungsdeformationen: Zielfunktionen zur Kontrolle von dihedralen Winkeln und Berg-Tal-Zuordnungen.

Wir wenden unsere Theorie und die daraus resultierenden Werkzeuge in der ersten interaktiven Software für das Design abwickelbarer Flächen an, die beliebiges Biegen, Strecken, Schneiden, (gekrümmtes) Falten sowie Glätten und Unterteilen unterstützt.

Acknowledgments

I have never thought I will end up doing a PhD. I had a great time, and would like to start by thanking all the people responsible for the sequence of random events that got me here. Thank you mom and dad, for suggesting I'll do a bachelors in mathematics. Thanks to Yoni Stanvescu, who was my "Number Theory" professor. Yoni sent an email to the course students, advertising a program in the Weizmann Institute for excellent bachelor students in sciences. I've ended up in the "Young Weizmann Scholars" program, and through that met Yaron Lipman. Thank you Yaron, for introducing me to geometry processing and guiding me through my first steps in research. Yaron's lab was, and still is, a great place to do research and witness mathematical wizards tackling practical geometry processing problems. Thank you Noam Aigerman and Shahar Kovalsky, who have been at the time in Yaron's lab, and were extremely hospitable and fun to be around with. Thanks to Yaron, I've ended up in the Interactive Geometry Lab (IGL) under Olga Sorkine-Hornung, who was generous enough to give me a scholarship and has advised me both during my master degree as well as my PhD.

Thank you Olga, a thousand times. They say the most important thing in a course of a PhD is to have a good advisor. I couldn't agree more, and I couldn't be more lucky. I had an advisor that is extremely patient, caring, and supporting person while also being an extraordinarily sharp, knowledgeable, and experienced researcher who does not compromise on the quality or clarity of her work and always asks the right questions.

Thank you Tim Hoffmann. Without you as a mentor and a collaborator, this thesis would never have been possible. Thank you for introducing me to discrete differential geometry, for our whiteboards sessions in Zurich, Munich and Berlin. For endless hours spent in Skype and emails, and for being the mixture of creative and rigorous mathematician that you are.

Thank you Roi Poranne and Daniele Panozzo, who together with Olga advised me in my master thesis. You were an endless source of knowledge through my master studies and afterwards, as I continued to follow your work and bug you with questions during my PhD.

Thanks to my officemates, Oliver and Katja for listening to my gibberish, helping with figures, colors, german translations of letters and thesis abstracts, and answering every

morning to me saying 'Wassssup?'. Thanks to my more recent collaborators: Floor Verhoeven, Alexandra Ion, Philipp Herholz and Amir Vaxman. It was a pleasure to work with you, and I'm looking forward to see our final products.

I'd like to thank my other friends and past or present members of IGL: Christian, Olga (Diamanti), Yifan, Shiaho, Victor and Marianna. For the good times in IGL-lunch, and for the lively social and professional environment that made the experience more gratifying.

Thank you Helmut Pottmann, for participating in my PhD committee. More importantly, thank you for being a research role model. I've always been a big fan of your work, as evident by this thesis. Olga used to frequently tease me on that, commenting on the fact that we are writing our DOG papers mostly for you to read. Funny as it is, this statement is quite accurate.

Giving presentations to other groups has been a delight, and I am very thankful to Leo Guibas, Denis Zorin, Mark Pauly, Justin Solomon, Mark Meyer, Jernej Barbic, Hao Li and Daniel Cohen-Or for inviting me to give talks in their labs. I'd also like to thank Johannes Wallner, Alexander Bobenko and Helmut Pottmann for inviting me to give talks in the geometry workshops at Obergurgl and in Strobl.

I have been fortunate to have the opportunity to discuss my research with other great minds, whose work I have followed closely throughout my PhD. Thank you Peter Schroeder and Mathieu Desbrun for inviting me to Caltech, and for sharing your thoughts and ideas on our work as well as on geometry processing in large. Thank you Erik Demaine and Tomohiro Tachi for our meetings in Tokyo and Boston, and for invaluable discussions on developable surfaces, origami and curved folding.

I would like to thank my friends in Zurich, who made this period unforgettable and full of beautiful experiences; Summer days spent around the river or Zurich lake, winter evenings at Bar3000/Parkplatz/Kasheme, snowboarding, clubbing, and film nights. I've formed lifelong friends.

I would like to thank my friends back in Tel Aviv for the good times I've spent during my frequent visits home; For their hummus companionships, verbal sparring in the sun, lower than average chess skills, Rothschild street night expeditions and their moral support.

This thesis is dedicated to my family; My parents Dafna and Leo, who dedicated their life for my upbringing. My older brothers, Eyal and Zvi, who always served as guides and role models. My younger brother Alon, who has a lot going on now, and just makes feel proud. For the good times we always have together.

Contents

CHAPTER 1

Introduction

Mathematics is like love;
a simple idea, but it can get complicated.

George Polya

The object of study in this thesis is, undoubtedly, quite simple. Discrete orthogonal geodesic nets, or DOGs, are regular quadrilateral meshes with equal angles around every vertex.

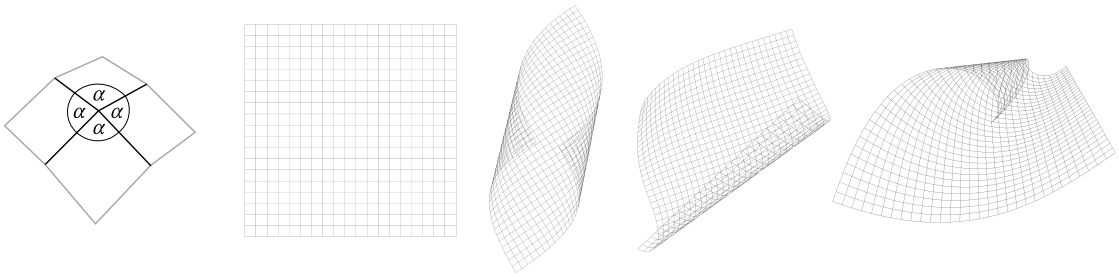


Figure 1.1: *Discrete orthogonal geodesic nets.*

DOGs are discrete analogues of *developable surfaces*, parameterized through orthogonal geodesics. In the rest of the thesis lies the author's attempt to persuade readers that DOGs are both useful and interesting, while developing relevant mathematical theory and a set of computational tools for modeling DOGs.

1.1 Developable surfaces

Imagine holding a planar sheet. You can roll it into a cylinder, bend it into a cone, and deform it to a variety of other shapes. As long as one does not stretch the surface, the generated shape is said to be *isometric* to the planar surface. Such shapes are said to be *developable*, because one can develop them onto the plane.

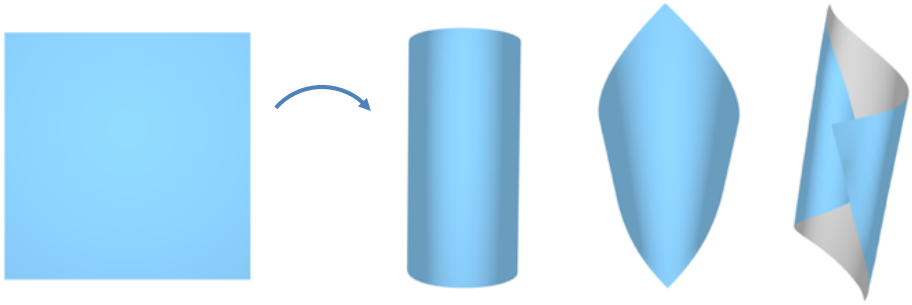


Figure 1.2: *Isometrically deforming a planar sheet.*

The notion of developability is local. A smooth surface is said to be developable if it is locally isometric to the plane. This means that for every point on the surface, one can cut a small neighborhood around it such that this neighborhood is then isometric to a planar surface (Fig. 1.3 left).

Most shapes however, are not developables. Gauss' Theorema Egregium coupled with Minding's theorem [do Carmo 1976] shows that smooth developable surfaces are exactly those with with zero Gaussian curvature. The majority of surfaces found in nature or designed by humans have non-zero Gaussian curvature, and indeed if one tries to wrap most surfaces with a piece of paper the result is often creased, with many parts that are not perfectly wrapped (see (Fig. 1.3 right).

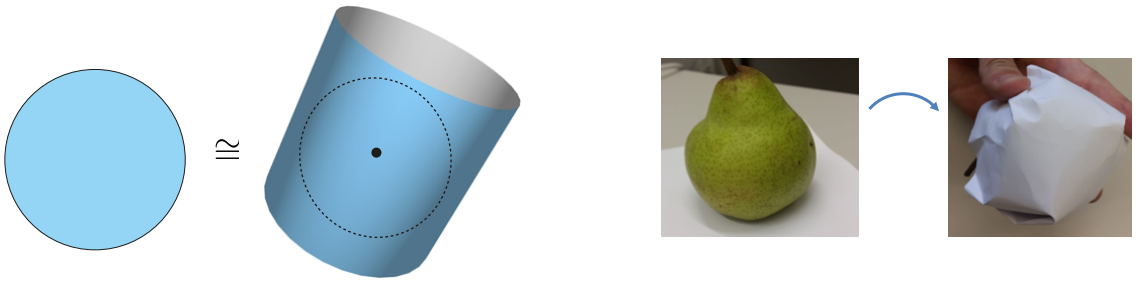


Figure 1.3: *Left: A developable surface is locally isometric to the plane. Right: Most surfaces are not developables, hence one cannot wrap them by bending a flat sheet.*

Though a small subset of surfaces, developable surfaces are useful in practice. In the physical world, these surfaces can be formed by bending thin flat sheets of material,

which makes them particularly attractive in manufacturing [Pérez and Suárez 2007; Harik et al. 2013], architecture [Shelden 2002; Adler 2004] and art [Wertheim 2004; Demaine et al. 2011c].

Figure 1.4: *Developable surfaces are easy to build, but hard to design. Developable surfaces are prominent in the manufacturing industry and in architecture. Left - ship hull design by [Pérez and Suárez 2007], center - bending of metal by Bernshaws, right - Frank Gehry's Walt Disney concert hall (photo by Jon Sullivan).*

1.2 Modeling developable surfaces

A developable surface is easy to build, 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 also 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). Consequently, the design of freeform developable surfaces has been an active research topic in computer graphics, computer aided design and computational origami for several decades.

The primary difficulty in modeling developable surfaces is finding a discrete model that is able to capture the full set of deformations while keeping the surface developable. A failure of a discrete model to represent the full range of smooth deformations is often termed *locking* [Solomon et al. 2012; Chapelle and Bathe 1998] and is the bane of most discrete developable models. Deformations can be extrinsic as well as intrinsic. The latter stretch the surface while keeping it developable, and are used for geometry exploration tasks where the size and shape of the flattened developable surface is unknown (see Fig. 1.5).

Figure 1.5: *Intrinsic deformations of a developable surface. 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. Strictly modeling isometries in our model is tackled in Chapter 6 and modeled in Chapter 7.*

Ruling based models [Liu et al. 2006; Kilian et al. 2008; Tang et al. 2016; Stein et al. 2018; Solomon et al. 2012] are limited to a partial set of extrinsic deformations, while isometry based methods [Grinspun et al. 2003; Burgoon et al. 2006; Goldenthal et al. 2007; Fröhlich and Botsch 2011] do not model intrinsic deformations by design, and are also prone to locking of various bending deformations [Chapelle and Bathe 1998; Alessio 2012], and often must be coupled with dynamic remeshing [Narain et al. 2012; Kilian et al. 2017; Narain et al. 2013; Schreck et al. 2015].

Our goal in this thesis 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. The thesis mostly deals with smooth deformations on developables, such as the rolling and bending of a planar sheet into a cone, rather than C^0 origami-like folding and creasing [Tachi 2010], however the last chapter of this thesis deals with designing piecewise smooth developable surfaces with classical origami folds, as well as curved folded surfaces.

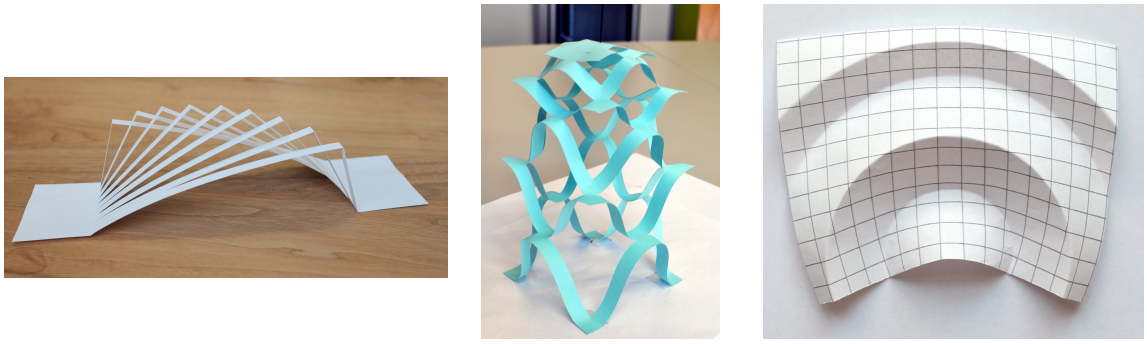


Figure 1.6: *Though the focus of this thesis is modeling smooth deformtions of developable surfaces, our tools and editing system developed throughout the thesis supports the modeling of cuts, folding, as well as curved folding.*

The basis of our theory is the model of discrete orthogonal geodesic nets.

1.3 A discrete theory with an eye towards computation

There is often more than one way to discretize a given geometry. In the case of a developable surface, as we will see in Chapter 2, there are multiple equivalent conditions for a smooth surface to be developable. These conditions can be formulated as equations, which in turn can be discretized, say by finite differences, and a mesh satisfying these constraints can be seen as a discrete model for a developable surface. While these equations are all equivalent for smooth surfaces, discrete objects satisfying these discrete conditions can behave very differently, a fact that lies at the core of discrete differential geometry [Bobenko and Suris 2008].

In essence, some discrete models are better than others, while some models are better suited for various tasks. For the purpose of freeform editing of developables, we seek a discrete model that is:

- Precise
- Amenable to optimization

A precise discrete model is one that does not only converge in some smooth limit to its smooth counterpart, but also one that behaves well as a discrete object. In practice, one often works with a limited set of resources and a good discrete model has similar behaviour and structure to its smooth counterpart, even on a rather coarse mesh. The problem of *locking* in various models of discrete developable surfaces can be addressed from that angle. Moreover, as we'll see in Chapter 2, often locking cannot be solved by using a denser mesh.

A model that is amenable to optimization is one that is simple and can be plugged into a well defined and well behaved optimization problem. Simplicity is crucial from both a practical implementation side, but also theoretically. It it often a necessity if one wants to fully understand an optimization problem and to have some practical guarantees, for instance on the existence of a solution for a given problem.

The model of discrete orthogonal geodesic net and its accompanying theory was conceived with these considerations in mind, and was developed hand in hand with a set of computational tools for solving practical geometry processing tasks.

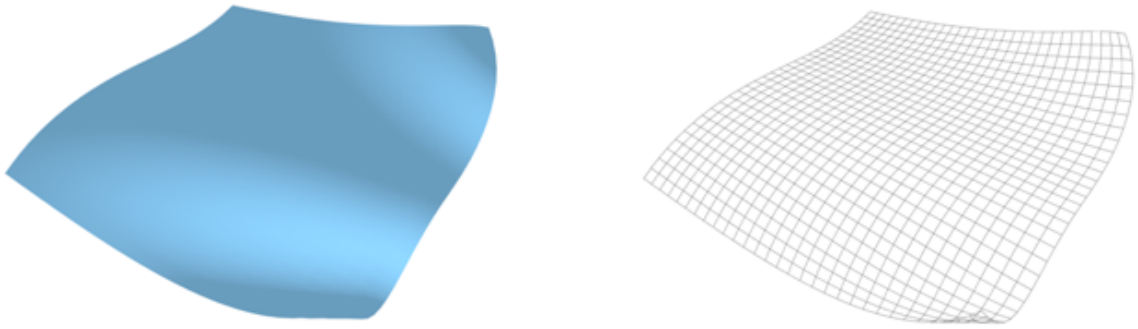


Figure 1.7: *This thesis presents DOGs as discrete model for developable surfaces, and develops the theory of DOGs together with applied tools to model them.*

1.4 Contributions of this thesis

Our contributions can be largely categorized into the following:

- We introduce discrete orthogonal geodesic nets (DOGs), study their properties and prove theorems about them in the framework of discrete differential geometry.
- We study smooth and discrete DOG deformations and apply our results into an optimization framework.
- We propose a simple theoretically grounded solution to the locking problem prominent in modeling isometries of planar sheets, based on DOGs.
- We extend our toolset to allow for modeling of creases and curved folds.
- We apply our tools to develop the first interactive editing system for developable surfaces that supports bending, creasing, cutting, and (curved) folding, on various topologies, and also includes smoothing and subdivision operations.

1.4.1 Structure of the thesis

This thesis is organized in eight chapters. The remaining chapters are organized as follows:

Chapter 2 introduces the problem of locking when modeling developables and isometries of planar surfaces. It explains locking in various models of discrete developables from both an optimization point of view, as well as a differential geometry perspective by looking at deformations on smooth nets, while also providing a short introduction to nets in discrete differential geometry.

Chapter 3 presents discrete orthogonal geodesic nets as a discrete model for developable surfaces. The chapter starts by deriving the DOG model and follows up with an analysis of its relation to smooth orthogonal geodesic nets and to several nets in discrete differential geometry. At the end of the chapter we show how the model can be used to develop a simple developable surface modeling system.

Chapter 4 inspects and derives various geometric attributes on discrete orthogonal geodesic nets. These include DOG normals, a DOG Laplacian, and different measures of curvature and can be plugged in a solver to facilitate the modeling of DOGs.

Chapter 5 studies the shape space of DOGs for a given net connectivity. We see that DOGs are generally smoothly deformable, and show how this can be used to model deformations on DOGs.

Chapter 6 studies discrete isometries of planar surfaces, supplying two solutions for the problem of locking of planar isometries, based on discrete models for isometries of smooth orthogonal geodesic nets.

Chapter 7 presents a computational framework for interactive design and exploration of curved folded surfaces, based on discrete orthogonal geodesic nets.

Chapter 8 summarizes our contributions and discusses interesting avenues for future works.

1.5 Publications

The work on this thesis resulted in the following peer-reviewed publications:

[**Rabinovich et al. 2018a**] M. Rabinovich, T. Hoffmann, and O. Sorkine-Hornung. Discrete geodesic nets for modeling developable surfaces. *ACM Transactions on Graphics*, 37(2), 2018a

[**Rabinovich et al. 2018b**] M. Rabinovich, T. Hoffmann, and O. Sorkine-Hornung. The shape space of discrete orthogonal geodesic nets. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)*, 37(6), 2018b

[**Rabinovich et al. 2019**] M. Rabinovich, T. Hoffmann, and O. Sorkine-Hornung. Modeling curved folding with freeform deformations. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)*, 38(6), 2019

We also presented some of the material in this thesis in a course on nets in discrete differential geometry given in SGP 2019:

[**Rabinovich and Sorkine-Hornung 2019**] M. Rabinovich and O. Sorkine-Hornung. Nets in discrete differential geometry. Symposium on Geometry Processing Graduate School, 2019

During the course of this thesis, the following peer-reviewed papers were also published:

[**Rabinovich et al. 2017**] M. Rabinovich, R. Poranne, D. Panozzo, and O. Sorkine-Hornung. Scalable locally injective mappings. *ACM Transactions on Graphics*, 36(2): 16:1–16:16, Apr. 2017

CHAPTER 2

Modeling developables, the locking problem, and discrete nets

This chapter starts with a review on smooth isometries of planar surfaces. We give a mathematical intuitive explanation to the many ways one can bend a flat sheet. We then continue by introducing the locking problem; We will see how a fairly straightforward attempt to devise a model for isometries of planar surfaces results in a model that is quite limited.
Throughout the chapter we will tackle questions such as:

- What does it mean for a discrete model to lock?
- Why does it lock?

The goal of the rest of the chapter is to understand locking geometrically from the point of view of smooth and discrete nets. After a brief introduction to smooth nets, we continue with an introduction to the beautiful theory of discrete nets in discrete differential geometry [Bobenko and Suris 2008], from a modeling and optimization perspective. Discrete nets offer simple, concise, and optimization-friendly discrete models for smooth geometries. We thoroughly discuss a classical model discrete developable surface model, show why the locking shown in the beginning of the chapter occurs from a smooth point of view of differential geometry, and briefly introduce orthogonal geodesic nets as parameterizations capturing that does not lock.

2.1 Isometries on planar surfaces

In how many ways can one bend a planar surface?

A planar sheet can be smoothly and isometrically deformed into a variety of shapes. In such an isometric deformation, length of curves are preserved as well as angles between different curves. This definition however is not really constructive or intuitive; What kind of shapes can be created by these deformations?

This question can be answered by looking at the connection between bending and straight origami folds. As an example, consider the following cylindrical origami fold pattern and a smooth cylindrical shape.

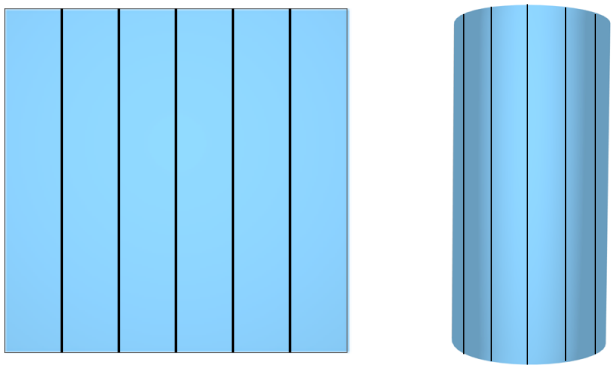


Figure 2.1: *A smooth cylinder can be thought of as the smooth limit of planar sheets folded along parallel lines. Left: A planar sheet and a family of straight lines to be folded. Right: A smooth cylinder, with a family of straight lines (also termed 'rulings').*

A cylinder is an example of a ruled surface, a surface swept by a moving straight line. It can be shown that a developable surface is locally planar, or is a ruled surface whose normals are constant along the rulings [Spivak 1999]. The latter are often called *torsal* surfaces. A torsal surface can be locally parameterized by

$$x(s,t) = \gamma(s) + t\,r(s),$$

where $r(s)$ corresponds to the direction vector of a ruling, and fixing the parameter t gives us another curve on the surface with non-vanishing curvature, $\gamma(s)$, from which the rulings emanate. Isometrically unrolling a developable surface onto the plane reveals the innate shape of its curves (see Fig. 2.2). In the following, we often

display a developable surface next to its flattened, isometric planar version, and refer to the geometry of a thereby flattened curve as the curve's intrinsic geometry. As an example, rulings or other geodesics on the surface are intrinsically straight. The intrinsic shape of a curve is determined by its geodesic curvature κ_g, which does not change under isometry.

Ruled developable surfaces are called torsal surfaces, and are typically classified based on the directions of their rulings: if they are parallel the surface is said to be cylindrical, if they intersect at a single point the surface is said to be conical, otherwise the rulings intersect along a curve and the surface is locally a tangent surface (see Fig. 2.3).

The possible presence of planar parts in developable surfaces further complicates their representation. A general developable surface is a composition of (possibly infinite) torsal and planar patches (Fig. 2.4). The connectivity between those patches is represented by a combinatorial structure termed the decomposition combinatorics [Tang et al. 2016].

If S is a smooth developable surface, then the set of points on it can be decomposed into flat points (having zero mean curvature $H = 0$) denoted A and non-flat points denoted $U = S - A$. By definition it is clear that A is a closed set while U is an open set. U is a union of ruled surfaces. By [Massey 1962] the rulings are "maximal" in the sense that they extend to infinity or hit the boundary, while the boundary points of U lie either on the boundary of the entire planar surface S, or on rulings which are limits of nearby rulings (see Fig. 2.5).

While bending a developable surface, planar parts can become bended and bended parts can be flattened, thus the decomposition combinatorics can also change. Even for a fixed decomposition combinatorics, bending typically smoothly changes the location of the patches as well as the rulings in each patch.

Given a planar surface S_0, smooth bending can be described as a smooth function $f(t) = S(t)$ with $f(0) = S_0$ that parameterizes surfaces isometric to the reference S_0. One can retrieve the bending "initial" decomposition and ruling patterns on the flat domain as the limit of these patterns for $f(\epsilon)$ where $\epsilon \to 0$. Thus, a smooth isometry on a planar sheet can be described by:

1. A choice of an initial domain decomposition for torsal patches neighbouring planar parts.
2. A choice of a smooth non intersecting family of lines for each torsal patch.
3. Bending each torsal patch, while smoothly changing the rulings and the decomposition boundaries.

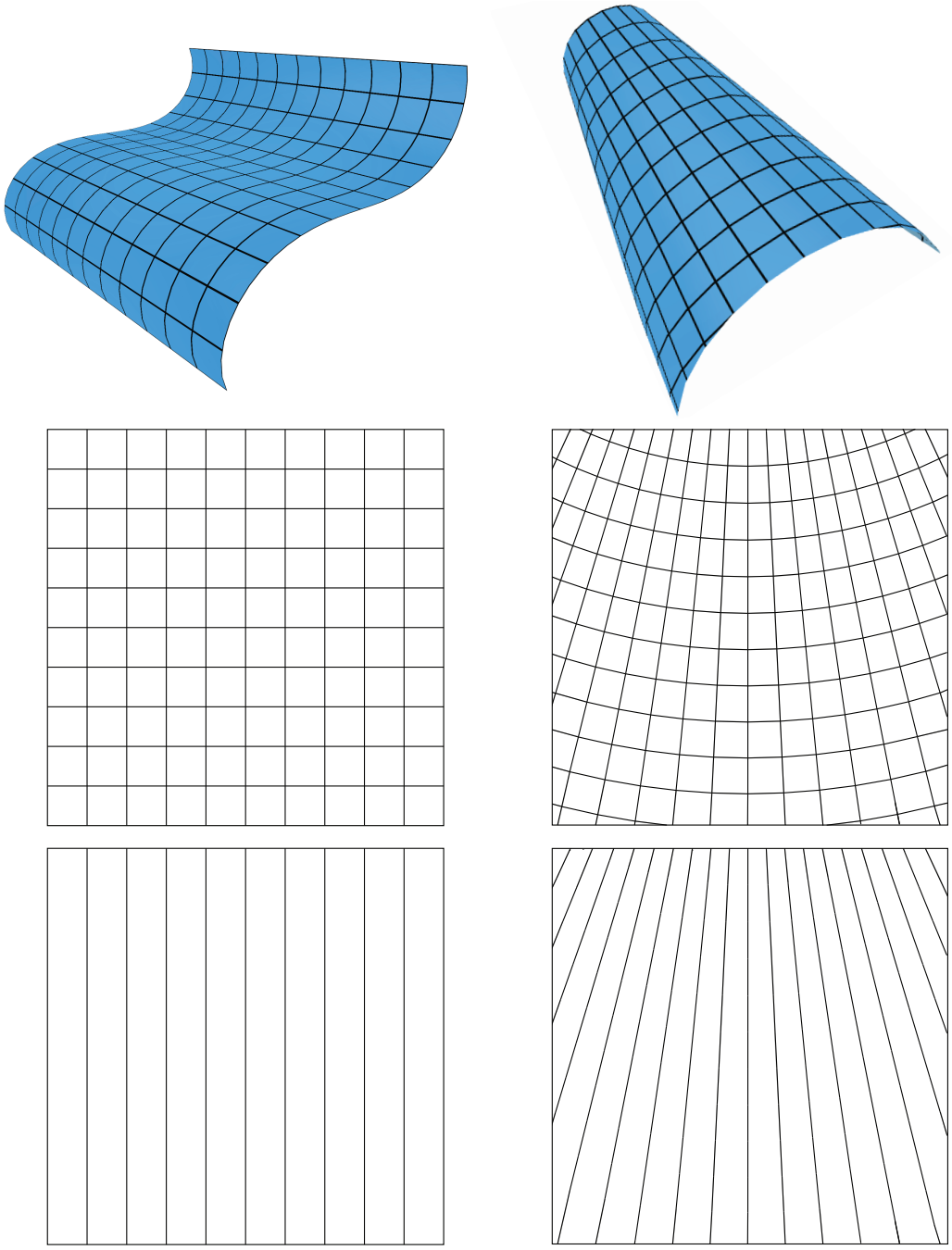


Figure 2.2: *Two isometric developable surfaces (a cylindrical and a conical one) in a ruled parameterization (top row), and their isometrically flattened versions (central row), which reveal the intrinsic geometry of the parameterization curves. The bottom row shows only the flattened rulings. Nearby rulings form a quad, which can be shown to be planar up to second order [Pottmann and Wallner 2001], making the resemblance to origami creases even clearer.*

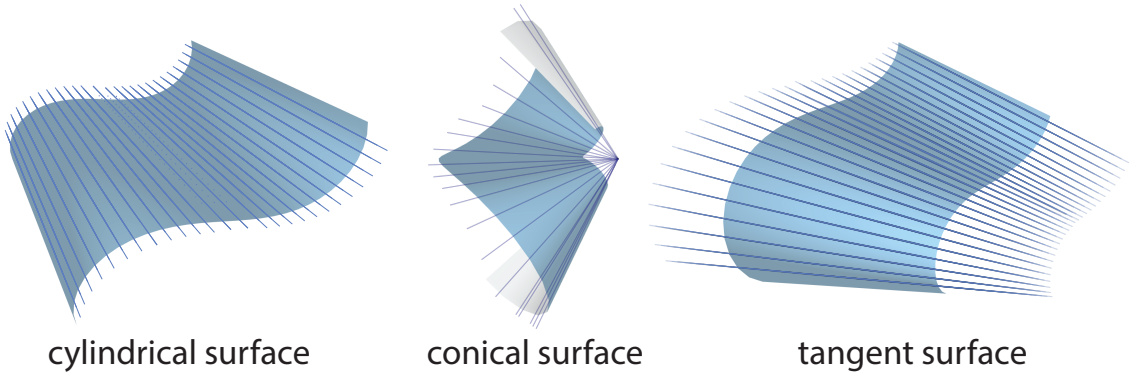


Figure 2.3: *Different types of developable surfaces and their rulings. Left: A cylindrical shape with parallel rulings. Center: all rulings meet in a point in a conical surface. Right: a tangent developable surface where the rulings meet at a curve.*

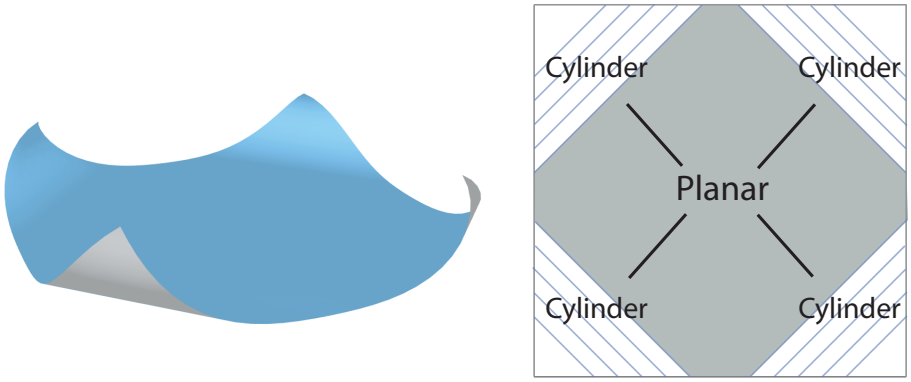


Figure 2.4: *A developable surface (left) and a flattened view of its decomposition graph and rulings (right). The surface is composed of a planar part at the center (colored in grey) and 4 cylindrical surfaces at its corners.*

2.2 Locking in discrete models of isometries on planar surfaces

The works of [Tang et al. 2016; Kilian et al. 2008; Liu et al. 2006] discretize torsal patches as planar quadrilateral strips, and developable surfaces as meshes decomposing into discrete torsal and planar patches, matching the characterization discussed above. While a very precise and elegant discretization, this has a notable drawback for modeling developable surfaces: it predetermines the decomposition combinatorics as well as the family of rulings on torsal patches (see Fig. 2.6).

One can also try to avoid this problem, and look for a more general notion of isometry on a planar meshing of some planar surface. There are many ways to mesh a planar sheet, and multiple ways to discretize isometric deformations by looking at discrete

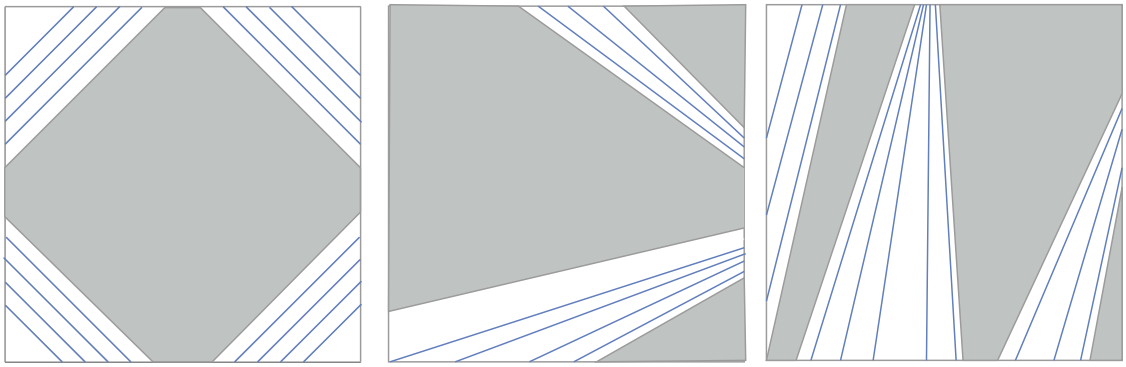


Figure 2.5: *Different decompositions of smooth developable surface into torsal patches and planar parts. Each torsal patch neighbours only planar patches and vice-versa. The boundaries of these patches are either rulings or the boundaries of the surface. Therefore, if the surface is intrinsically a polygon (as in these examples), then all of the patches are polygons as well.*

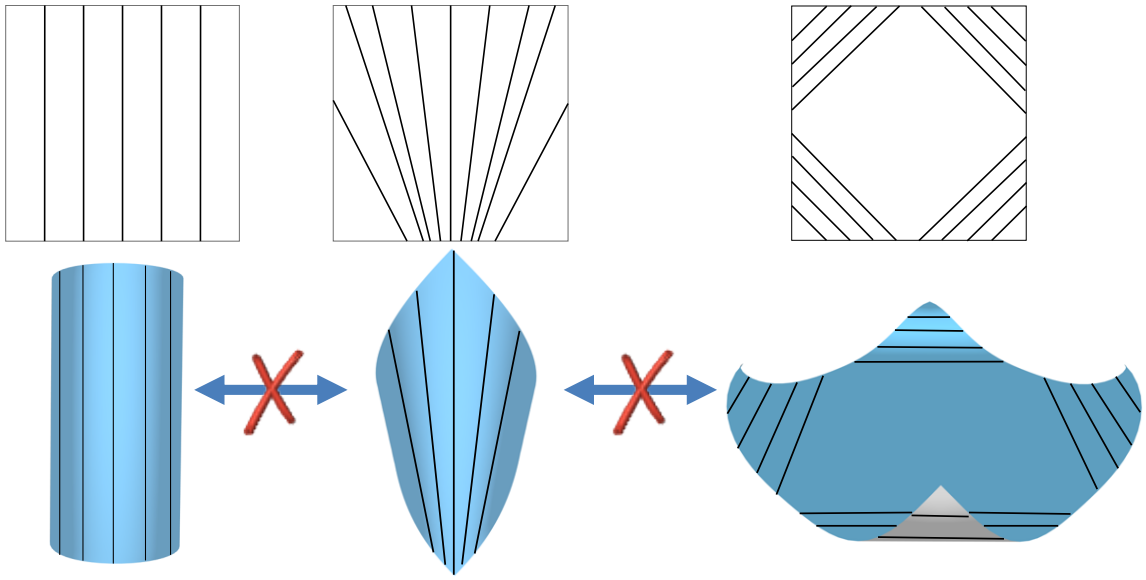


Figure 2.6: *Locking between developable surfaces in ruling based parameterization. If one starts with a planar mesh whose ruling lines are parallel, any small smooth deformation will create a mesh that is cylindrical or very close to being cylindrical, as the rulings smoothly deform as well. Thus, the meshing of the initial models induce locking starting from a planar path. Moreover, using this representation fixes the decomposition combinatorics a-priori.*

isometries of a mesh as deformations fixing edge lengths and angles. This however, results in a restricted locked model [Izmestiev 2016], as illustrated in Fig. 2.7.

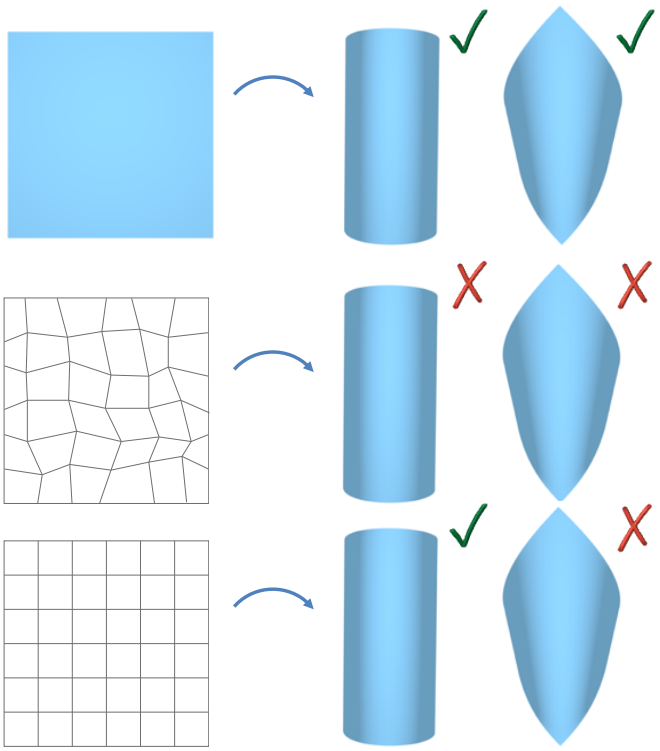


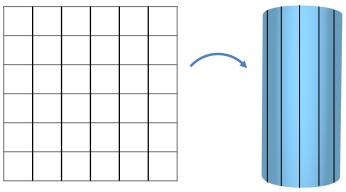
Figure 2.7: *A smooth planar surface (up) can be isometrically deformed to a variety of shapes, including cylinders and cones. A smooth isometry deforms the surface while not changing lengths of curves along the surface or angles between them. Modeling isometries by first meshing the planar surface and then considering deformations with fixed edge lengths and angles, leads to locking; An arbitrary quad mesh is entirely rigid (center), while a regular orthogonal grid mesh (down) can only deform to cylinders, but not to cones or other developables.*

Locking is a result of a choice of a mesh together with a set of constraints. In both cases the locking is independent of the resolution: further subdividing the orthogonal grid will still only allow for cylindrical bending, but the model cannot represent cones or most developable geometries. These discrete isometry models are *over constrained*.

2.2.1 Analyzing constraint models

The discrete isometry models mentioned above are an example of constrained models, which can be written as a set of n variables, $x \in \mathbb{R}^n$, satisfying a set of m constraints $\phi_i(x) = 0$. The variables in our case are the set of vertices coordinates in $\mathbb{R}^3$, $x \in \mathbb{R}^n$ with $n = 3\#V$ and $\#V$ as the number of mesh vertices, and a set of m constraints $\phi_i(x)$ with constraint fixing some length or angle in the mesh.

A smooth deformation on this discrete developable surfaces is a smooth function $F(t) = x(t)$ defined on the set of vertices such that each $x(t)$ satisfies the constraint. The arbitrary meshing isometry model above is *rigid*, i.e. can only be deformed by $F(t)$ that are global rigid motions, while the planar regular orthogonal mesh above can only be isometrically deformed into cylinders [Izmestiev 2016]. The latter is true as it can be shown that under any deformation that preserve all edge lengths and all angles on the regular grid mesh, one of the family of curves, the vertical or horizontal, has to remain straight. Thus, folding only occurs along the straight lines by preserving dihedral angles and the model is not completely rigid, but is obviously more rigid than it should be.



There are a variety of ways one can study constraint models and their set of solutions. An often used tool for that is the constant rank theorem, encountered both in geometry textbooks [Spivak 1999] and in the optimization literature [Nocedal and Wright 2006]. Intuitively, the constant rank theorem allows one to count the degrees of freedom of a constrained model as $k = n - m$, i.e. the difference of variables minus the number of constraints, under certain assumptions on the constraint set $\phi_i(x)$.

Theorem 1. *Constant rank theorem.* Let $x_0 \in \mathbb{R}^n$ satisfy $\phi_i(x_0) = 0, \forall i = 1..m$. If the functions $\phi_i(x)$ are smooth and if the gradients of the functions at x_0, $\frac{\partial \phi_i}{\partial x}(x_0)$ are linearly independent, then in a neighborhood of $x_0 \in \mathbb{R}^n$ the set of solutions to $\phi_i(x) = 0$ is a smooth manifold of dimension $k = n - m$.

In our setup, this means that locally one can smoothly deform the mesh while keeping all the constraints satisfied, and there are k smooth degrees of freedom while doing so. Such a deformation corresponds to a smooth choice of tangent vectors on the k dimensional tangent spaces on the smooth constrained manifold. In both examples listed above $m > n$, and it is therefore immediate that the gradients of the constraints are linearly dependent and the precondition for the constant rank theorem does not apply. Intuitively in such a case there is a redundancy in the set of constraints, just as in a linear system, and it is not clear how many solutions are there in that neighborhood. One might be able to formalize the same constrained model by using a smaller more compact set of constraints whose gradients are not linearly dependent, or alternatively use a completely different set of constraints to express the same model.

As we will later see, optimization problems based on discrete nets [Bobenko and Suris 2008] are well understood, and are defined by a set of minimal and simple geometric constraints.

2.2.2 Optimization of over constrained models

Interactive modeling of constrained meshes, for instance in an editing system, is typically achieved by solving optimization problems of the form

$$\begin{aligned} &\arg\min_x f(x) \\ &\text{subject to} \\ &\phi_i(x) = 0, \quad i = 1, \ldots, m, \end{aligned} \tag{2.1}$$

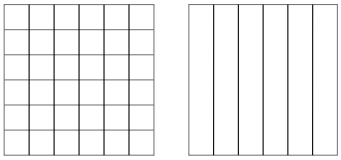
with an objective $f(x)$ defined to articulate some design task, for instance fitting positional constraints or minimizing bending. In many optimization routines, over-constrained problems can result in instability, especially when using second order constrained optimization methods such as SQP [Nocedal and Wright 2006]. A common practice in optimization is to remove linearly dependent constraints, for instance by performing a QR decomposition on their gradients. The latter can be more expensive than solving the system if there are many constraints [Rabinovich et al. 2018b]. Removing linear constraints is not necessarily enough, as it might be that the optimization problem has a very limited solution space, as in the above examples for discrete isometry models. In general it is often advised to model the problem, if possible, with a set of minimal linearly independent constraints [Nocedal and Wright 2006].

A common practice to handle such over constrained models in geometry processing is by enforcing the constraints in a least squared manner [Burgoon et al. 2006], effectively using a penalty based method. The constrained optimization problem (2.1) is then replaced with a non constrained problem:

$$\arg\min_x f(x) + \rho \sum \phi_i^2(x) \tag{2.2}$$

with a penalty parameter $\rho > 0$. This approach has several drawbacks; It is not clear how one should set $\rho > 0$. Setting it too low will result in a model that is too permissive, allowing modeling of non developable geometries such as a spherical or hyperbolic ones. Setting it to high still suffers from locking [Chapelle and Bathe 1998; Alessio 2012], as naturally the problem is over constrained as $\rho \to \infty$, while also resulting in a slow optimization since penalty based methods are ill-conditioned by nature [Nocedal and Wright 2006].

Understanding the geometry of the constrained model is quite beneficial, and can be used to better formulate an optimization task, or improve performances significantly. Another look into the discrete isometry model for the regular orthogonal mesh reveals that there are in fact redundant quads as well as redundant vertices;



Each mesh in such a model contains planar quads and a family of straight lines, implying that quads along the straight lines are coplanar and can be replaced by a single planar quad. The model is then equivalent to the same planar quadrilateral strip model for torsal patches we have seen at the beginning of this chapter. While equivalent geometrically, the planar quad strip models contains much less vertices, and a minimal set of linearly independent constraints that is better suited for second order optimization algorithms.

In the following we will see how the theory of smooth and discrete nets can be used to better understand constrained geometric models, and to devise good discrete models for various smooth geometries.

2.3 Nets in differential geometry

A parameterization, also termed a net, is a smooth map $f : \mathbb{R}^2 \to \mathbb{R}^3$ used to locally represent a smooth surface (Fig. 2.8).

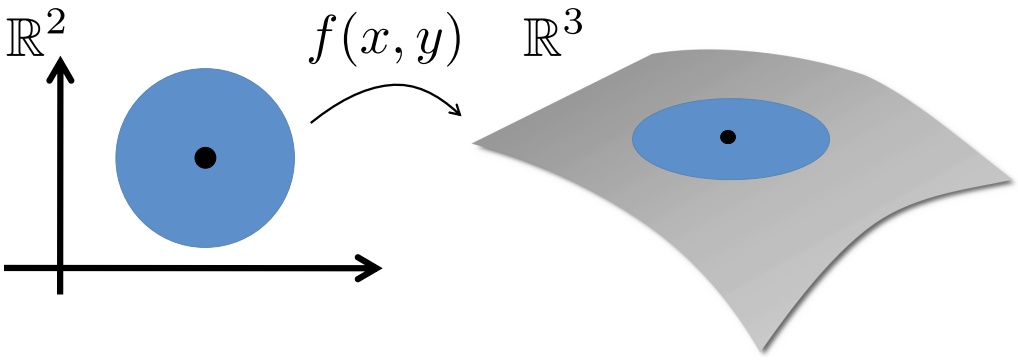
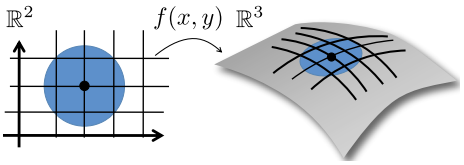


Figure 2.8: *A net is a smooth map from* $\mathbb{R}^2$ *to* $\mathbb{R}^3$*, here mapping the blue circle into an open neighborhood of a surface embedded in* $\mathbb{R}^3$*.*

Through a parameterization one can access a surface, "walk" on it, perform various measurements and study the surface. A parameterization is far from being unique. The same smooth surface can be represented by many different parameterizations, or nets. Many notions in geometry however are *parameterization independent*, for instance lengths and areas, different curvatures, and the Laplace-Beltrami operator. In practice, for the study of surfaces one either assumes nothing on the choice of parameterization, or chooses a parameterization that is most convenient for a given task.

These *convenient* parameterizations typically differ by the properties of their coordinate curves:

$$f(x0+t, y0), f(x0, y0+t)$$



Prominent examples of nets include:

- Curvature line nets (Fig. 2.9), where the coordinate curves are principal curvature lines.
- Asymptotic nets, whose coordinate curves trace the asymptotic directions of a surface.
- Geodesic nets, whose coordinate curves trace geodesics.
- Chebyshev nets, whose coordinate curves maintain the length of their preimage in $\mathbb{R}^2$.

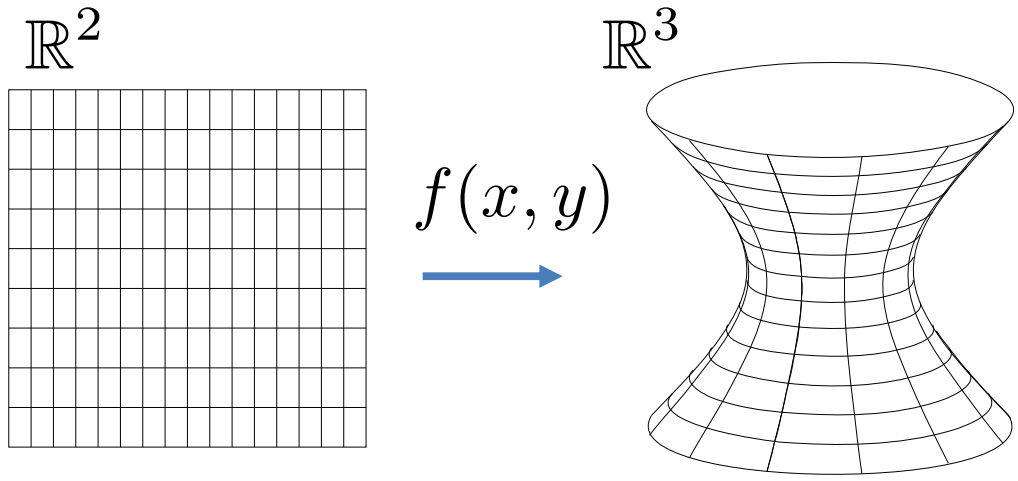


Figure 2.9: *Curvature line parameterization of a smooth minimal surface (right). Coordinate 'x' and 'y' lines on the left are mapped to curvature lines of a minimal surface on the right (in the figure, only a small finite set of lines are displayed).*

Every surface can be locally parameterized by curvature line nets, geodesic nets, or Chebyschev nets, but not every surface possesses asymptotic coordinates. Thus, certain nets can only be used to parameterize certain surfaces. Special classes of nets can often be used to charectarize various surfaces. For instance, if a surface can be parameterized by a net that is geodesic and curvature line net, then the surface is cylindrical. The latter is a small subset of developable surfaces. This kind of characterization is quite common, and it turns out that many classically studied surfaces can be characterized by special classes of nets. In particular: minimal

surfaces, pseudospherical surfaces, and developable surfaces, all typically defined by their curvatures, can be equivalently defined by the parameterization they admit.

Minimal surfaces have zero mean curvature ($H = 0$), and are surfaces that can be parameterized by a net of orthogonal asymptotic curves [do Carmo 1976].
Pseudospherical surfaces have constant negative Gaussian curvature (up to scale $K = -1$), and are surfaces that can be parameterized by an asymptotic Chebyshev net [Wunderlich 1951].
Developable surfaces have zero Gaussian curvature ($K = 0$), and are surfaces that can be locally parameterized by a conjugate ruled net [Sauer 1970; Pottmann and Wallner 2001], or by orthogonal geodesics [Rabinovich et al. 2018a].

2.4 Discrete nets in discrete differential geometry

While a geometer represents a smooth surface with a net, a computer scientist often use a mesh to represent a discrete surface. The two are intimately linked and in both cases one can represent the same geometry using different nets or meshes.

In geometry processing one often tries to design algorithms that are *mesh independent*, behaving similarly on different meshing of a similar surface. A prime example of the quest for mesh independence is the cotan Laplacian [Pinkall and Polthier 1993]. In fact, the most common sources of meshes in geometry processing are those created by scanning real-life objects or sampled from an analytical surface generated by a CAD software. Shape is king, while a mesh is often seen as a necessary evil, used to access an underlying smooth surface or to solve equations on it. The meshing used is just an arbitrary choice, just as a smooth map is. Thus *mesh independence* is a discrete analogue for *parameterization independence*.

Discrete differential geometry is the study of discrete counterparts of notions in differential geometry. Instead of smooth curves and surfaces, there are polygons, meshes, and simplicial complexes. A triumph of the field is a successful study of discrete analogues for a variety of surfaces, including constant Gaussian curvature surfaces [Wunderlich 1951; Bobenko and Pinkall 1996], minimal surfaces [Bobenko et al. 2006] and isothermic surfaces [Bobenko and Suris 2009]. As the freedom to choose various parameterizations for a given geometry exists also in the discrete setting, usually a discrete model of a surface in discrete differential geometry is coupled with a given net in mind. For example, discrete minimal surfaces have been defined through curvature line nets, and discrete constant negative Gaussian curvature surfaces through nets of asymptotic lines [Bobenko and Pinkall 1996; Bobenko et al. 2006].

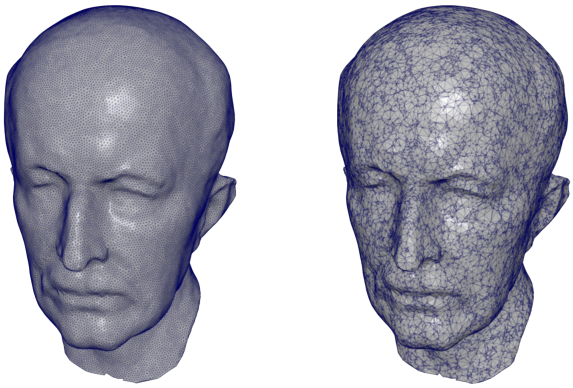


Figure 2.10: *Typical tools and algorithms in geometry processing strive to behave similarly on similar geometries regardless of the mesh used. This property is often termed 'meshing independence', and is analogue to parameterization independence in the smooth setup. Image taken from [Rabinovich et al. 2017].*

A *discrete net* (see Fig. 2.11) is a discrete analogue to a smooth net, defined as a quadrilateral mesh, often one that is mostly regular. Typically, quadrilaterals of

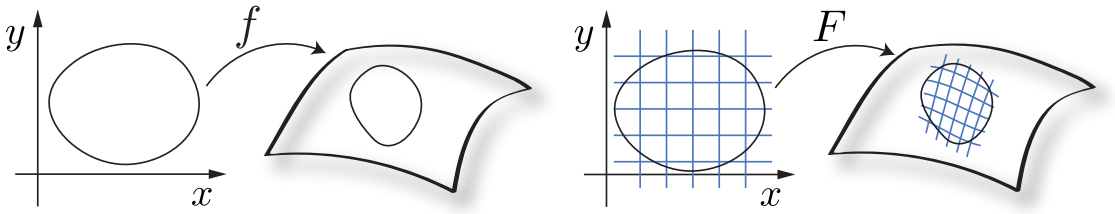


Figure 2.11: *A smooth net* $f : \mathbb{R}^2 \to \mathbb{R}^3$ *and a discrete net, here defined as a map* $F : \mathbb{Z}^2 \to \mathbb{R}^3$*, though a discrete net does not necessarily need to be a regular quad mesh.*

discrete nets are not planar. It is not clear how to "fill" the quads to a surface, how to render them, or which kind of smooth surface passes through the net exactly. From the point of view of discrete differential geometry, there is nothing but the net. The object of study is discrete by nature, without some hidden surface behind it. The study of nets, smooth or discrete, is not parameterization or mesh independent.

Just as in the classical theory of smooth nets, special nets in discrete differential geometry are classified based on the properties of their coordinate curves. Thus discrete curvature line nets, asymptotic nets, geodesic nets, Chebyshev nets and other sorts of nets are defined in terms of conditions that the mesh vertices satisfy, analogues to various differential equations satisfied by coordinate curves of smooth nets. The resulting conditions are unlike those derived by standard numerical discretization

techniques, and are designed to capture fundamental structures or invariants of the classical geometries. As a result, discrete nets are often simpler than their smooth counterparts and are often formalized based on geometric primitives such as lines, planes, circles, angles and edge lengths. This simplicity pays off, helping to shed light on some fundamental structures at the very basis of classical differential geometry and on the theory of integrable systems [Bobenko and Pinkall 1999; Bobenko and Suris 2008].

As an example, discrete circular nets are discrete analogues for smooth curvature line nets (see Fig. 2.12). Their definition, as nets whose quads can be inscribed in a circle, is undoubtedly simpler than their smooth counterparts. This point of view can also be flipped, and one can think of smooth curvature line nets as a special case of circular nets, as explained by Fig. 2.12. Further results regarding convergence and structure perseverance are detailed in [Bobenko and Suris 2008].

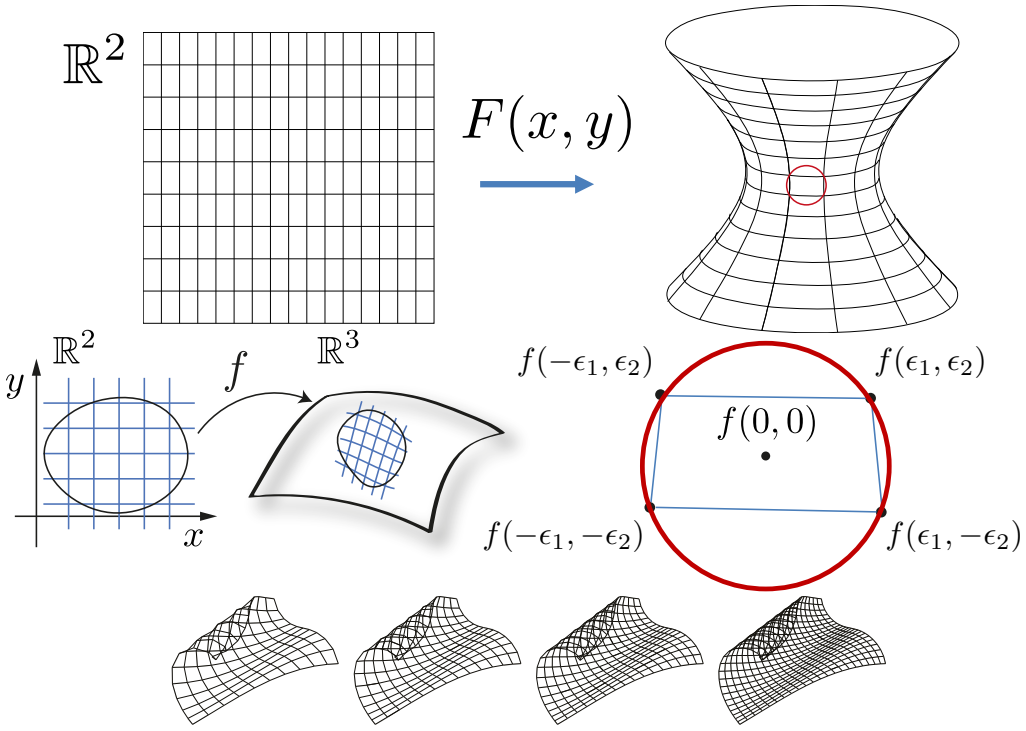


Figure 2.12: *Up - Circular nets are discrete analogues for curvature line nets. These are discrete nets with the property that each quad can be inscribed in a circle, implying that opposite angles of the quads sum up to π. Circular nets and smooth curvature line nets are invariant to Moebius transformations and are both nets where close-by normals are coplanar [Bobenko and Suris 2008]. Down - A smooth net is a curvature line net if and only if it is circular up to second order [Bobenko and Pinkall 1999]. Thus, a sampling of a smooth curvature line nets converges quickly into a discrete circular net.*

As noted, a good discrete model does not only converge to its smooth counterpart, but also shares a similar structure and behaviour with the smooth surface even when the discrete mesh is rather coarse. This structure can often be understood by an exact geometric construction of a discrete net with a minimal set of parameters.

Intuitively, in the case of circular nets, a geometric construction offers an exact answer to questions such as:

1. How many circular nets exist for a fixed quad mesh connectivity, and how can they be characterized?
2. How can one smoothly deform circular nets while keeping them circular?

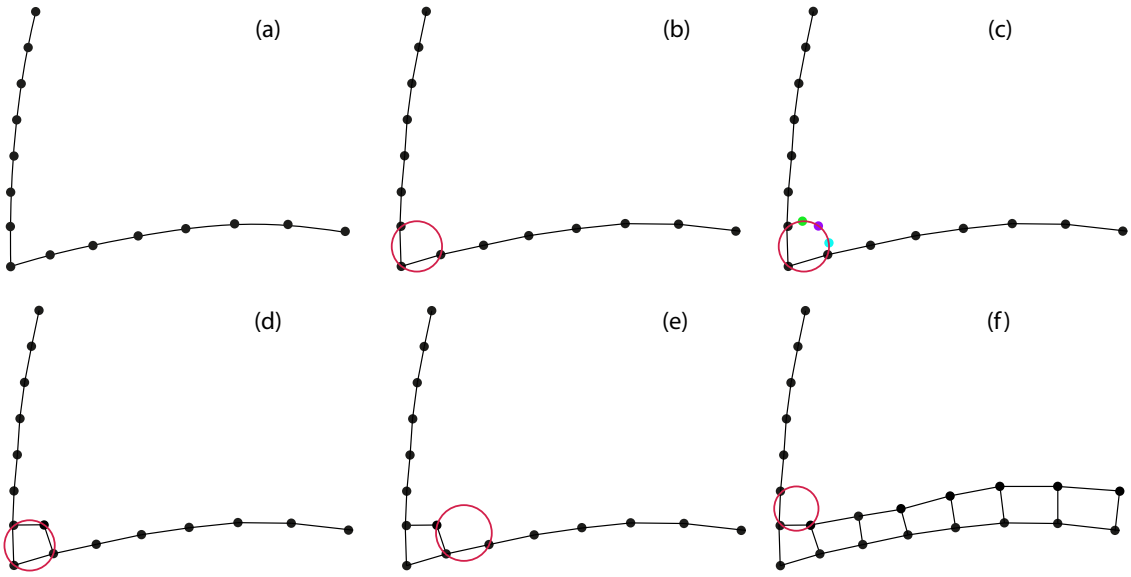


Figure 2.13: *Construction of a circular net from two discrete coordinate curves. This process is used to analyze the degrees of freedom in the model, showing how points can be added to satisfy the circular quads constraints. (a) - An input of two discrete coordinate curves. (b) - There is a unique circle passing through 3 points in general positions. (c) - Any choice of a point on that circle will generate a circular quad. Each such choice can be described by a smooth variable, for instance representing an angle. (d) - Setting the smooth variable results in a circular quad. (e-f) - The process of passing a circle through 3 points and choosing a smooth variable representing the point on the circle can be repeated for the entire row, and for subsequent rows generating a circular net.*

The results of this analysis, for circular nets and for various other discrete nets, are analogues to their smooth counterparts. Thus, showing that these models are not over-constrained, i.e. do not *lock*, but are also not too permissive (see Fig. 2.14).

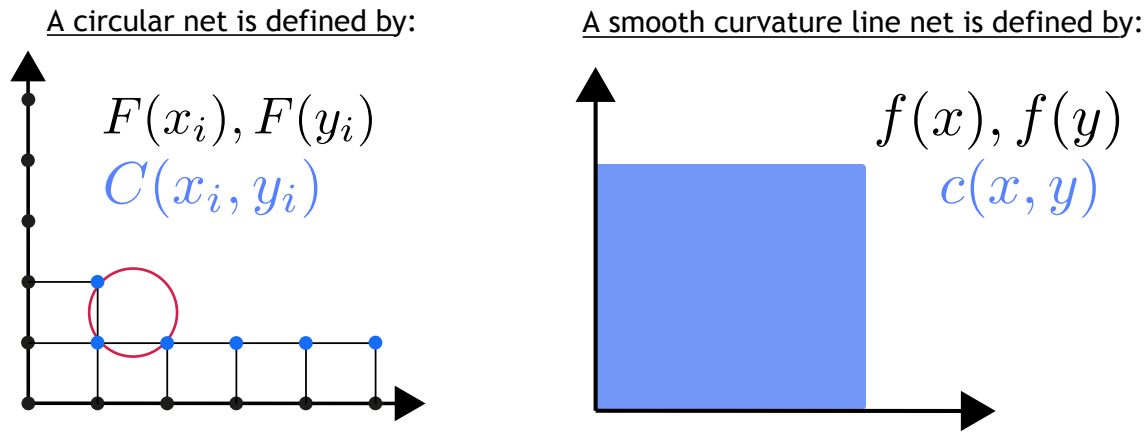


Figure 2.14: *The analysis of the geometric construction for circular nets shown at Fig. 2.13 provides a complete characterization for discrete circular nets through a finite set of smooth variables: The coordinates of two discrete curves $F(x_i), F(y_i)$, and a choice of circle for each quad $C(x_i, y_i)$. One can smoothly vary these parameters, generating a smooth deformation of discrete circular nets, as every choice of these parameters generates a different circular net. Conversely, every circular net can be uniquely mapped to these parameters. These degrees of freedom are analogue to the smooth case ([Bobenko and Suris 2008]), as it can be shown that every smooth curvature line net is defined by two smooth functions on the coordinate lines $f(x), f(y)$ (parameterizing two coordinate curves) and a smooth function on the blue domain $c(x, y)$ (analogue to the function $C(x_i, y_i)$ choosing the circles at Fig. 2.13). Thus, the analysis shows that the discretization is not prone to locking while also not being too permissive.*

From an applied geometry processing perspective, modeling geometries often leads to non-linear and non-convex optimization problems, generated by discretizing smooth non-linear equations. These result in two main problems. The optimization problems pose a numerical challenge, while it is also not clear if the model is prone to modeling related issue such as locking. Optimization problems for modeling discrete nets however have a well understood and well behaved solution space([Rabinovich et al. 2018b; Liu et al. 2006; Pottmann et al. 2015; Bobenko et al. 2010]). The geometric construction and the conclusions as displayed in Fig. 2.14 show that the set of circular meshes for a fixed combinatorics is a smooth manifold, and circular meshes are discrete objects that can be smoothly deformed. From a modeling perspective, this can help prove that there is no locking. From an optimization perspective, this generates a guarantee on the existence of nearby solutions if one starts in the feasible

set, as when modeling smooth flows/deformations of these nets. Moreoever, the set of constraints on discrete nets is often minimal, a property that is desirable by optimization methods such as SQP ([Nocedal and Wright 2006]).

2.5 Developable nets and their deformations

A parameterization of a developable surface through its rulings is called a developable *conjugate net* [Liu et al. 2006]. To clarify the previous statement, we elaborate on the definition of a conjugate net in a more general context, where f is a smooth net that is not necessarily developable. A smooth parameterization f is a conjugate net if

$$\langle n_x, f_y \rangle = 0, \quad \text{where } n = \frac{f_x \times f_y}{\|f_x \times f_y\|}.$$

Here, n is the normal map of f, and a subscript denotes differentiation with respect to the coordinate in the subscript. In this case the tangents f_x, f_y are said to be conjugate directions. The condition is equivalent to $f_{xy} \in \text{span}\{f_x, f_y\}$. Intuitively, in such a parameterization, infinitesimally small squares in the parameter domain are mapped to planar quads on the surface up to second order. Hence, planar quad meshes are seen as a discretization of conjugate nets [Bobenko and Suris 2008]. Note that curvature line nets are a special case of conjugate nets. In the case of a developable surface, the normal n is constant along a ruling, hence $n_x = 0$ and therefore any developable net parameterized through rulings is in fact a conjugate net. A well established *discrete* model for a conjugate developable net is a planar quad strip [Sauer 1970; Pottmann and Wallner 2001; Liu et al. 2006].

2.5.1 Locking explained by smooth flows on conjugate developable nets

Around non-planar points of a developable surface every direction has a unique conjugate direction. Thus the conjugate directions to a tangent of an arbitrary curve on the developable are always the rulings. Around planar points of developable surface however, every two directions are conjugate directions. Ruling on planar points are not defined, as there are many straight lines emanating from each point. The same planar surface can be parameterized by many different conjugate parameterizations $f_i^0(x, y)$, where different indices i represent a different net (see Fig. 2.15).

A smooth deformation on these surfaces can be written as a map from a parameter t to a net $f_i(t)$ with $f_i(0) = f_i^0(x, y)$. Planar ruled surfaces F_i^0 are discrete analogues of these parameterizations. At the beginning of this chapter (Sec. 2.2), we looked at smooth deformations $F_i(t)$ with $F_i(0) = F_i^0$ on these nets. Locking should not come

as a surprise, as it is just a consequence of the following fact: most isometries on planar surfaces do not keep all directions conjugate, and more clearly do not preserve an arbitrary given set of lines straight. The problem of over-constraining comes from the fact that the isometry model at Sec. 2.2 not only constrains the deformations to be isometries on the geometries parameterized by $f_i^0(t)$, but is not parameterization independent and discretizes only isometric deformations that keeps the coordinate curves of the arbitrary initial net conjugate, which in the discrete regime means keeping the quadrilaterals of $F_i^t()$ *planar* (see Fig. 2.15).

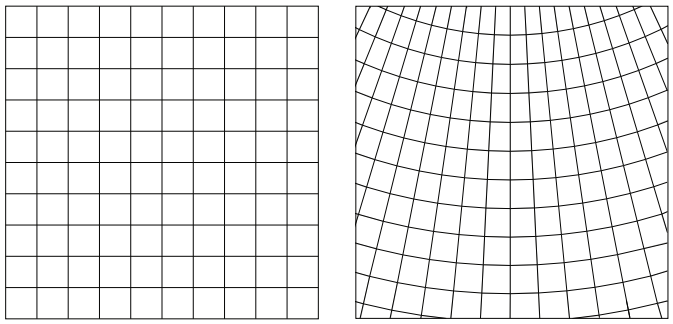


Figure 2.15: *There are infinitely many smooth conjugate nets parameterizing a planar mesh where one family of the coordinate curves are straight, with different set of straight lines. These correspond to different meshing of the planar strip where one of the family of coordinates are straight lines. The phenomenon of locking, or over-constraining, is just a result of discretizing smooth deformations that are not only isometries on the geometry, but also keep the coordinate curves of an arbitrary initial net conjugate. Thus, the discretization is over constrained, or locks, as most isometries on a planar surface do not preserve the conjugacy of an arbitrary family of curves.*

Planarity of the quadrilaterals might contribute to locking also when considering a larger set of deformations larger than isometries, for instance all deformations keeping a developable surface developable, by a similar argument.

2.5.2 Developable surface through orthogonal geodesic nets

We propose to look at a different type of parameterization of developable surfaces, one which does not limit the set of isometries in the smooth case, and is better suited for our interactive editing goals. Imagine taking a flat piece of paper with a square grid texture and watching the vertices of the squares while curving and rolling the paper. Squares, which started as planar, transform, but the intrinsic distances

between all points stay the same, as long as one does not tear or stretch the paper. This is analog to our model. We propose a discrete model of developable surfaces through intrinsic entities: geodesics, which are invariant under isometries. A net f is a geodesic net if its coordinate curves trace geodesics on the surface. On a developable surface, geodesics are straight lines when developed onto the plane. As we still have a degree of freedom in choosing the directions of the intrinsic lines, we set them to be of the simplest form – orthogonal – as in a rectangular grid (see Fig. 3.5). By employing geodesics rather than rulings and conjugate directions, we overcome the aforementioned locking problem and are able to define a notion of discrete isometry for such surfaces. This thesis deals with modeling developable surfaces through a discrete analogue to these smooth nets.

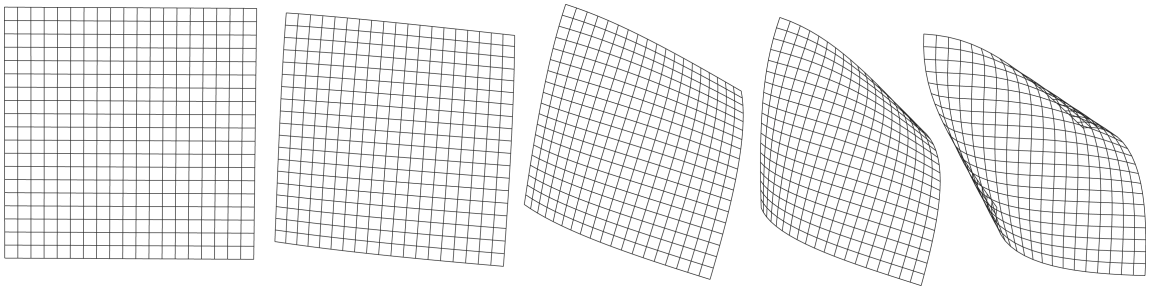


Figure 2.16: *Tracing orthogonal geodesics while rolling a planar surface into a circular cone.*

CHAPTER 3

Discrete orthogonal geodesic sets

This chapter lays the foundations for Discrete Orthogonal Geodesics (DOGs), a discrete model for developable surfaces parameterized through orthogonal geodesics. These are quadrilateral meshes satisfying simple angle constraints. The basis of the model is a lesser known characterization of developable surfaces as manifolds that can be parameterized through orthogonal geodesics. The model is simple, local, and, unlike previous works, it does not directly encode the surface rulings. This allows for modeling of continuous deformations of discrete developable surfaces independently of their decomposition into torsal and planar patches or the surface topology.

We start by deriving the model of DOGs, and then continue to prove and experimentally demonstrate strong ties between the discrete model to smooth developable surfaces, including a theorem stating that every sampling of the smooth counterpart satisfies our constraints up to second order. At the end of the chapter we use this model to build a simple editing system for developable surfaces with point handles as user interface. Our system can smoothly transition between a wide range of shapes while maintaining developability, and, unlike previous methods, does not require the user to specify global rulings or any other global structure of the unknown desired shape.

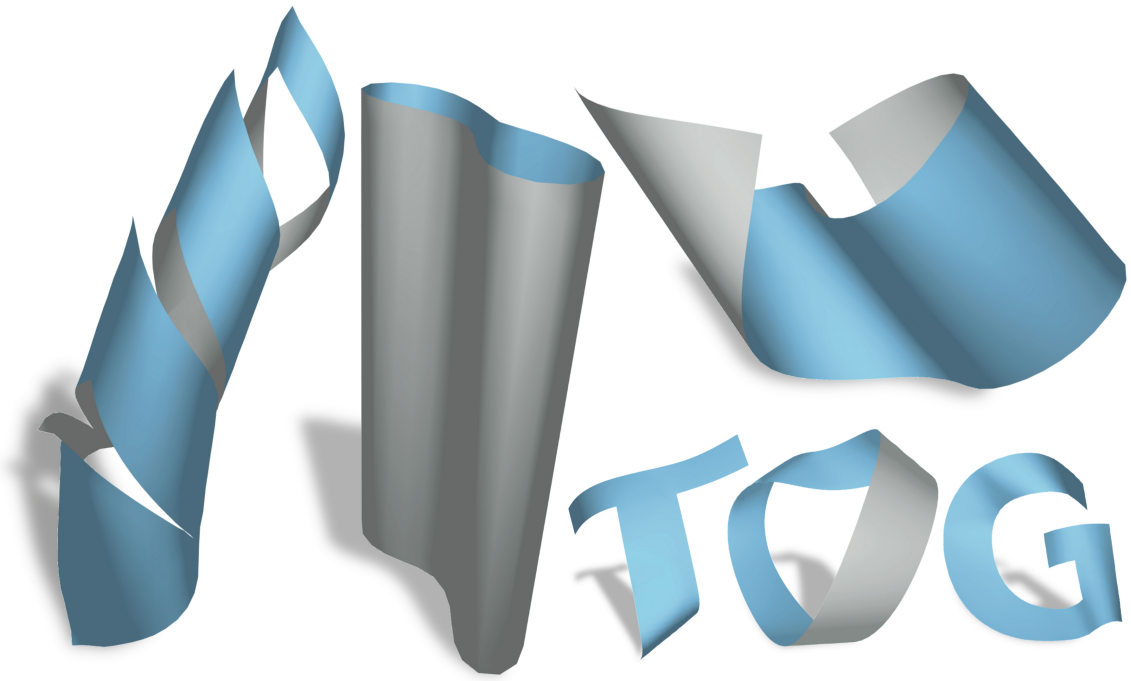


Figure 3.1: *We propose a discrete model for developable surfaces. The strength of our model is its locality, offering a simple and consistent way to realize deformations of various developable surfaces without being limited by the topology of the surface or its decomposition into torsal developable patches. Our editing system allows for operations such as freeform handle-based editing, cutting and gluing, modeling closed and un-oriented surfaces, and seamlessly transitioning between planar, cylindrical, conical and tangent developable patches, all in a unified manner.*

3.1 Related work

3.1.1 Developable surfaces

The theory of surfaces formed by local C^2 isometries of the plane is covered in the differential geometry literature [do Carmo 1976; Spivak 1999] and traces back to the works of Euler and Monge in the eighteenth century [Lawrence 2011]. Gauss' Theorema Egregium coupled with Minding's theorem shows that C^2 developable surfaces are surfaces with zero Gaussian curvature. Intuitively, this means that the image of their Gauss map is a curve or a point. Another point of view is the characterization of developable surfaces as special ruled surfaces, namely, those with constant tangents along rulings [Pottmann and Wallner 2001]. Hence, a developable surface is locally a planar or a torsal surface. A torsal surface can be constructed by a single curve: For example, one can pass a torsal surface through a curvature line curve and its parallel Bishop frame [Bishop 1975], or through a geodesic and its Frenet frame [Graustein 1917].

The study of C^1 and C^0 developable surfaces is a much newer area, stirred by the beautiful models and work of Huffmann [Huffman 1976; Wertheim 2004] and more recently by the field of computational origami [Demaine and O'Rourke 2007], which examines shapes created by straight and curved folds. Straight folds are C^0 creases through lines on a paper. Any shape created by repeated application of these folds is piecewise planar [Demaine et al. 2011a]. Curved folds are C^0 creases through arbitrary curves on a paper.

The study of smooth developable surfaces is analytic in nature, whereas the study of origami folds is in essence combinatorial. Our work focuses on modeling smooth deformations.

3.1.2 Modeling with developable surfaces

Though well understood mathematically [do Carmo 1976; Spivak 1999; Pottmann and Wallner 2001], computer aided modeling of developable surfaces has been proven to be a challenge and is an active research area. The primary difficulty lies in finding a discrete model that is able to deal with locking, as explained in Chapter 2. Works on deformations of developable surfaces can be largely categorized into *discrete developable models* and *discrete isometry models*.

Discrete models for developable surfaces are not tied to a fixed isometric reference. They mainly consider and discretize the notion of developability, or vanishing Gaussian curvature. The foundation of these works is a discrete developable surface model, i.e., an *exact* definition of the set of discrete developable surfaces. The definition should be flexible enough for the user to explore a wide range of shapes, while capturing important properties of the smooth surface. The works of Liu et al. [Liu et al. 2006] and Kilian et al. [Kilian et al. 2008] model torsal surfaces as planar quad strips, which are a discretization of developable conjugate nets. In [Tang et al. 2016] the authors model smooth torsal surfaces as developable splines. These are represented as ruled surfaces connecting two Bézier curves satisfying a set of quadratic equations that guarantee a constant normal along rulings. The work of [Bo and Wang 2007] models a torsal surface by a single geodesic curve and rulings emanating from it, i.e., the *rectifying developable* of a curve. The work of [Hwang and Yoon 2015] constructs developables by successive mappings to cones and cylinders. All works above model a general developable surface as a composition of multiple torsal surfaces, explicitly encoding rulings and sharing the locking problem we discussed in Chapter 2. We refer the reader to the 'Limitations' and 'Future work' paragraphs in Section 7 of [Tang et al. 2016] for an in-depth discussion of these shortcomings.

In contrast to geometric models, discrete isometry models are coupled with a fixed reference surface, often also modeling a material's behavior through energy minimization and simulating isometries of physical shapes when applying forces. These

include elastic simulation [Burgoon et al. 2006], or paper crumpling and tearing [Narain et al. 2013; Schreck et al. 2015, 2017]. Isometries of developable surfaces can be indirectly approximated by discrete shell models [Grinspun et al. 2003; Fröhlich and Botsch 2011] when starting from a flat sheet and setting a very high penalty in the stretch component of the elastic energy. Isometry based methods do not model intrinsic deformations by design, confining the user when designing a developable surface, as the geometry of the desired flattened shape is not necessarily known in advance. Current works on isometry models are also limited in terms of their extrinsic deformations, and also suffer from the locking problem ([Chapelle and Bathe 1998; Alessio 2012]), as also discussed in Sec. 2.1. In an attempt to accomodate that, these are often coupled with dynamic remeshing [Narain et al. 2012; Kilian et al. 2017; Narain et al. 2013; Schreck et al. 2015].

The work of [Solomon et al. 2012] presents an origami based editing system for developable surfaces, allowing the user to navigate through the highly nonlinear space of admissible folds of a sheet. By involving a mean curvature bending energy, the user can further ask to relax the folds, resulting in a smoother looking, yet always piecewise planar surface [Demaine et al. 2011a]. Due to the reliance on global folds, this method shares a similar dependency on rulings with the previously mentioned works, which also complicates the user interface.

3.1.3 Developable surfaces in discrete differential geometry

As mentioned in Sec. 2.5, the work of Liu et al. [Liu et al. 2006] discretizes developable surfaces through conjugate line nets as planar quad strips, where the transversal quad edges lie on rulings. In contrast, our proposed discretization is through orthogonal geodesic nets, which is especially convenient when modeling deformations and isometries of developable surfaces. Our discretization is inspired by the work of Wunderlich [Wunderlich 1951] on discrete Voss surfaces, which are surfaces parameterized through conjugate lines that are also geodesics. Voss surfaces include surfaces that are not necessarily developable, and modeling with conjugate *orthogonal* geodesics is quite limiting, since any such net is in fact a cylindrical shape. Therefore, as a base for our model we use the same notion of a geodesic net set by Wunderlich but drop the conjugacy requirement, which means that our model allows for non planar quads. This notion emphasizes geodesics as curves that are as straight as possible, similar to the work of [Polthier and Schmies 2006], which discretizes geodesics on polyhedral surfaces. The work of [Hoffmann et al. 2017] unites various discrete surface parameterizations by introducing edge-constraint nets, containing points and normals coupled by simple constraints. This results in a new discrete parameterized surface theory in $\mathbb{R}^3$, including a discrete definition for Gaussian curvature based on offset surfaces, where edge-constraint nets with vanishing Gaussian curvature are viewed as discrete developable surfaces. The authors note that the only examples of

such developable nets shown in that work are ruling based, besides a single Schwarz lantern.

3.2 Notations and setup

We denote continuous maps in lower case letters and their discrete equivalents by upper case. 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. Derivatives with respect to the coordinates x and y are denoted by subscripts, e.g., tangent vectors f_x, f_y and derivatives of the normal n_x, n_y. We denote the unit tangents of the coordinate lines by $t_1 = f_x/\|f_x\|$, $t_2 = f_y\|f_y\|$, which are linearly independent as f is an immersion.

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. Discrete unit tangents are denoted by T_1, T_2. We define these quantities in the following for our particular setting, namely discrete geodesic nets.

As is customary in discrete differential geometry, we slightly abuse the naming and employ shift notation to refer to vertex positions on our net, denoting

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

where $j, k \in \mathbb{Z}$, i.e., the lower index denotes the coordinate number to shift, and a bar above it indicates a negative shift (see Fig. 3.2). The unit-length directions of

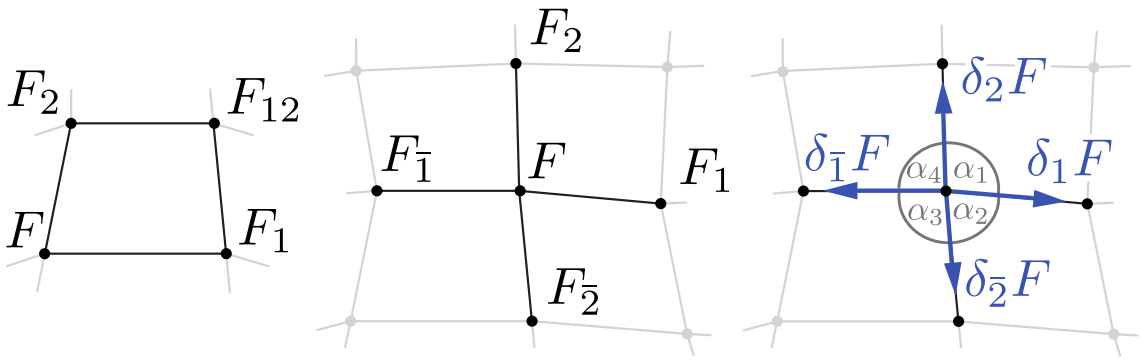


Figure 3.2: *The shift notation on a quad (left) and a star (center); edge directions and star angles (right).*

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_i, ordered consecutively (see Fig. 3.2). We assume our net is a discrete immersion, which means that the edge directions $\delta_i F, \delta_{\bar{i}} F$ are distinct. In practice, 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*. Our discrete nets neither require nor assume any global orientation on the mesh. The shift notation requires only a local arbitrary orientation per quad or star, and is used for convenience.

3.3 Discrete orthogonal geodesic nets

We are interested in defining conditions on F, i.e., on the positions of our mesh vertices, such that it represents a discrete developable surface parameterized by orthogonal geodesic lines. In the following, we develop the necessary definitions and their properties, to arrive at the following condition:

Definition 1. A discrete net F is said to be a discrete orthogonal geodesic net, if for every star all angles between consecutive edges are equal, i.e. $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$.

See Fig. 3.2 for notation. To develop the rationale for the condition above, we start by looking at smooth developable geodesic nets.

3.3.1 Smooth developable geodesic nets

When is a geodesic net a developable net? Let $f : \mathbb{R}^2 \to \mathbb{R}^3$ be a geodesic net and $P = \{(x, y) \in \mathbb{R}^2 \mid x_0 \le x \le x_1,\ y_0 \le y \le y_1\}$ an axis-aligned rectangle. The rectangle is mapped by f to a "curved rectangle" $f(P)$. Let $\theta_j,\ j = 1, \ldots, 4$, be the interior angles at the vertices of $f(P)$, measured as the angles between the respective tangent directions, e.g., $\theta_1 = \sphericalangle\,(f_x(x_0, y_0),\ f_y(x_0, y_0))$.

Lemma 2. A geodesic net f is developable if and only if for every axis-aligned rectangle $P \subset \mathbb{R}^2$, the angles of the mapped curved rectangle $f(P)$ satisfy:

$$\sum_{j=1}^{4} \theta_j = 2\pi. \tag{3.1}$$

Proof. Applying the local Gauss-Bonnet theorem to P (see [do Carmo 1976]), we get

$$\int_{f(P)} K\,dA + \int_{\partial f(P)} \kappa_g\,ds \;=\; \sum_{j=1}^{4} \theta_j - 2\pi, \tag{3.2}$$

where K is the Gauss curvature and κ_g is the geodesic curvature. Since f is a geodesic net, the images of P's edges under f are geodesics, and so $\kappa_g = 0$ on the curves of $\partial f(P)$, hence $\int_{\partial f(P)} \kappa_g\,ds = 0$.
[$\Rightarrow$] Assume $f(P)$ is developable. Then K vanishes and $\int_{f(P)} K\,dA = 0$, hence $\sum_{j=1}^{4} \theta_j = 2\pi$.
[$\Leftarrow$] Assume $f(P)$ is not developable. Then there exists a point $p = f(x_*, y_*)$ such that $K(p) \neq 0$; assume w.l.o.g. $K(p) > 0$. There is a sufficiently small neighborhood U with $(x_*, y_*) \in U$ such that $K > 0$ on $f(U)$. Let $P \subset U$ be an axis-aligned rectangle, then $\int_{f(P)} K\,dA > 0$ and from (3.2) we have $\sum_{j=1}^{4} \theta_j > 2\pi$, contradicting our condition (3.1). □

Corollary 3. An orthogonal geodesic net f, i.e., a geodesic net with $\sphericalangle(t_1, t_2) = \frac{\pi}{2}$, is a developable net.

An isometry f of a planar region $U \subseteq \mathbb{R}^2$ is an orthogonal geodesic net, as it maps a regular grid in the plane to orthogonal geodesics. Therefore the opposite is also true: every developable net can be parameterized by an orthogonal geodesic net. This is summarized by the following corollary:

Corollary 4. A smooth surface is developable if and only if it can be locally parameterized by orthogonal geodesics.

We are now ready to discuss discrete geodesic nets and our derivation of an equivalent condition for their orthogonality.

3.3.2 Discrete geodesic nets

As a base for our model we use the following definition:

Definition 2. A discrete net F is a *discrete geodesic net* if each two opposing angles made by the edges of a star in the net are equal (see Fig. 3.3).

This is a modification of a definition set by Wunderlich [Wunderlich 1951] in his work discretizing Voss surfaces, which are surfaces parameterized through conjugate geodesics. By [Wunderlich 1951], a discrete net F is a discrete Voss surface if it

is a planar quad net that also satisfies the angle condition in Def. 2. We remove the planarity restriction, as we are interested in discretizing geodesics that are not necessarily conjugate.

To obtain an intuition, consider the polylines $(F_{\bar{1}}, F, F_1)$ and $(F_{\bar{2}}, F, F_2)$ as two discrete coordinate curves passing through point F. A geodesic curve is "as straight as possible", dividing the angle deviation from π on both sides equally, i.e., $\alpha_1 + \alpha_2 = \alpha_3 + \alpha_4$ for the first curve and $\alpha_2 + \alpha_3 = \alpha_4 + \alpha_1$ for the second, where $\alpha_1, \ldots, \alpha_4$ are the angles around the star of F (see Fig. 3.3). Together, these two conditions are equivalent to $\alpha_1 = \alpha_3$ and $\alpha_2 = \alpha_4$, as in Def. 2.

We define tangents and normals on discrete geodesic nets through their (discrete) coordinate lines, mimicking the properties of their continuous counterparts. On a smooth geodesic net f, let $p = f(x_0, y_0)$ be some point and $\gamma_1(t) = f(x_0 + t, y_0)$, $\gamma_2(t) = f(x_0, y_0 + t)$ the coordinate lines through p. The curves γ_1 and γ_2 are geodesics emanating from p at two linearly independent directions $\gamma_1'(0) = f_x$, $\gamma_2'(0) = f_y$. If $\gamma_1(t)$ is regular and non-degenerate at 0, i.e., $\gamma_1'(0), \gamma_1''(0) \neq 0$, it has a well defined Frenet frame $\{t_1, n_1, b_1\}$ and an osculating plane Π_1 spanned by t_1, n_1. Since $\gamma_1(t)$ has zero geodesic curvature, its curvature is equal to the normal curvature of the surface, which implies that the curve's normal is in fact parallel to the surface normal at p: $n_1 \parallel n$ (where $n = \frac{t_1 \times t_2}{\|t_1 \times t_2\|}$). If also γ_2 has non-vanishing first and second derivatives, the surface normal n is parallel to the intersection line between the two osculating planes Π_1, Π_2. We can find a natural discrete model for those quantities for a discrete geodesic net F.

Let F be a vertex on a discrete geodesic net, and let Γ_1, Γ_2 be discrete geodesic curves through $F_{\bar{1}}, F, F_1$ and $F_{\bar{2}}, F, F_2$, respectively. We say that the curve Γ_j is non-degenerate if the three points $F_{\bar{j}}, F, F_j$ are not collinear. In that case, we can define the osculating plane and Frenet frame:

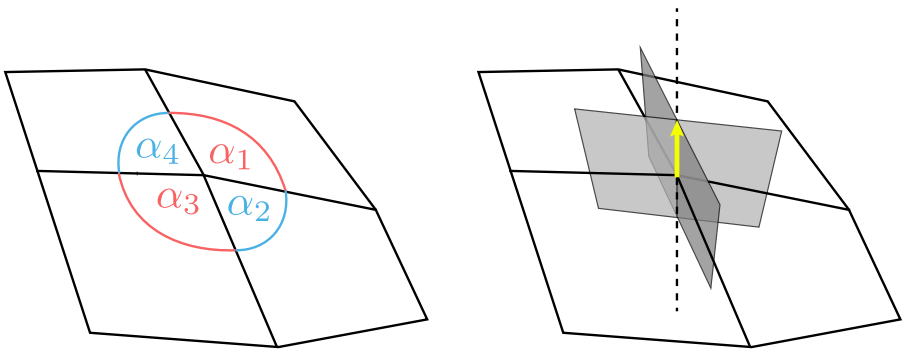


Figure 3.3: *Left: A star in a discrete geodesic net has equal opposing angles. Right: On a geodesic star, the intersection of the osculating planes of the discrete coordinate curves is the surface normal.*

Definition 3. The osculating plane Π_j, $j = 1, 2$, of a non-degenerate discrete curve Γ_j through vertices $F_{\bar{j}}, F, F_j$ is the plane passing through these three points. The Frenet frame of Γ_j at F is denoted by $\{T_j, N_j, B_j\}$, where

$$T_j = \frac{\delta_j F - \delta_{\bar{j}} F}{\|\delta_j F - \delta_{\bar{j}} F\|}, \quad N_j = -\frac{\delta_j F + \delta_{\bar{j}} F}{\|\delta_j F + \delta_{\bar{j}} F\|}, \quad B_j = T_j \times N_j.$$

See Fig. 3.4 for an illustration. Note that T_j are well defined also when $F_{\bar{j}}, F, F_j$ are collinear, and are never zero as our net is assumed to be a discrete immersion.

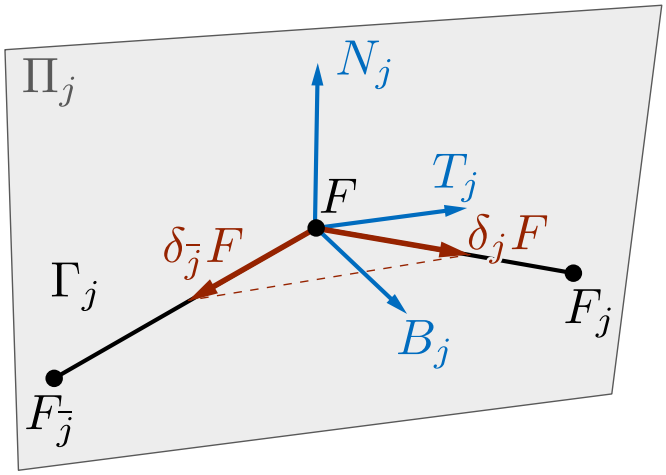


Figure 3.4: *A discrete coordinate curve Γ_j at F (in black), its osculating plane Π_j spanned by the edges of Γ_j, the Frenet frame (in blue): tangent T_j, normal N_j and binormal B_j. The dashed red vector is $\delta_j F - \delta_{\bar{j}} F$.*

Definition 4. The discrete Gauss map of a geodesic net F is

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

where T_1, T_2 are defined as above.

Just as in the continuous case, the principal normals of discrete geodesic curves and the surface normal agree, as shown by the following lemma:

Lemma 5. Let Γ_1, Γ_2 be two non-degenerate discrete curves around a vertex of a discrete geodesic net F and $\{T_1, N_1, B_1\}, \{T_2, N_2, B_2\}$ their discrete Frenet frames. Then N_1, N_2 and the discrete surface normal N (see Def. 1) are all parallel and lie on the intersection of the osculating planes Π_1 and Π_2.

Proof. By construction, $N_1 \perp T_1$, and by direct computation using the opposite angles condition (Def. 2) we have $\langle N_1, T_2 \rangle = 0$. Therefore $N_1 \parallel N$. Similar computation shows $N_2 \perp T_1$ and therefore $N_2 \parallel N$. □

Note that N is the angle bisector of both discrete curves meeting at F, see Fig. 3.3.

3.3.3 Discrete developable geodesic nets

Using the tangents defined above, we are now ready to define discrete developable surfaces through nets of orthogonal geodesics:

Definition 5. A discrete orthogonal geodesic net is a discrete geodesic net where at every star, the discrete tangents of the two discrete coordinate curves are orthogonal: $T_1 \perp T_2$. Such a net is a discrete developable surface in orthogonal geodesic parameterization.

This definition obviously reflects the smooth case, where an existence of an orthogonal geodesic net on a surface is equivalent to developablity (Cor. 4).

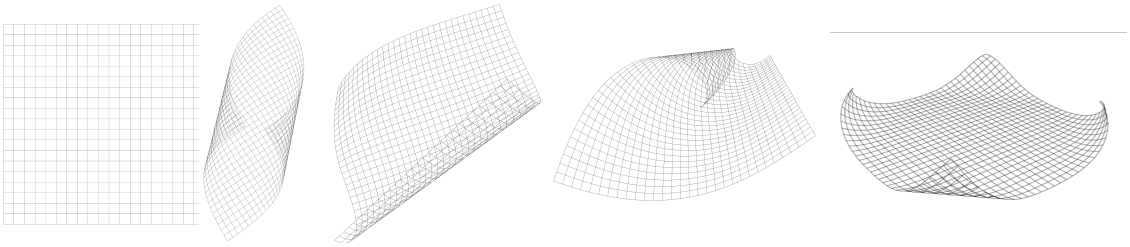


Figure 3.5: *Discrete orthogonal geodesic nets are regular quadrilateral meshes with equal angles around every vertex.*

The following theorem provides useful interpretations of our net and helps to see why this definition is equivalent to Def. 1 (see also Fig. 3.6).

Theorem 6. Assume a star has equal opposing angles, i.e., it fulfills the angles condition for discrete geodesic nets (Def. 2). Then the following conditions are equivalent:

1. The discrete tangents of the coordinate curves are orthogonal: $T_1 \perp T_2$.
2. The edges of the star form a right-angle cross when projected into the discrete tangent plane, which is the plane orthogonal to the discrete normal N.
3. All angles between consecutive edges of the star are equal.

Proof. By Lemma 5, N is a bisector of the unit-length vectors $\delta_1 F, \delta_{\bar{1}} F$, as well of the unit-length vectors $\delta_2 F, \delta_{\bar{2}} F$. Adding the formulas for T_1, T_2 from Def. 3 we have

$$\delta_1 F + \delta_{\bar{1}} F = \tilde{a}\, N, \qquad \delta_2 F + \delta_{\bar{2}} F = \tilde{c}\, N,$$
$$\delta_1 F - \delta_{\bar{1}} F = \tilde{b}\, T_1, \qquad \delta_2 F - \delta_{\bar{2}} F = \tilde{d}\, T_2,$$

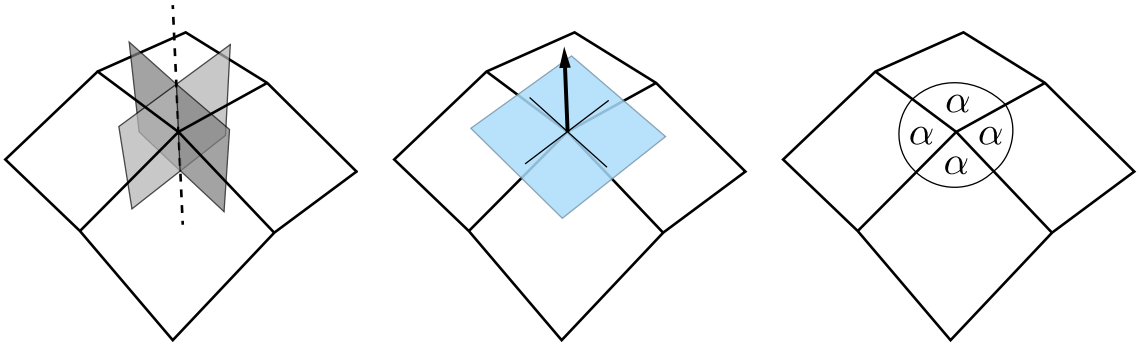


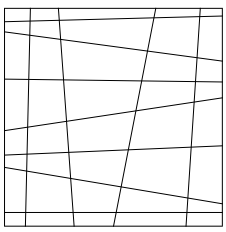
Figure 3.6: *These three conditions on a geodesic star are equivalent: the two osculating planes are perpendicular to each other (left), the projection of the star's edges onto the tangent plane forms an orthogonal cross (middle), all angles around the star are equal (right).*

for some $\tilde{a}, \tilde{b}, \tilde{c}, \tilde{d} \in \mathbb{R}$. By adding/subtracting the respective equations of the second row to/from the first row, we can write the star's edge directions as

$$\begin{aligned} \delta_1 F &= a\,N + b\,T_1, & \delta_2 F &= c\,N + d\,T_2, \\ \delta_{\bar{1}} F &= a\,N - b\,T_1, & \delta_{\bar{2}} F &= c\,N - d\,T_2, \end{aligned}$$

for some $a, b, c, d \in \mathbb{R}$.
$[\,(1) \Longleftrightarrow (2)\,]$ Projection to the tangent plane is equivalent to removing the normal component N from each vector, hence the direction vectors of the projected star edges are $bT_1, -bT_1, dT_2, -dT_2$ and the claim follows.
$[\,(3) \iff (1)\,]$ As we assume opposing angles in the star are equal, $(3) \iff \langle \delta_1 F,\ \delta_2 F \rangle = \langle \delta_{\bar{1}} F,\ \delta_2 F \rangle \iff \langle aN + bT_1,\ cN + dT_2 \rangle = \langle aN - bT_1,\ cN + dT_2 \rangle$, which is equivalent to $T_1 \perp T_2$ for a non-degenerate star. □

Note that the third condition (all angles in the star are equal) subsumes the condition for a discrete geodesic net (Def. 2) and conveniently encapsulates discrete orthogonal geodesic nets, as we expressed in Def. 1.



We can generalize Def. 5 to discrete geodesic developable nets that are not orthogonal, by considering discrete geodesic nets with an angle sum of 2π in every quadrilateral, based on Lemma 3.2.

This requires a useful definition for angles between discrete geodesic nets. We first note that measuring the angles between the osculating planes (or equivalently, the net tangents defined at this section), does not give a useful definition; Indeed given a discrete developable Voss surface, i.e. a discrete geodesic planar net, one can always isometrically fold the net along one family of the coordinate curves. Such a folding change the osculating plane angles, but as an isometry we expect it to preserve the discrete tangent angles,

and of course to remain discrete developable by our definition. Though discrete Voss surfaces are a small subset of developable geodesic nets, we would like to have them included in our geodesic developable nets definition. We therefore suggest another definition for angles on a geodesic net:

Definition 6. Let F be a vertex on a geodesic net, and let θF be the sum of all edge angles around F. The geodesic angle between the coordinate curve in the directions $F_{\bar{1}}, F, F_1$ and $F_{\bar{2}}, F, F_2$ at point F is given by:

$$\angle(F_x, F_y) = \frac{2\pi\angle(F_2, F, F_1)}{\theta F}$$

.

As expected, by switching the direction of one of the curves we change the measured angle to its complement, due to the geodesic star angles conditions. Measuring geodesic angles by scaling the edge angles, such that they will sum to 2π, was used for angles between geodesic curves on polyhedral surfaces in [Polthier and Schmies 2006]. By this definition, angles of planar Voss surfaces are not affected by isometric folding. This definition is also consistent with our definition of discrete orthogonal geodesic nets (Def. 5). We therefore use it for the more general definition of discrete developable geodesic nets:

Definition 7. A discrete developable geodesic net is a discrete geodesic net where at every quadrilateral the sum of angles between the quadrilateral curves, by definition Def. 6, is 2π.

3.4 Analysis and parallels with the smooth model

In this section we further study discrete orthogonal geodesic nets, drawing parallels between the discrete and continuous cases.

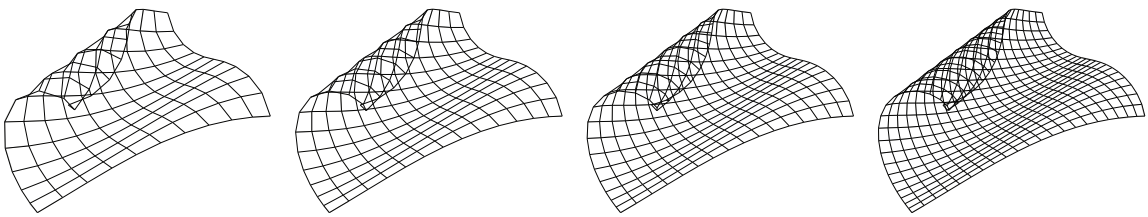


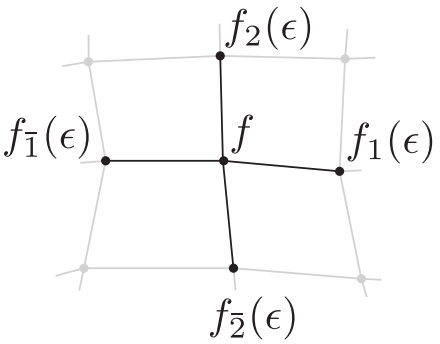
Figure 3.7: *A series of samplings of a smooth orthogonal geodesic net f with increasing sampling density. By Theorem 7, the stars of these discrete nets have equal angles up to second order, hence a discrete orthogonal geodesic net F can also be viewed as an approximate sampling of a smooth orthogonal geodesic net f.*

3.4.1 Approximation of an analytical, smooth orthogonal geodesic net

Let f be an arbitrary analytical smooth net and $p = f(x, y)$ a point on the surface. Imagine sampling points around p to generate a discrete star. We show that this star is a discrete orthogonal geodesic star as in Def. 1 up to second order if and only if f is an orthogonal geodesic net (Fig. 3.7).

Let $\epsilon > 0$ and let $f = f(x, y),\ f_1(\epsilon) = f(x + \epsilon, y),\ f_{\bar{1}}(\epsilon) = f(x - \epsilon, y),\ f_2(\epsilon) = f(x, y + \epsilon),\ f_{\bar{2}}(\epsilon) = f(x, y - \epsilon)$.

From here on, we refer to this set of points as an ϵ-*star* of the net f around the point p (see inset). The unit-length directions of the star edges are denoted as $\delta_j f(\epsilon),\ \delta_{\bar{j}} f(\epsilon)$.



By Def. 1, an ϵ-star is a discrete orthogonal geodesic star if all its angles are equal, i.e., if

$$\langle \delta_j f(\epsilon),\ \delta_{j+1} f(\epsilon) \rangle - \langle \delta_{j+1} f(\epsilon), \delta_{j+2} f(\epsilon) \rangle = 0, \tag{3.3}$$

where we use the notation $\delta_3 f(\epsilon) = \delta_{\bar{1}} f(\epsilon)$ and $\delta_4 f(\epsilon) = \delta_{\bar{2}} f(\epsilon)$ to enumerate all incident edges. We show that our discretization is indeed loyal to the smooth case in the following theorem.

Theorem 7. Equal angles on ϵ-stars.

1. An analytic net f is an orthogonal net, meaning $f_x \perp f_y$, if and only if all its ϵ-stars are discrete orthogonal geodesic stars up to first order, i.e., $\langle \delta_j f(\epsilon),\ \delta_{j+1} f(\epsilon) \rangle - \langle \delta_{j+1} f(\epsilon),\ \delta_{j+2} f(\epsilon) \rangle = o(\epsilon)$.
2. An analytic net f is an orthogonal geodesic net if and only if all its ϵ-stars are discrete orthogonal geodesic stars up to second order, i.e., $\langle \delta_j f(\epsilon),\ \delta_{j+1} f(\epsilon) \rangle - \langle \delta_{j+1} f(\epsilon),\ \delta_{j+2} f(\epsilon) \rangle = o(\epsilon^2)$.

Proof. Assuming f is analytic, with the shorthand $f_x = f_x(x, y), f_{xx} = f_{xx}(x, y)$, we use Taylor expansion to write the nearby points of f in the form

$$f_1(\epsilon) = f + \epsilon f_x + \frac{\epsilon^2}{2} f_{xx} + o(\epsilon^3), \qquad f_{\bar{1}}(\epsilon) = f - \epsilon f_x + \frac{\epsilon^2}{2} f_{xx} + o(\epsilon^3),$$
$$f_2(\epsilon) = f + \epsilon f_y + \frac{\epsilon^2}{2} f_{yy} + o(\epsilon^3), \qquad f_{\bar{2}}(\epsilon) = f - \epsilon f_y + \frac{\epsilon^2}{2} f_{yy} + o(\epsilon^3).$$

The rest of the proof requires writing the first coefficients of the Taylor expansion of the edge directions $\delta_j f(\epsilon), \delta_{\bar{j}} f(\epsilon)$. Here we derive the coefficients of $\delta_1 f(\epsilon)$, and the

other coefficients are analogous. The edge vector $f_1 - f$ can be written as

$$f_1(\epsilon) - f = \epsilon f_x + \frac{\epsilon^2}{2} f_{xx} + \dots$$

and so $\delta_1 f(\epsilon)$ can be written as

$$\delta_1 f(\epsilon) = \frac{\epsilon(f_x + \frac{\epsilon}{2} f_{xx} + \dots)}{\|\epsilon(f_x + \frac{\epsilon}{2} f_{xx} + \dots)\|} = \frac{f_x + \frac{\epsilon}{2} f_{xx} + \dots}{\|f_x + \frac{\epsilon}{2} f_{xx} + \dots\|}.$$

Let a_i be the Taylor coefficients of $\delta_1 f(\epsilon)$, by direct computation:

$$\begin{aligned} a_0 &= \delta_1 f(0) = \frac{f_x}{\|f_x\|} \\ a_1 &= \delta_1 f'(0) = -\frac{\langle f_{xx}, f_x \rangle}{2\langle f_x, f_x \rangle^{3/2}} f_x + \frac{1}{2\sqrt{\langle f_x, f_x \rangle}} f_{xx}. \end{aligned}$$

Similarly performing this for $\delta_{\bar{1}} f(\epsilon), \delta_2 f(\epsilon), \delta_{\bar{2}} f(\epsilon)$ and plugging the expressions in Eq. (3.3) gets us

$$\langle \delta_1 f(\epsilon), \delta_2 f(\epsilon) \rangle - \langle \delta_2 f(\epsilon), \delta_{\bar{1}} f(\epsilon) \rangle = \frac{2\langle f_x, f_y \rangle}{\|f_x\| \|f_y\|} + o(\epsilon)$$

and by symmetry we get exactly the same for the other angles. Therefore, the angles of an ϵ-star are equal up to first order if and only if f is an orthogonal (not necessarily geodesic) net. If f is orthogonal, then by plugging in $\langle f_x, f_y \rangle = 0$ we see that:

$$\begin{aligned} \langle \delta_1 f(\epsilon), \delta_2 f(\epsilon) \rangle &= \epsilon \frac{\langle f_x, f_{yy} \rangle + \langle f_{xx}, f_y \rangle}{2\|f_x\| \|f_y\|} + o(\epsilon^2) \\ \langle \delta_2 f(\epsilon), \delta_{\bar{1}} f(\epsilon) \rangle &= \epsilon \frac{-\langle f_x, f_{yy} \rangle + \langle f_{xx}, f_y \rangle}{2\|f_x\| \|f_y\|} + o(\epsilon^2) \\ \langle \delta_{\bar{1}} f(\epsilon), \delta_{\bar{2}} f(\epsilon) \rangle &= \epsilon \frac{-\langle f_x, f_{yy} \rangle - \langle f_{xx}, f_y \rangle}{2\|f_x\| \|f_y\|} + o(\epsilon^2) \\ \langle \delta_{\bar{2}} f(\epsilon), \delta_1 f(\epsilon) \rangle &= \epsilon \frac{\langle f_x, f_{yy} \rangle - \langle f_{xx}, f_y \rangle}{2\|f_x\| \|f_y\|} + o(\epsilon^2) \end{aligned}$$

Equality of all the linear terms implies $\langle f_x, f_{yy} \rangle = 0$ and $\langle f_y, f_{xx} \rangle = 0$. Together with $f_x \perp f_y$, this implies that f is a geodesic orthogonal net. To see that, let n^x be the principle normal of the x coordinate curve and let $f_{xx} = a f_x + b n^x$ for some $a, b \in \mathbb{R}$. Then $0 = \langle f_{xx}, f_y \rangle = \langle a f_x + b n^x, f_y \rangle = \langle b n^x, f_y \rangle$ and so $n^x \perp f_y$. By construction, the principle normal satisfies $n^x \perp f_x$, which means that the principle normal of the x coordinate curve is parallel to the surface normal and so the curve is a geodesic. By a similar calculation, the principle normal of the y coordinate curve is parallel to the surface normal. □

3.4.2 Relation to conjugate and curvature line nets

Here we prove discrete versions of analogue to theorems in differential geometry, connecting conjugate nets, curvature line nets with geodesic nets and orthogonal geodesic nets in particular.

The following theorems show why one must drop conjugacy/planarity of quadrilaterals for modeling, in order to model shapes that are not cylindrical.

Theorem 8. A smooth orthogonal geodesic net f that is also a conjugate net has planar coordinate curves, and is generated by two families of parallel curves.

Proof. As f is orthogonal and conjugate, f is also a curvature line net, and since f is both a geodesic net and curvature line net, the coordinate lines are planar [do Carmo 1976]. In case the geometry parameterized by f is not planar, than it is cylindrical: The rulings are orthogonal to the fixed osculating plane of a curved coordinate curve, as they are orthogonal to its tangent as well as to its principle normal which is also a normal to the surface. □

Discrete conjugate nets are discrete nets with planar quadrilaterals.

We will show that conjugate DOG coordinate curves are planar, while each family of curves is parallel in the sense that tangents in a given lattice direction stays the same when moving on the other lattice direction. Let T^x, T^y, N be the DOG Frenet frame at a point F, with T^x be the tangent of the x direction and T^y of the orthogonal direction. We will use the index notation on these vectors, i.e. denoting T^x_1 as the x tangent at F_1.

Theorem 9. If F is a DOG and a discrete conjugate net, i.e. with planar quadrilaterals, then $T^y \parallel T^y_1$ (likewise $T^x \parallel T^x_2$), and the coordinate curves are planar.

Proof. Assume F is a DOG with planar quadrilaterals, then the osculating plane at F at the x direction is the unique plane containing both $\|F_1 - F\|$ and also bisecting the dihedral folding angle between the planar neighbouring quads to $\|F_1 - F\|$, and is therefore the osculating plane at F_1 at the x coordinate direction. As a result the x coordinate curve must have its discrete osculating plane as the unique plane orthogonal to the fixed y curve tangent along it, T^y, or otherwise be a straight line. Thus, coordinate curves of a conjugate DOG net form a family of planar parallel curves. □

The following theorem connects geodesic nets, not necessarily orthogonal, curvature line nets, orthogonal geodesic nets and cylinders.

Theorem 10. A smooth geodesic net f that is also a curvature line net is also an orthogonal geodesic net, and the net is formed by two families of planar parallel curves.

Proof. If f is a curvature line net then f_x and f_y are orthogonal, hence by Cor. 3 f is developable. By Theorem 8 f parameterizes a cylindrical shape. □

Conical meshes [Liu et al. 2006] are known to be a discrete analogue of curvature line nets. An inner vertex v is conical if all the four oriented face planes meeting at v are tangent to a common oriented cone of revolution, and a mesh is conical if its quads are planar and all of its inner vertices are conical.

Theorem 11. A discrete geodesic net F that is also a conical net is a discrete orthogonal geodesic net, and the net is formed by two families of planar parallel curves.

Proof. Using the notation of Fig. 3.3, a net is conical if and only if its quads are planar and every inner vertex satisfies the angle balance $\alpha_1 + \alpha_3 = \alpha_2 + \alpha_4$ [Wang et al. 2007]. Since the net is also a discrete geodesic net, $\alpha_1 = \alpha_3$ and $\alpha_2 = \alpha_4$ and therefore $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$, as in Def. 1. The rest follows from Theorem 9. □

3.5 Flattened isometric surface

Any smooth orthogonal geodesic net is a developable surface and can therefore be isometrically flattened to an orthogonal grid on the plane. If one considers isometry as exact preservation of edge lengths then this property doesn't hold *exactly* in most DOGs, but as we will see it does hold at the smooth limit.
A general DOG has non-planar quadrilaterals. Similarly to other nets in DDG, e.g., discrete K-surfaces, the geometric information of our net is only the vertex positions. Edges should not be seen as part of the surface, and the net quadrilaterals are generally non-planar. Note that this would also be the case for a dense sampling of a general smooth orthogonal geodesic net, which approximates our model, as discussed in Sec. 3.4.1.

Thus, while a DOG star can be isometrically flattened, a general DOG net cannot be (see Fig. 3.8); The reason is that opposite edges do not have exactly the same length, and therefore one cannot construct an orthogonal grid in the plane with these edge lengths. 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 $\|f_x\|_y = \|f_y\|_x = 0$. The following theorem shows that a smooth orthogonal geodesic net is also a Chebyshev net:

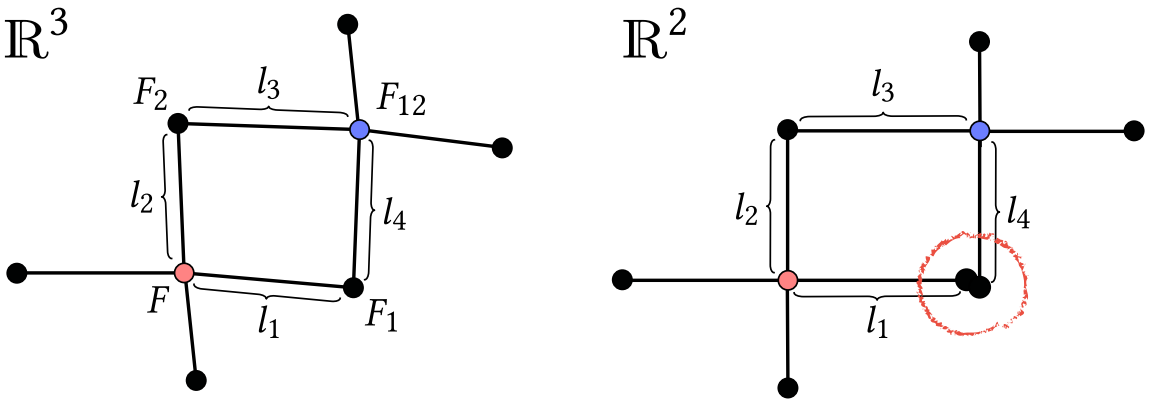


Figure 3.8: *Planar DOG nets are orthogonal grids, but for general DOG nets, one can locally flatten each star while exactly preserving the edge lengths, but generally not the entire net. Left: a DOG net in* $\mathbb{R}^3$ *containing two stars (red and blue). Right: individually flattened red and blue 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.*

Theorem 12. A smooth orthogonal geodesic net f is also a Chebyshev net, i.e., it satisfies $\|f_x\|_y = \|f_y\|_x = 0$.

Proof. 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 $\|f_x\|_y = \|f_y\|_x = 0$. □

Therefore a DOG net is approximately a discrete Chebyshev net as well as flattenable, and approaches being exactly one in the (smooth) limit. The following theorem hints at the limited expression power of DOGs that are also Chebyshev:

Theorem 13. A DOG that is also a discrete Chebyschev net has at most two different angles between edges along the entire net.

Proof. Quadrilaterals in a discrete Chebyschev net have opposite equal edges, which is equivalent to having equal opposite angles [Bobenko and Suris 2008]. The result then follows by propagation of both angle constraints: If the angle around a given DOG's star is α_1 and the angle around a neighbour star is α_2, then these are the only angles between edges in the net. See Fig. 3.9 □

We will see in Sec. 4.4 that these can only model a very partial set of of developable surfaces.

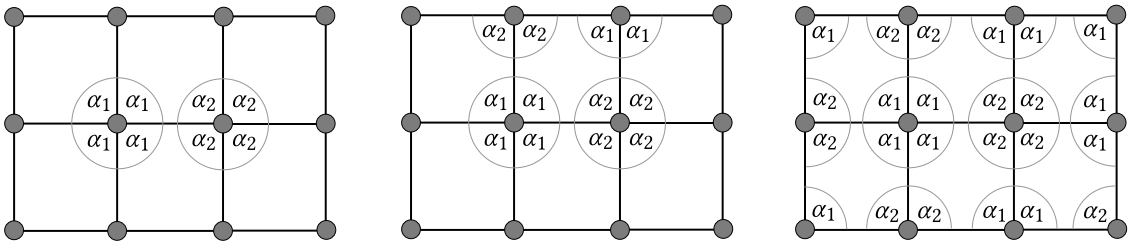


Figure 3.9: *Propagation of angle constraints of a Chebyshev DOG. Left: Two nearby stars in a Chebyshev DOG with equal angles around the vertices, α_1, α_2. Center: The angles of a nearby quadrilateral are determined as a Chebyshev quadrilateral have opposite equal angles. As the net is a DOG, other angles around the vertex are also equal. Right: The Chebyshev and DOG constraints propagate to the entire net. The net has at most two different angles between edges.*

3.6 Optimization and editing system

Our definition of discrete developable surfaces (Def. 1) is simple and local, such that it can be easily used in applications. We demonstrate this in an interactive editing system for discrete developable surfaces. Starting from a given discrete orthogonal geodesic net F^0, e.g., an orthogonal planar grid or a cylinder, the user can fix and move vertices around, as well as glue together or sever vertices. The latter is permitted only in case the operation keeps the mesh a (not necessarily oriented) manifold. We denote the set of vertices manipulated by the user (the handles) by $\mathcal{H}$. Whenever the user moves the handle vertices, the system computes a result from the space of discrete orthogonal geodesic nets, which is as close as possible to the prescribed handle positions. We analyze this shape space in Sec. 5.1. To choose a *good*, or intuitive solution, our optimization includes isometry and smoothness regularizers, as well as constraints for boundary vertices.

3.6.1 Orthogonal geodesic constraints

Def. 1 gives us the feasible shape space through a set of constraints on each inner vertex of F and a generalization for boundary vertices. We constrain every vertex to have all its corner angles equal. Let $e_j,\ j = 1, \ldots, l$, be the set of edges originating at a vertex v, ordered such that consecutive edges share a quad. Then the condition $\sphericalangle(e_j, e_{j+1}) = \sphericalangle(e_{j+1}, e_{j+2})$ is equivalent to:

$$\langle e_j, e_{j+1}\rangle \|e_{j+2}\| - \langle e_{j+1}, e_{j+2}\rangle \|e_j\| = 0. \tag{3.4}$$

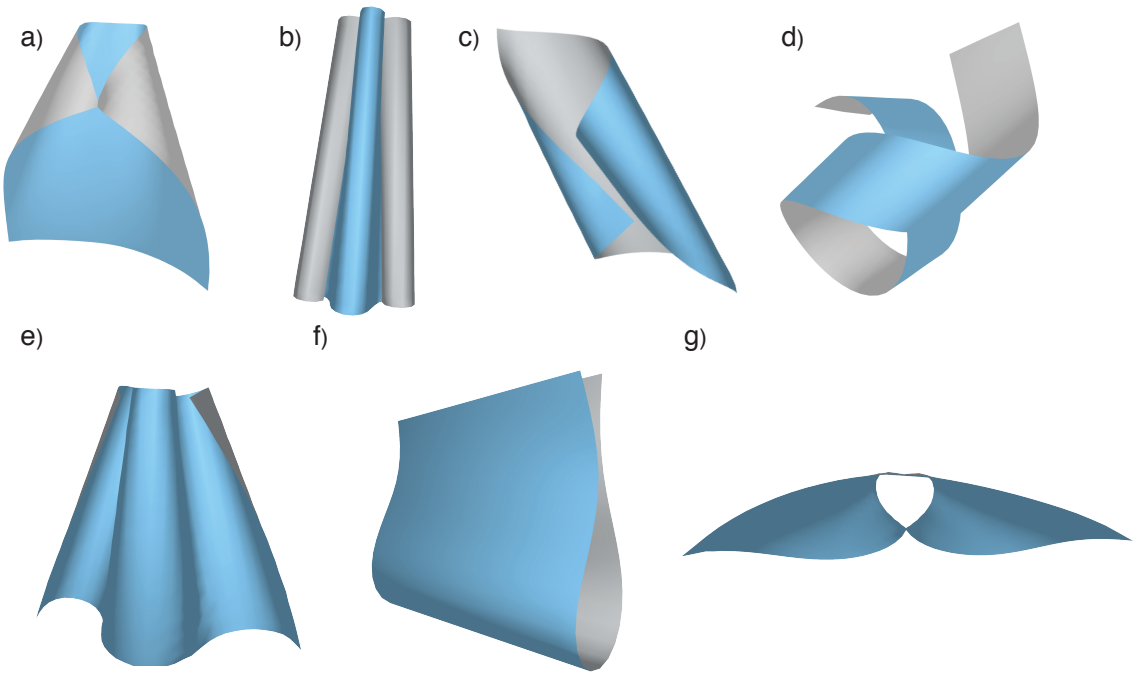


Figure 3.10: *Developable surfaces created using our vertex-handle based editing system. These examples were designed by deforming a flat sheet.*

In case of a corner boundary vertex with only two incident edges e_1 and e_2 and one angle, we constrain the angle to remain as in the reference shape:

$$\frac{\langle e_1, e_2 \rangle}{\|e_1\| \, \|e_2\|} - \arccos(\alpha) = 0, \tag{3.5}$$

where $\alpha = \sphericalangle(e_1, e_2)$ in F^0. We denote the constraints (3.4), (3.5) as $c_i(F) = 0$, $i = 1, \ldots, m$, where i enumerates all the inner and boundary vertices and their relevant incident edges.

3.6.2 Smoothness and isometry regularizers

The constraints above do not encode smoothness or isometry, and simply projecting a given initial guess onto the feasible space might lead to unintuitive results. To generate smooth and aesthetically pleasing deformations, we seek a feasible solution that minimizes a deformation energy $E(F)$. We employ a simple smoothness term, namely the Laplacian energy of the displacement w.r.t. the current state of the shape, or the current "frame", F^k:

$$E_{\text{smooth}}(F) = \left\| L(F) - L(F^k) \right\|^2, \tag{3.6}$$

where we use the simple uniform Laplacian L. The second energy term encourages maintaining isometry of the boundary, intuitively helping to control the scaling of

the deformation:

$$E_{\text{iso}}(F) = \sum_{e_j \in \partial F} (\|e_j\| - l_j)^2, \tag{3.7}$$

where ∂F is the set of boundary edges of F, and l_j's are the edge lengths in F_0. Finally, we add the positions of the handle vertices as soft constraints, since the user is likely to manipulate the handles in ways that are at odds with the developability constraints. The overall deformation energy is therefore

$$E(F) = E_{\text{smooth}} + w_{\text{iso}} E_{\text{iso}}(F) + w_{\text{pos}} \sum_{v \in \mathcal{H}} \|v - v_c\|^2, \tag{3.8}$$

where v_c are the handle positions prescribed by the user and w_{iso}, w_{pos} are scalar weights.

3.6.3 Optimization

In each frame, we solve the following optimization problem:

$$\begin{aligned} \arg\min_{F} \quad & E(F) \\ \text{subject to} \quad & c_i(F) = 0, \quad i = 1, \ldots, m. \end{aligned} \tag{3.9}$$

We use the quadratic penalty method [Nocedal and Wright 2006], which converts the above constrained minimization to a series of unconstrained problems of the form

$$\arg\min_{F} \quad w\, E(F) + \sum_i c_i(F)^2. \tag{3.10}$$

The above is iterated starting with $w = w_0$ and halving the weight w in each subsequent iteration, until the constraints are satisfied numerically, i.e. $\sum_i c_i(F)^2 < \epsilon$. The minimizations (3.10) are solved using using L-BFGS [Nocedal 1980], where we use ARAP [Sorkine and Alexa 2007] with the given positional constraints to get an initial guess. The figures in this section and the accompanying video to [Rabinovich et al. 2018a] were generated with the parameters $w_0 = 1,\ w_{\text{iso}} = 1,\ w_{\text{pos}} = 0.1,\ \epsilon = 1\text{e}{-12}$, and the input mesh was first scaled to have an average edge length of 1.

3.6.4 Results

We implemented our editing system on a 3.4 GHz Intel Core i7 machine, on which our single threaded implementation can handle around 1000 vertices interactively. The results in Fig. 3.10 demonstrate a variety of rolled, paper-like shapes similar to the results of [Solomon et al. 2012], but made with a more intuitive, vertex-handle based editing system (see also the accompanying video). Our system can seamlessly

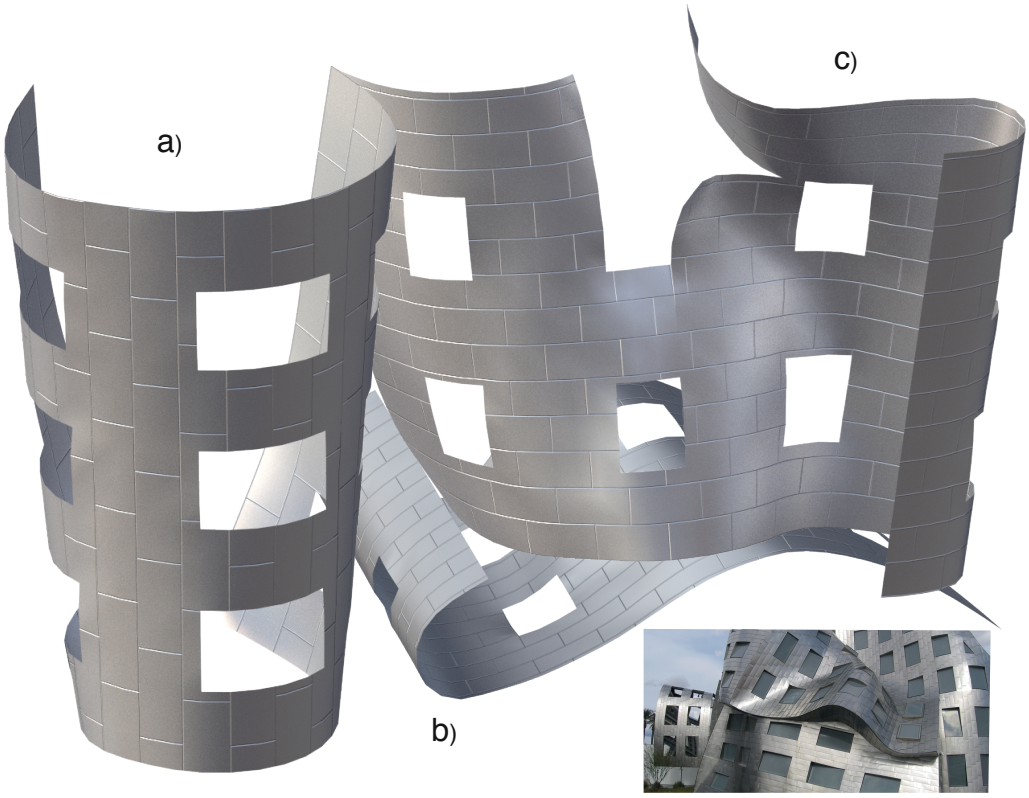


Figure 3.11: *An example of developable shapes with nontrivial topology, created in our interactive editing system (rendering done offline). Inspired by Frank Gehry's design of the Lou Ruvo Center for Brain Health (lower right). The texture coordinates for our model are simply obtained from the vertex coordinates of a planar rectangular grid with the same boundary edge lengths.*

handle surfaces with nontrivial topology, as well as non-orientable surfaces, as shown in Figs. 3.1, 3.12, 3.11.

The non-planarity of the quads in our model model implies that we can only render and fabricate our surfaces by arbitrarily triangulating them. Nevertheless, we demonstrate in Fig. 7.8 that our discrete model could be used for fabrication purposes. We supply further measurements supporting this claim in Table 3.1, computing approximated Gaussian curvatures for the models presented in this section using the finite difference based method of [Rusinkiewicz 2004].

3.7 Concluding remarks

This chapter focused on the geometric model of DOGs, its connections to the smooth case, and a straightforward integration of the model in an editing systems. Various

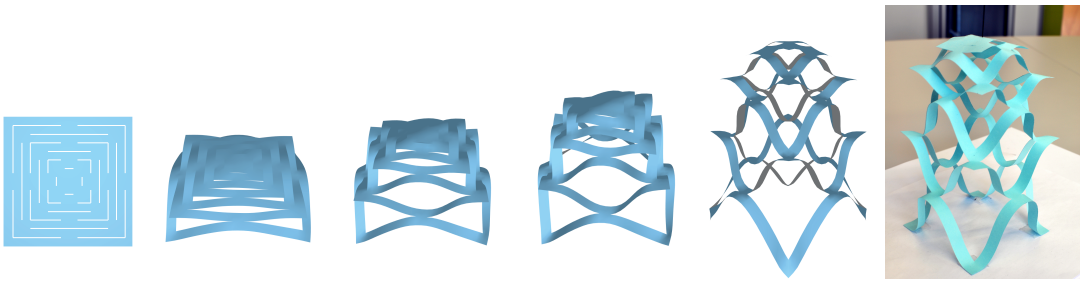


Figure 3.12: *A lantern shaped developable surface, demonstrating how our discrete model can seamlessly and effortlessly model developable surfaces with nontrivial topology. This figure was created from an alternating cut pattern on a square sheet (left). The shapes in the middle were formed by pulling the central vertex up while constraining the corners to stay on their initial plane. Right: A physical model made of paper with the same cut pattern (we glued the corners to the table and lifted the center point using a thin thread).*

Table 3.1: *Approximated Gaussian curvature*

Model	**Mean approx.** $\lvert K\rvert$	**Max approx.** $\lvert K\rvert$
Fig. 3.10-a	1.967×10^{-4}	0.0059
Fig. 3.10-b	5.922×10^{-4}	0.0169
Fig. 3.10-c	4.016×10^{-4}	0.0087
Fig. 3.10-d	2.041×10^{-4}	0.0026
Fig. 3.10-e	9.257×10^{-4}	0.0208
Fig. 3.10-f	2.648×10^{-5}	0.000043
Fig. 3.10-g	5.202×10^{-4}	0.0151
Fig. 3.11-a	3.659×10^{-4}	0.0067
Fig. 3.11-b	2.542×10^{-4}	0.0051
Fig. 3.11-c	2.543×10^{-4}	0.0051
Fig. 7.8	1.716×10^{-4}	0.0072

Gaussian curvature approximation using the finite difference based method of [Rusinkiewicz 2004] on our coarse models, with an average of less than 1000 vertices.

Figure 3.13: *Validation by fabrication. We 3D-printed a "sandwich" (top row) whose inner cut surface is the result of deforming a flat square using ARAP (left column) and our editing system for discrete developable surfaces (right column) using the same (soft) positional constraints. We cut two squares out of a thin copper sheet with the dimensions of the initial model before deformation, and we sandwiched these squares in the fabricated pieces (second row). Not surprisingly, since the ARAP result is not developable, the sheet wrinkles and buckles (bottom left), while our result exhibits pure bending (bottom right). Note that our result is smooth everywhere except at the boundary, where a cone-like kink is created in the digital model to remedy the doubly curved ARAP surface.*

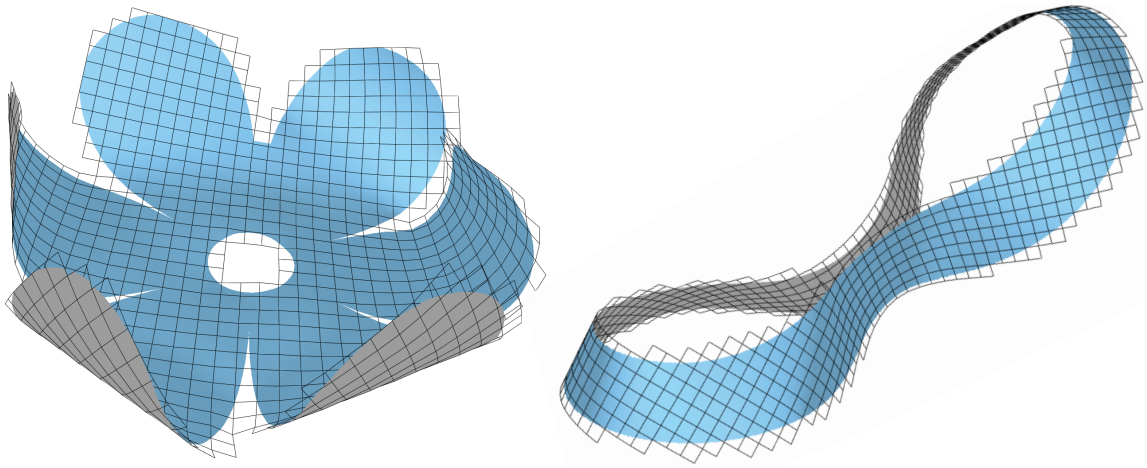


Figure 3.14: *Editing a flower and an 'O' shaped developable surface with curved boundaries. Such boundaries can be approximated up to any precision by an orthogonal geodesic net. In practice however, for the purpose of interactive editing, our grid resolution is limited by our L-BFGS optimization. Our current pragmatic solution to alleviate the jagged boundary appearance is culling using alpha-textures. In the future we plan to explore the discretization of general curved boundary conditions and prescribing geodesic curvature.*

practical as well as theoretical problems remain unanswered, some will be answered in the following chapters.

Deformation algorithms for DOGs. The deformation algorithm used in this chapter iterates between an initial guess, generated by ARAP, and a DOG constraints projection routine.
By using the DOG projection, we did not properly derived useful *optimization objectives* for various design tasks, and thus devised an editing system with limited modeling power. In Chapter 4 we will derive a variety of quantities such as mean curvature and a Laplacian, aiding modeling of DOGs.
Besides objectives, we also did not properly addressed the *DOG constraints*. Our DOG projection is rather slow; We used an out-of-the-box L-BFGS solver as a projection, which is limited to nets with ca. 1000 vertices, and scales badly. This limits the surfaces we are able to model (see Fig. 3.14). More importantly, we do not have any optimization guarantees, and it is somehow surprising that our optimization always seems to return meshes that numerically satisfy our many constraints. In Chapter 5 we will discuss this further, as well as supply an algorithm for modeling DOGs using smooth flows, generating guarantuees on the constraints satisfiability.

Isometry. While the editing system developed in this chapter consists of a slightly penalized isometry term in the objective, it is still unclear how should one model

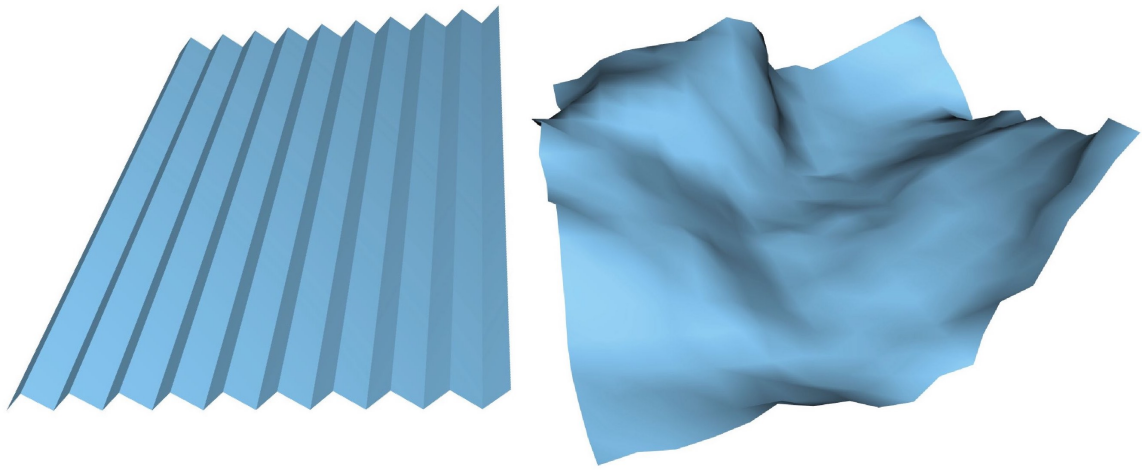


Figure 3.15: *Non-smooth discrete orthogonal geodesic nets. Left: A non-smooth cylindrical shape. Right: A creased surface, created after optimizing without* E_{smooth}*. We plan to examine these configurations in future work.*

exact isometries with DOGs, especially in light of the discussion in Sec. 3.5. This will be discussed in details in Chapter 6.

Origami folds and curved folds. Our model also supports non-smooth nets when optimizing without E_{smooth} (see Fig. 3.15), for instance non-smooth cylinders, as well as many other non-smooth configurations with creases. Though our discrete conditions are inspired by discrete quantities such as tangents and normals that are analogue to smooth ones, these do not encode smoothness by themselves. We will discuss folds and curved folds at Chapter 7.

CHAPTER 4

Geometric attributes of discrete orthogonal geodesic nets

In this chapters we exploit the highly regular nature of DOGs to derive a variety of discrete attributes on them, such as rulings, curvatures, and a Laplacian (see Fig. 4.1). These will be used to facilitate the design process of DOGs, as well as to better understand the net.

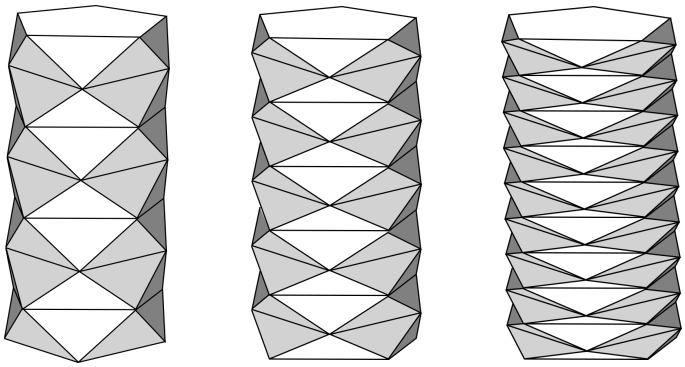


Figure 4.1: *Different sampling rates of the Schwarz Lantern, a classic example of pointwise convergence of the surface without convergence of its cotan Laplacian normals [Wardetzky 2007]. Our novel DOG Laplacian operator converges under sampling of any analytical orthogonal geodesic net, including a smooth Schwartz Lantern. For clarity, we display the triangular mesh, but in reality such a sampling consists solely of quadrilaterals.*

4.1 Notation

Throughout the chapter we use the shift notation used in Chapter 3, as well as denote the coordinate angles of a star and its edge lengths as displayed in Fig. 4.2.

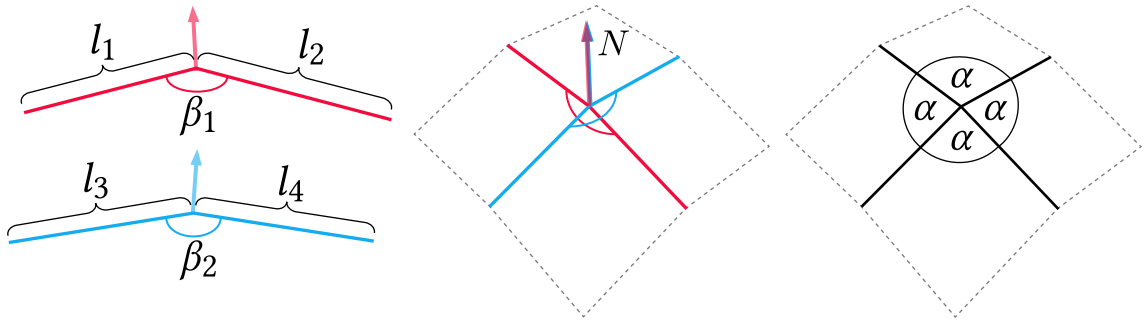


Figure 4.2: *Notation around a DOG star. The DOG's 4 edge lengths $l_j, j = 1..4$ the 2 curve coordinate angles β_1, β_2 (left), its opening angle α and normal N.*

4.2 Normals and rulings

We continue investigating our discrete developable surface model by looking at the Gauss map and a simple local definition of the rulings. Although our model does not explicitly enforce any properties of these two objects, we empirically see that their behavior corresponds well to the expected properties of a developable surface. This observation is further explained by a later convergence analysis on DOG normals in Theorem 17.

4.2.1 One-dimensional Gauss map

In the continuous case, a smooth developable surface f has vanishing Gaussian curvature. Since it corresponds to the area of the Gauss map, it means that the normal map n of f is one-dimensional [do Carmo 1976]. Sec. 3.3 supplies us with a discrete per-vertex normal map, N, on a discrete geodesic net F. N is defined in Sec. 3.3 as the intersection of the osculating planes of the discrete curves:

Definition 1. 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

$$N \parallel (\delta_1 F + \delta_{\bar{1}} F) \parallel (\delta_2 F + \delta_{\bar{2}} F), \tag{4.1}$$

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 $(F_{\bar{2}}, F, F_2)$.

We can view the collection of all vertex normals with the connectivity of F as a discrete net N. We show in Fig. 4.3 and the supplementary video of [Rabinovich et al. 2018a], that editing with our system results in a discrete Gauss map that is approximately one-dimensional.

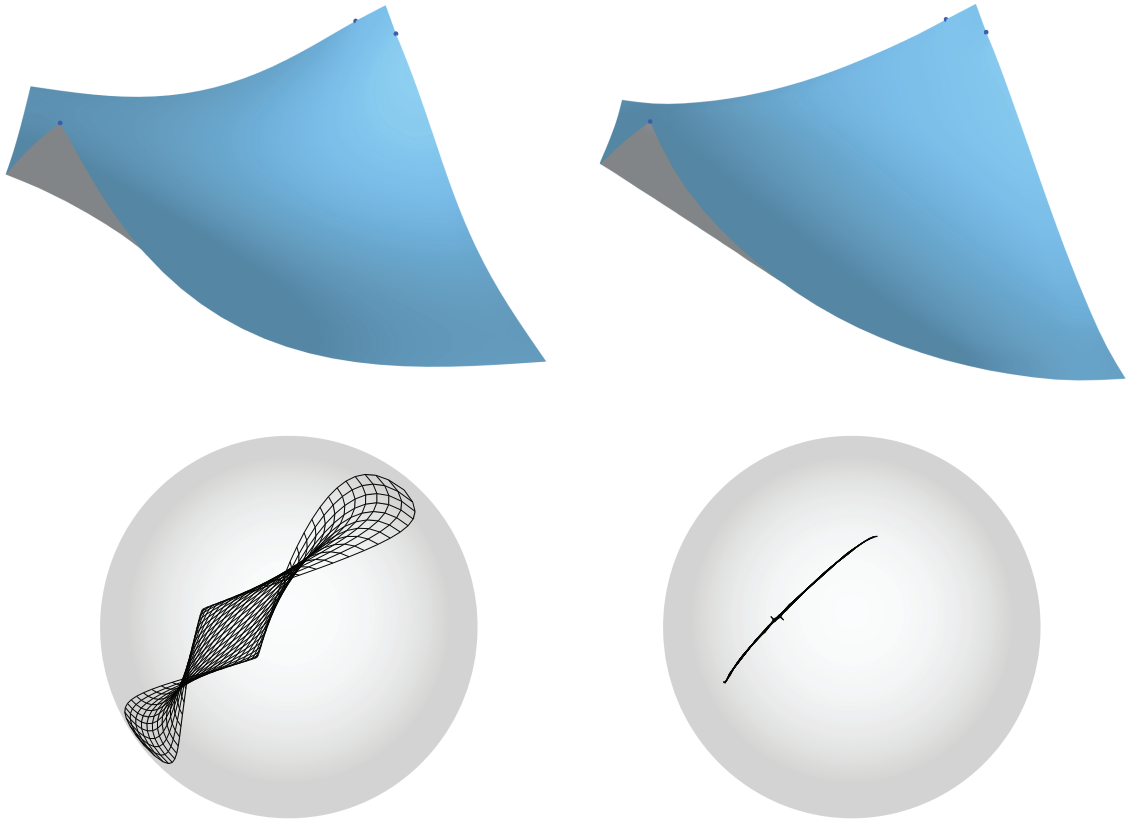


Figure 4.3: *Left: The result of moving the lower left corner of a planar mesh towards the upper right corner using ARAP deformation [Sorkine and Alexa 2007]. Right: the same positional constraints are employed in our developable surface deformation system (Sec. 3.6), using the result of ARAP on the left as the initial guess. In this case the soft positional constraints are satisfied up to high precision. Below each net we display the image of its Gauss map N (which in this case is virtually indistinguishable from the standard vertex-based normals of a triangle mesh obtained by triangulating the quad net). Note that our Gauss map tends to be one-dimensional.*

4.2.2 Vertex based rulings

Intuitively, rulings are line segments on a surface generated by the intersection of infinitesimally close tangent planes. As mentioned above, the Gauss map n of a

smooth developable net f has a one-dimensional image, or, equivalently, parallel partial derivatives: $n_x \parallel n_y$. There is a unique ruling emanating from every non-planar point on the surface in a direction r that is orthogonal to n. The ruling is a curvature line, hence it is also orthogonal to the other principal direction $n_x \parallel n_y$ [do Carmo 1976]. Therefore, if w.l.o.g. $\langle n_x, n_y \rangle \geq 0$, then $r \parallel n \times (n_x + n_y)$. This holds even if one of the terms n_x, n_y vanishes. This can be readily discretized:

Definition 8. The direction of a discrete ruling, emanating from a point F of a discrete geodesic developable net is

$$R = N \times (N_x + N_y),$$

where $N_x = N_1 - N_{\bar{1}}$ and $N_y = N_2 - N_{\bar{2}}$, oriented such that $\langle N_x, N_y \rangle \geq 0$.

Def. 8 is entirely local, however in practice the discrete rulings tend to fit the surface globally, see Fig. 4.4. Note that the definition above is only valid at inner vertices with all neighbors being inner vertices as well, such that N_x, N_y are defined. Unlike in the continuous case, N_x and N_y are not necessarily exactly parallel.

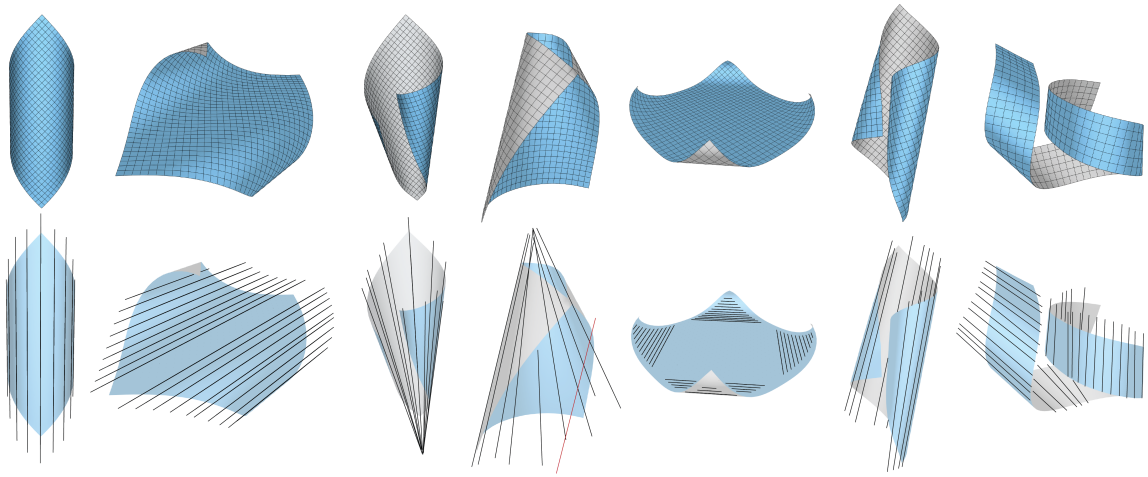


Figure 4.4: *Discrete developable geodesic nets and their vertex based rulings. From left to right: Two cylinders, two cones, a planar region connected to four cylinders, two tangent developable surfaces. Remarkably, the rulings tend to fit the shapes globally, despite their entirely local definition (Def. 8). They are, however, less stable around planar regions due to the calculation of N_x, N_y (see the red ruling in the center). We visualize the rulings sparsely for clarity. See the accompanying video at [Rabinovich et al. 2018a] for a three dimensional view.*

4.3 DOG Laplacian and mean curvature

Here we derive one a Laplacian, one of the most fundamental tools for deforming and smoothing geometries, specifically for DOGs. By utilizing the regularity of

our models, we derive a Laplacian that converges under sampling of an analytical net. Besides convergence under sampling, our DOG Laplacian possesses the desired properties of a discrete Laplace operator: it is symmetric, positive semidefinite and linearly precise. We then continue to define a convergent notion of mean curvature on DOGs, useful for editing tasks.

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 Alexa and Wardetzky [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 Laplacian satisfying a desired set of properties on DOGs.

4.3.1 Related work

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 section, 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 Chapter 3, 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. 4.1). 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.

4.3.2 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. 4.5). Therefore, for a vertex with index i in F, the area is $A_i = \frac{1}{4}\sum_{j\in\text{Quads}(i)} Q_j$, where Q_j is the planar quad area of face j.

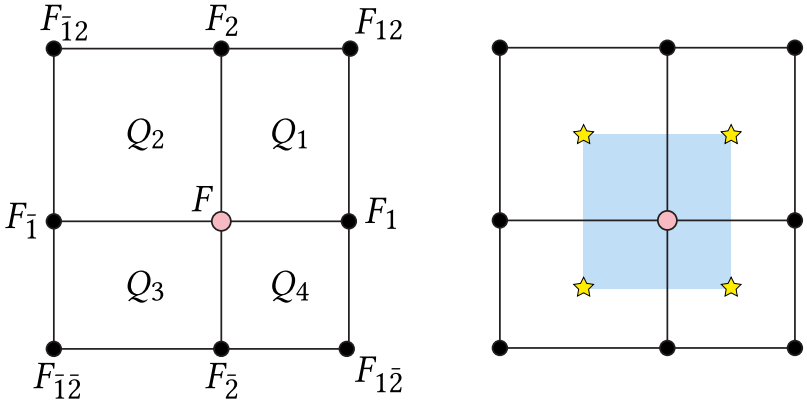


Figure 4.5: *Left: A planar DOG net around a vertex F (colored light red) and its four surrounding quads, whose areas are denoted by Q_i. Right: The area of the vertex is the area of its dual face (in blue), formed by connecting the centroids of its neighboring faces (marked by stars).*

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. 4.6. As explained in Sec. 3.5, DOGs are typically not discrete Chebyshev nets, and do not have opposite edges with exactly the same length. As also explained at Sec. 3.5 orthogonal geodesic nets, the smooth analogue for DOGs, are in fact Chebyshev, hence a DOG net is approximately a discrete Chebyshev net and approaches being exactly one in the (smooth) limit. We therefore define the area of a quad in a DOG net by a symmetric formula:

Definition 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. 4.6.

Definition 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. 4.5.

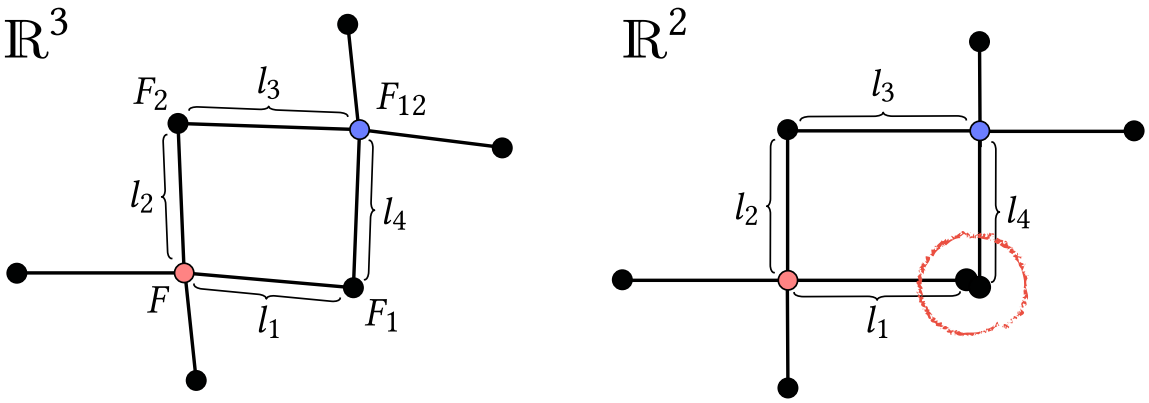


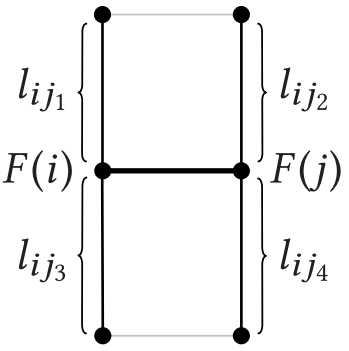
Figure 4.6: *Planar DOG nets are orthogonal grids, but for general DOG nets, one can locally flatten each star while exactly preserving the edge lengths, but generally not the entire net. Left: a DOG net in* $\mathbb{R}^3$ *containing two stars (red and blue). Right: individually flattened red and blue 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* $\frac{1}{4}(l_1 + l_3)(l_2 + l_4)$.

4.3.3 DOG Laplacian

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

$$\begin{aligned} \mathbb{L}F(i) = \frac{\partial \mathcal{A}}{\partial F(i)} &= \sum_{j \in \mathcal{N}(i)} \omega_{ij} \left(F(i) - F(j)\right), \\ \omega_{ij} &= \frac{1}{4} \cdot \frac{l_{ij_1} + l_{ij_2} + l_{ij_3} + l_{ij_4}}{\|F(i) - F(j)\|}, \end{aligned} \tag{4.2}$$

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).



To see this, we first compute the gradient of the area of a single DOG quad, defined in Sec. 4.3.2 as $Q = \frac{1}{4}(l_1 + l_3)(l_2 + l_4)$ using the notation of Fig. 4.6. We note that:

$$\frac{\partial \|F - F_1\|}{\partial F} = \frac{F - F_1}{\|F - F_1\|}. \tag{4.3}$$

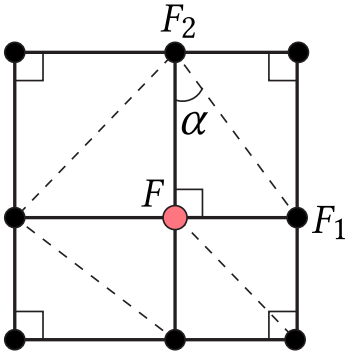
Plugging this in and using the chain rule leads to

$$\begin{aligned}\frac{\partial Q}{\partial F} =& \frac{1}{4}\frac{\partial(\|F-F_1\|+\|F_{12}-F_2\|)(\|F-F_2\|+\|F_{12}-F_1\|)}{\partial F} = \\ & \frac{1}{4}\left(\frac{\|F-F_2\|+\|F_{12}-F_1\|}{\|F-F_1\|}(F-F_1) + \right. \\ & \left. \frac{\|F-F_1\|+\|F_{12}-F_2\|}{\|F-F_2\|}(F-F_2)\right).\end{aligned} \tag{4.4}$$

Eq. (4.2) now follows by summing up the contribution of all faces incident on F and rearranging the terms per edge.

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.

The following theorem shows that on planar meshes the DOG Laplacian weights coincide with the cotangent Laplacian weights [Pinkall and Polthier 1993], implying that it vanishes on planar nets, i.e. DOG laplacians satisfy the liner precision property.



Theorem 14. If a DOG vertex and its 8 surrounding neighbors lie on a plane then the DOG Laplacian weights on its neighbouring edges coincide with cotangent weights ([Pinkall and Polthier 1993]) of any triangulation of the planar mesh.

Proof. 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. □

4.3.4 Laplacian mean curvature normal

The Laplacian Δ on a smooth surface f satisfies

$$\Delta f = -2Hn, \tag{4.5}$$

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 15. 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:

$$-\Delta f = 2Hn = k^x n^x + k^y n^y. \tag{4.6}$$

Proof. Let II 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 $k^x = |II(f_x, f_x)|\,,\ k^y = |II(f_y, f_y)|$. Let e_1, e_2 be the principal directions of the developable surface f such that $II(e_1, e_1) = 2H,\ II(e_2, e_2) = 0$, 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 $k^x + k^y = |II(f_x, f_x)| + |II(f_y, f_y)| = 2H\cos^2(\theta) + 2H\sin^2(\theta) = 2H$. □

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 Def. 1, as the unique bisector for both curves in the respective osculating planes.

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

$$\begin{aligned} \mathbb{L}F &= \frac{\partial \mathcal{A}}{\partial F} = -A\,(K^1 N + K^2 N), \\ K^j &= \frac{2\,\|\delta_j F + \delta_{\bar{j}} F\|}{\|F_j - F\| + \|F_{\bar{j}} - F\|} = \frac{4\cos\frac{\beta_j F}{2}}{\|F_j - F\| + \|F_{\bar{j}} - F\|}, \quad j = 1, 2. \end{aligned} \tag{4.7}$$

Proof. Let Q_1, Q_2, Q_3, Q_4 be the quad areas around vertex F, as denoted in Fig. 4.5. Assuming the DOG is Chebyshev, the following holds:

$$\begin{aligned} \|F_{12} - F_1\| &= \|F_2 - F\| = \|F_{\bar{1}2} - F_{\bar{1}}\|, \\ \|F_1 - F_{12}\| &= \|F - F_{\bar{2}}\| = \|F_{\bar{1}} - F_{\bar{1}2}\|, \\ \|F_{12} - F_2\| &= \|F_1 - F\| = \|F_{1\bar{2}} - F_{\bar{2}}\|, \\ \|F_2 - F_{\bar{1}2}\| &= \|F - F_{\bar{1}}\| = \|F_{\bar{2}} - F_{\bar{1}\bar{2}}\|. \end{aligned} \tag{4.8}$$

Plugging this into Eq. (4.4), we get:

$$\begin{aligned}
\frac{\partial Q_1}{\partial F} &= -\frac{\|F-F_2\|}{2}\delta_1 F - \frac{\|F-F_1\|}{2}\delta_2 F,\\
\frac{\partial Q_2}{\partial F} &= -\frac{\|F-F_{\bar{1}}\|}{2}\delta_2 F - \frac{\|F-F_2\|}{2}\delta_{\bar{1}} F,\\
\frac{\partial Q_3}{\partial F} &= -\frac{\|F-F_{\bar{2}}\|}{2}\delta_{\bar{1}} F - \frac{\|F-F_{\bar{1}}\|}{2}\delta_{\bar{2}} F,\\
\frac{\partial Q_4}{\partial F} &= -\frac{\|F-F_1\|}{2}\delta_{\bar{2}} F - \frac{\|F-F_{\bar{2}}\|}{2}\delta_1 F.
\end{aligned} \tag{4.9}$$

And therefore:

$$\frac{\partial \mathcal{A}}{\partial F} = \sum_{i=1}^{4}\frac{\partial Q_i}{\partial F} = -\left(s^2\left(\delta_1+\delta_{\bar{1}}\right)+s^1\left(\delta_2+\delta_{\bar{2}}\right)\right), \tag{4.10}$$

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. (4.1), the above is a linear combination of two vectors parallel to N. Plugging in the vertex area $A = s^1 s^2$, we get:

$$\begin{aligned}
\frac{\partial \mathcal{A}}{\partial F} &= -s^1 s^2\left(\frac{\delta_1 F+\delta_{\bar{1}}F}{s^1}+\frac{\delta_2 F+\delta_{\bar{2}}F}{s^2}\right) = \\
&= -A\left(\frac{2\left\|\delta_1 F+\delta_{\bar{1}}F\right\|}{\|F-F_1\|+\|F-F_{\bar{1}}\|}+\frac{2\left\|\delta_2 F+\delta_{\bar{2}}F\right\|}{\|F-F_2\|+\|F-F_{\bar{2}}\|}\right)N.
\end{aligned} \tag{4.11}$$

□

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 $\int_A \left(k^x+k^y\right)dA$, which can be discretized using Lemma 16. In that sense, Lemma 16 shows how the Laplacian mean curvature is analogous to Theorem 15, where K^1, K^2 are the curvatures of the discrete coordinate curves.

Definition 4. The normal curvatures of a DOG net, K^1, K^2, are defined as:

$$K^j = \frac{2\left\|\delta_j F+\delta_{\bar{j}}F\right\|}{\|F_j-F\|+\|F_{\bar{j}}-F\|} = \frac{4\cos\frac{\beta_j F}{2}}{\|F_j-F\|+\|F_{\bar{j}}-F\|},\quad j=1,2. \tag{4.12}$$

Definition 5. The mean curvature of a DOG net is $H = \frac{K^1+K^2}{2}$.

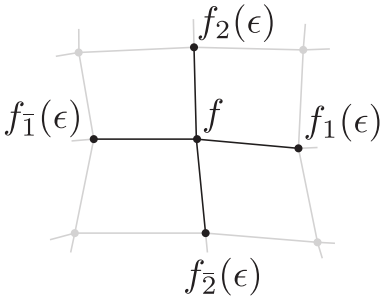
One could devise a Laplacian that recovers the discrete Gauss map precisely for every DOG net, not necessarily Chebyshev, by defining an area of a star using weights

induced only by its edge lengths and avoiding the averaging in Def. 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.

4.3.5 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. 2018a]. 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 17. If f is an analytical smooth orthogonal geodesic net, then the following holds:

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

Proof. 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:

$$\lim_{\epsilon \to 0} K^1 N = \lim_{\epsilon \to 0} 2 \frac{\delta_1 f + \delta_{\bar{1}} f}{\|f_1 - f\| + \|f_{\bar{1}} - f\|}. \tag{4.13}$$

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. 2018a]:

$$\begin{aligned}
\delta_1 f &= \frac{f_x}{\|f_x\|} + \epsilon\left(-\frac{\langle f_{xx}, f_x\rangle}{2\langle f_x, f_x\rangle^{3/2}} f_x + \frac{f_{xx}}{2\|f_x\|}\right) + o(\epsilon^2),\\
\delta_{\bar{1}} f &= -\frac{f_x}{\|f_x\|} + \epsilon\left(-\frac{\langle f_{xx}, f_x\rangle}{2\langle f_x, f_x\rangle^{3/2}} f_x + \frac{f_{xx}}{2\|f_x\|}\right) + o(\epsilon^2),\\
\delta_1 f + \delta_{\bar{1}} f &= 2\epsilon\left(-\frac{\langle f_{xx}, f_x\rangle}{2\langle f_x, f_x\rangle^{3/2}} f_x + \frac{f_{xx}}{2\|f_x\|}\right) + o(\epsilon^2). \qquad (4.14)
\end{aligned}$$

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

$$f_x = \lambda t,\ f_{xx} = \lambda_x t + \lambda^2 \kappa n,\ \lambda_x = \langle f_x, f_x\rangle_x = \frac{\langle f_{xx}, f_x\rangle}{\|f_x\|}. \qquad (4.15)$$

Plugging this in Eq. (4.14), 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. (4.13) 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 12 it is also a Chebyshev net satisfying $\|f_x\|_y = \|f_y\|_x = 0$, and therefore Eq. (4.8) 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. □

4.4 Ruling angles, curves torsion, cylinders and cones

Here we show how higher order information can be derived from local information on DOGs. We start by noting the connection between the coordinate line curvatures of orthogonal geodesic nets and its second fundamental form.

Let f be a smooth orthogonal geodesic net with coordinate tangents directions f_x, f_y, curvatures k_x, k_y and mean curvature $H \neq 0$. The second fundamental form of f, $II(v)$, can be written in coordinates f_x, f_y as $II(v) = v^t R^T D R v$, where:

$$R = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix}, D = \begin{pmatrix} 2H & 0 \\ 0 & 0 \end{pmatrix} \tag{4.16}$$

With $H \geq 0$, $0 \leq \theta \leq \pi$. As noted before, $H = \frac{k_x + k_y}{2}$ as the curves are both geodesics and orthogonal. The value θ then satisfies the following:

$$tan^2(\theta) = \frac{k_x}{k_y}$$

Thus we can tell the ruling, up to a symmetry, by just looking at the curvatures of the coordinate curves. Given the curvature there are only 2 possible values for curvatures, a disambiguate that can often be resolved especially when considering smooth flows on orthogonal geodesic nets, with a given starting ruling direction.

A flow $f(t)$ on a developable surface that preserve the angle between the rulings and the coordinate lines satisfies $\theta(t)' = 0$ for every time step of the flow, i.e. keeps the ratio $\frac{k_x}{k_y}(t)$ fixed when deforming the surface. This is analogue to rigid, origami like folding of a paper.

The value of θ in a given point on a surface does not shed any information on the geometry parameterized, as it depends on the arbitrary rotation of the orthogonal coordinate tangents from the curvature lines. However, the derivative of θ along a fixed coordinate line, for instance θ_x, does characterize the geometry: if $\theta_x = 0$, meaning θ is a constant, the rulings angle to the tangents do not change and they are parallel, hence the surface is cylindrical. If θ_x is constant but not zero, i.e. θ is a linear function along the coordinates, then the surface is locally conical [Graustein 1917]. These conditions do not suffer from the symmetry or ambiguity of θ as they are defined on its derivative and could be used in optimization, offering a simpler intrinsic and more local alternative to the extrinsic rulings defined by cross products of the Gauss map at Sec. 4.2.2.

We also note that the torsion of two orthogonal geodesic curves is equal and satisfies [Graustein 1917]:

$$\tau^2 = k_x k_y$$

We note that the discrete analogues to τ^2, as well as $tan^2(\theta)$ can be directly discretized by using the definitions of K_1, K_2 above. Moreover, the change of these quantities, i.e. their derivatives, can also be discretized. For instance, one can constrain rulings preserving flows by demanding that the discrete change in θ vanishes, i.e. $\theta' = 0$ just by keeping the ratio $\frac{K_1}{K_2}$ constant at a given chosen set of points, or similarly optimize a shape to be cylindrical or conical at some areas. All of the above mentioned quantities benefit from the convergence analysis at Theorem 17, and also converge under sampling.

Lastly, we note that one does not necessarily need to compute $K_x K_y$ to get a discrete notion of squared torsion for the parameter lines, but simply look at the opening angle of a star, α (see Fig. 4.2 for notation), by the following theorem.

Theorem 18. Let F be a point on a DOG net, N it's normal, α be it's opening angle and β_1, β_2, A and be the star vertex area, then $4cos(\alpha) = AK_1K_2$.

Proof. The angles the outgoing edges make with the vertex normal are $\frac{\pi+\beta_1}{2}$ and $\frac{\pi+\beta_2}{2}$. Next consider the spherical triangle that is formed by the normal N and two neighbouring edges. It has sides $\frac{\beta_1}{2}$ and $\frac{\beta_2}{2}$ and α, and from spherical geometry $\cos(\alpha) = cos(\frac{\beta_1}{2})cos(\frac{\beta_2}{2})$ □

One can immediately see that 0 discrete torsion on a given point corresponds to $K_1 = 0$ or $K_2 = 0$, implying that $\alpha = \frac{\pi}{2}$. In particular, just as in the smooth case, a DOG whose coordinate curves have vanishing torsion parameterizes a cylindrical shape in curvature line parameterization.
This definition together with Theorem 13 at Sec. 3.5 helps us see that Chebyshev DOG nets are a model limited to only smooth developables with the same equal torsion of the orthogonal coordinate curves throughout the entire net. Cylindrical shapes in curvature line parameterization, as well as a parameterization of a circular cylinder by orthogonal geodesics helices are examples of such nets.

CHAPTER 5

The shape space of discrete orthogonal geodesic nets

This chapter takes a deeper look into the angle constraints behind DOGs, tackling two main questions:

1. What are the shapes that can be modeled with DOGs?
2. How to deform DOGs?

Question (1) is addressed at Sec. 5.1, and deals with the rigidity and flexibility of DOGs. The analysis at Sec. 3.4.1 shows that DOGs can be used to approximate any smooth developable surface. Thus, DOGs are flexible enough. A good developable discrete model should also be sufficiently restrictive, or rigid, as to not model smooth doubly curved shapes.

Question (2) is of both practical and theoretical importance and is tackled in Sec. 5.2. We develop essential tools to effectively deform and model DOGs. We achieve this by looking at smooth deformations on DOGs. The mere existence of such flows depends on the geometry of the constraints. We start with the theoretical backbone: we prove that such flows generally exist, and we count exactly how many. The theory developed will lead to practical guarantees on the feasibility of optimization problems for DOGs modeling.

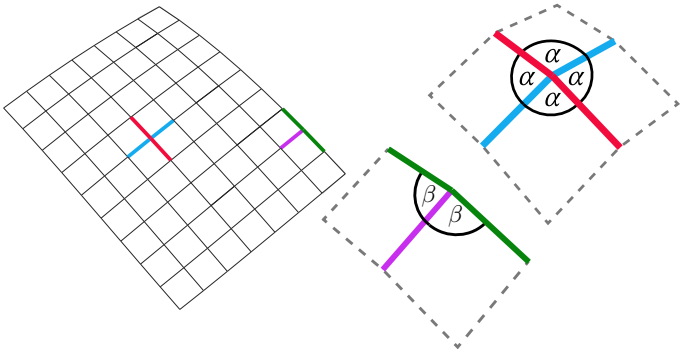
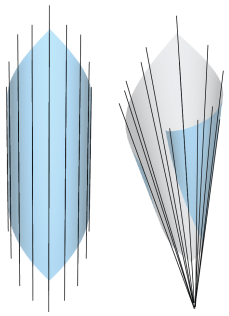


Figure 5.1: *DOGs are a set of highly constrained regular quadrilateral meshes, with almost as much as constraints as variables. The constraints are non-linear and are not convex. Our shape space analysis shows the existence of a rich set of smooth flows on DOGs, and provides guarantuees on the feasibility when optimizing and modeling DOGs.*

5.1 The rigidity of DOGs

5.1.1 Rigidity through surface extension and constraints evolution

Applying a deformation on a smooth developable surface locally generally dictates its shape globally. One way to see this is by looking at the rulings;



On a smooth developable surface, the rulings are global, in the sense that they either extend infinitely, or their endpoints must hit the boundaries of the surface [Spivak 1999]. Flipping this point of view, one can ask how to extend a developable surface at its boundary: the possibilities are generally quite limited, since the points along the rulings are uniquely determined. Note that arbitrarily extending rulings often results in singularities. Our discrete model shares a similar rigid structure, as shown in the following analysis. The analysis is in the spirit of other geometric constructions for discrete nets in discrete differential geometry, for instance the one at Fig. 2.13.

Assume we have a vertex F in our discrete net, as well as some neighboring vertices to its left (or right) and bottom (Fig. 5.2, right). The position of the top neighbor F_2 is then generally uniquely determined, as shown by the following two lemmas. Therefore, a given discrete orthogonal net can generally be extended at its boundary by setting only a small number of parameters, as illustrated in Fig. 5.3. The process is analogue to the smooth case explained above, but it is not based on rulings.

Lemma 19. (Direction propagation). Let F be a vertex in an orthogonal geodesic

net that has only three neighbors $F_1, F_{\bar{2}}, F_{\bar{1}}$ such that the discrete curve Γ_1 through $F, F_1, F_{\bar{1}}$ is non-degenerate, and the two angles around F are equal. Then there is a unique direction $\delta_2 F$ such that $F, F_1, F_{\bar{2}}, F_{\bar{1}}, F_2$ is an orthogonal geodesic star (where F_2 lies on the ray through $\delta_2 F$; see Fig. 5.2).

Proof. By Theorem 6, the vector $\delta_2 F$ must be in the direction of the reflection of $\delta_{\bar{2}} F$ w.r.t. the plane Π_1 spanned by $F, F_1, F_{\bar{1}}$. □

In the case where Γ_1 is a straight line, there is a family of solutions consisting of all vectors that are orthogonal to Γ_1.

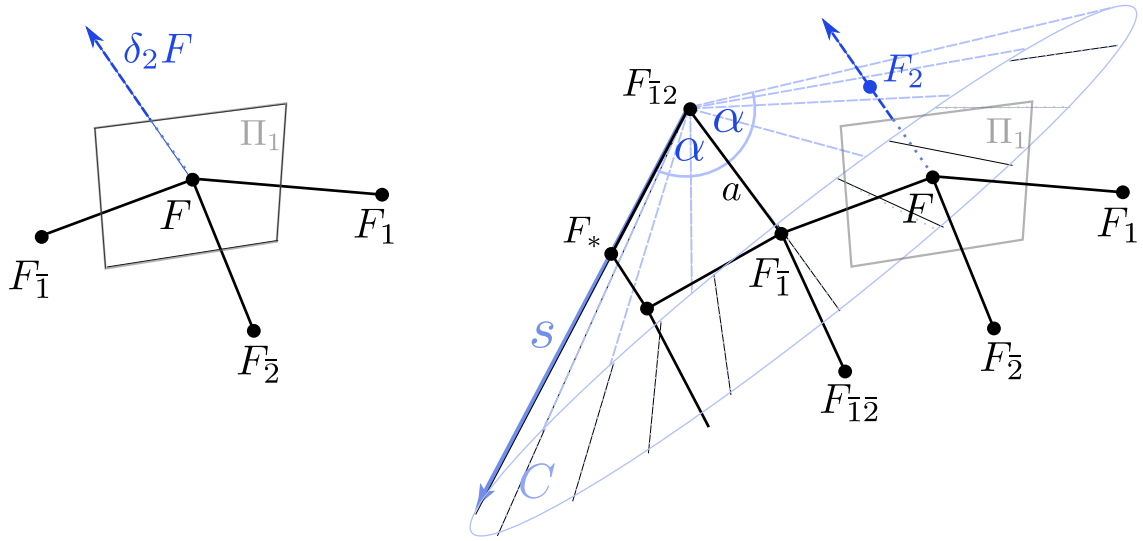


Figure 5.2: *Left: By Lemma 19 the direction $\delta_2 F$ is the reflection of $\delta_{\bar{2}} F$ w.r.t. the plane Π_1. Right: By Lemma 20, the same direction $\delta_2 F$ intersects a cone C with the apex at $F_{\bar{1}2}$, determining the position of the point F_2.*

Lemma 20. (Cone-ray intersection). Given a vertex F in an orthogonal geodesic net that has at least all the neighboring vertices denoted as in Fig. 5.2 (right side). Let C be the cone or plane generated by revolving the ray s emanating from $F_{\bar{1}2}$ through F_* about the axis $a = F_{\bar{1}} - F_{\bar{1}2}$ (see Fig. 5.2). Then, the vertex F_2 has to lie on the intersection of C and a line emanating from F (Fig. 5.2).

Proof. By Def. 1, the angle α between the net edges $a = F_{\bar{1}} - F_{\bar{1}2}$ and $F_2 - F_{\bar{1}2}$ must be equal to the angle between a and $F_* - F_{\bar{1}2}$, and so F_2 must lie on C. □

Given the construction for F_2 above, we see that, speaking informally, extending a discrete orthogonal geodesic net by one vertex at its boundary is a determined process if we already have neighbors below and to the left or to the right. The only degrees of freedom are available when one begins adding a new row to the grid, without yet having neighbors on the left or right but only below, see Fig. 5.3. Assuming general

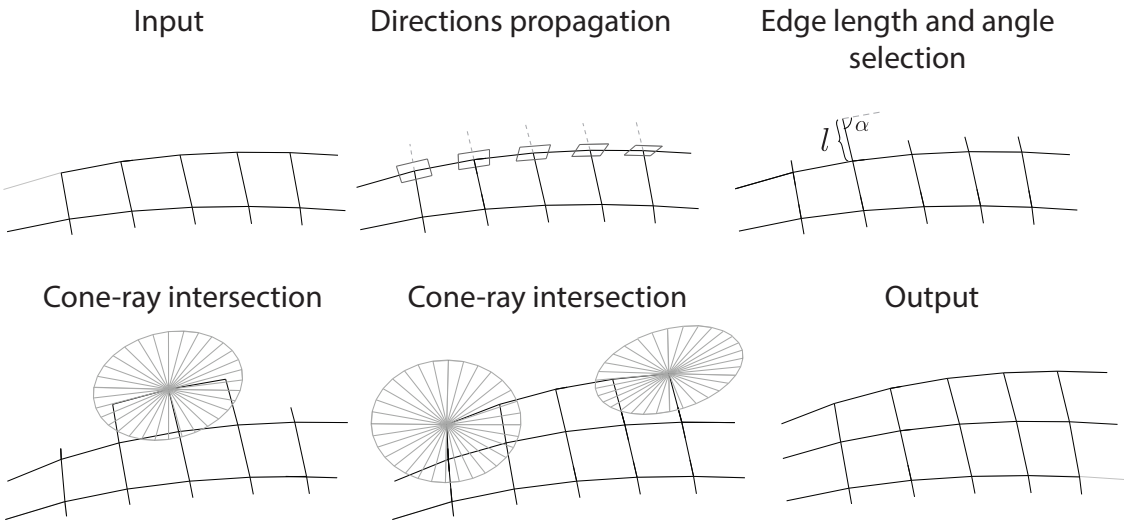


Figure 5.3: *Extension of a discrete orthogonal net. Given a choice of two parameters: edge length l and one angle α, the row propagates by Lemmas 19 and 20.*

position, we first use Lemma 19 to compute the directions of the new net edges that point upwards. We can then select the length l of the first new edge, effectively setting a vertex of the new row, as well as the cone half-angle α for the first cone C of the new row. Then, the remaining vertices of the row are determined using Lemma 20, as illustrated in Fig. 5.2 and Fig. 5.3.

5.2 Smooth flows on DOGs

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.

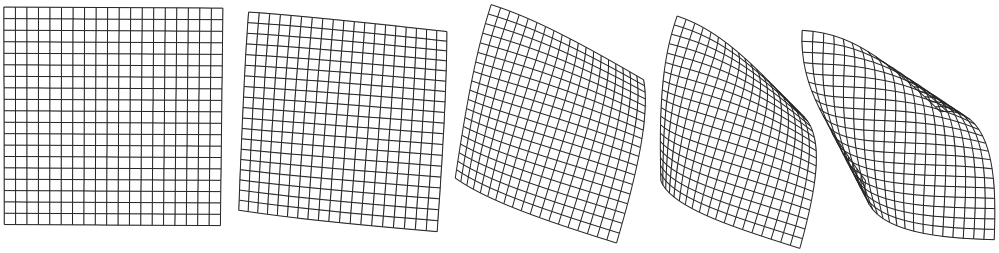


Figure 5.4: *Smooth flow on a DOG.*

5.2.1 Related work

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. [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. 5.13).

5.2.2 A computational framework for deforming DOGs

Modeling DOGs requires solving highly constrained and nonlinear optimization problems, yet as we will see one can generally guarantee the existence of nearby solutions if one starts at a feasible point. This observation is useful in practice: DOGs exploration performs well using smooth flows that keep the DOG constraints feasible. In the optimization, methods to solve an optimization problem by smoothly interpolating the objective are known as homotopy based optimization methods [Nocedal and Wright 2006] (see Fig. 5.6).

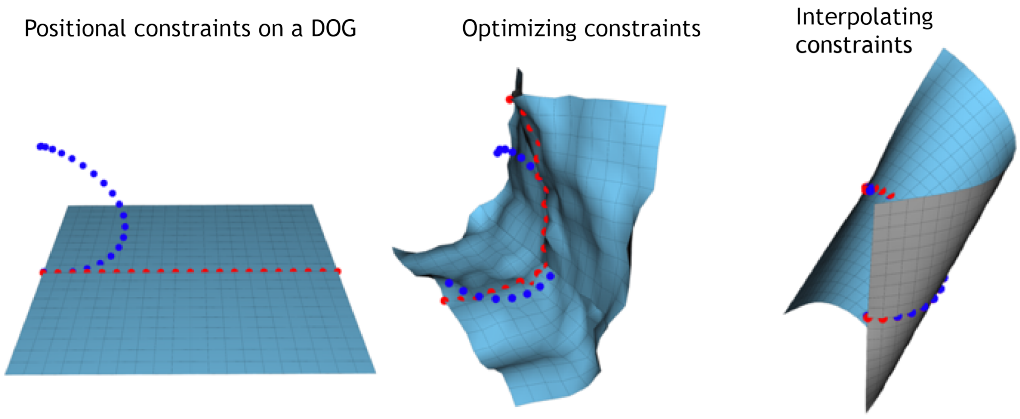


Figure 5.5: *Setting soft positional constraints on a curve (left). The same optimization algorithm fails completely when setting all constraints at once, returning a mesh that does not satisfy the non-linear DOG constraints (center). In contrast, using a homotopy based method to interpolate the curve positional constraints by isometrically interpolating it returns a smooth DOG (right).*

Our analysis answers two main questions:

1. Can DOGs smoothly deform?
2. How to deform DOGs?

Question (1) can be answered for a set of DOGs we call *generic DOGs*.

Definition 9. 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 10. 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.

We prove that the space $\mathcal{M}_G$ is a smooth manifold. Smooth flows can then be constructed by a smooth choice of vectors on the manifold's tangent spaces, possibly selected to minimize a desired smooth objective function. We will show two proofs for the fact that $\mathcal{M}_G$ is a smooth manifold, each with its own merits.

The first is a direct consequence of the geometric construction at Sec. 5.1.1. Indeed using the construction at Fig. 5.3, we can see that each mesh in $\mathcal{M}_G$ is locally defined by a set of smooth parameters: a set of coordinates for a single curve, a set of directions for edges emanating from that curve and an additional edge length (l) and angle (α) for each row of quads. Smoothly varying each one of these parameter then smoothly deforms a DOG, while keeping the angle constraints satisfied exactly. This

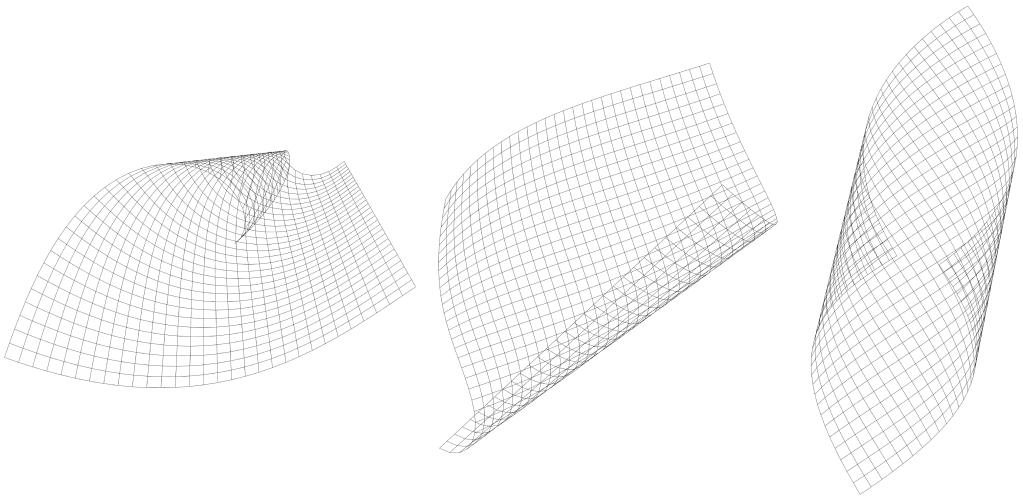


Figure 5.6: *Generic DOGs have at most straight lines in one coordinate axis.*

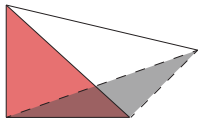
geometric construction is similar to other constructions for nets in discrete differential geometry ([Bobenko and Suris 2008]) and also has an important consequence for Chapter 6. Directly parameterizing generic DOGs with this construction however is not practical. In fact, merely constructing them using these minimal set of parameters is extremely unstable, hence we resort in practice to an implicit representation of the mesh as a set of vertices satisfying a set of constraints, as common in optimization.

The other proof better fits such an implicit representation and use the combinatorics and geometry of the DOGs constraints. It allows us to also answer question (2), by offering a computational tool to compute tangents on the smooth manifold to minimize an objective while also offering insight relevant to singularities handling. Our key technique is analyzing the linear dependence of the gradients of a set of angle constraints by characterizing a *constraints graph* of minimally dependent gradients using a few geometric observations.

Throughout the section, 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 6. 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.

5.2.3 Notation and setup

Following the notation of Sec. 3.2 and Sec. 4.1, we denote our net by the map $F : V \to \mathbb{R}^3$, where $V \subseteq \mathbb{Z}^2$. 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*. Slightly abusing notation, an individual vertex is then called simply F, vertices in its vicinity are referenced using lower shift indices as in Sec. 3.2, and the inner angles around F as α_j, ordered consecutively clockwise (see Fig. 3.2). We denote the angles of the coordinate curves by $\beta_1 F, \beta_2 F$. 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 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 $\mathbf{F} = (F(1), \ldots, F(|F|) \in \mathbb{R}^n$.

5.2.4 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. 3.2). 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 $\phi_i(\mathbf{F}) = 0$:

$$\begin{aligned} \phi_1(\mathbf{F}) &:= \cos(\alpha_1) - \cos(\alpha_2) = 0, \\ \phi_2(\mathbf{F}) &:= \cos(\alpha_2) - \cos(\alpha_3) = 0, \\ \phi_3(\mathbf{F}) &:= \cos(\alpha_3) - \cos(\alpha_4) = 0. \end{aligned} \tag{5.1}$$

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.

5.2.5 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 21. $\mathcal{M}_s$ 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 $\mathbb{R}^3$.

Proof. 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 Sec. 3.3. 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. □

The parameters are illustrated in Fig. 5.7. 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. (7.15), i.e., we can view $\mathcal{M}_s$ as a 12-dimensional manifold embedded in a 15-dimensional ambient space.

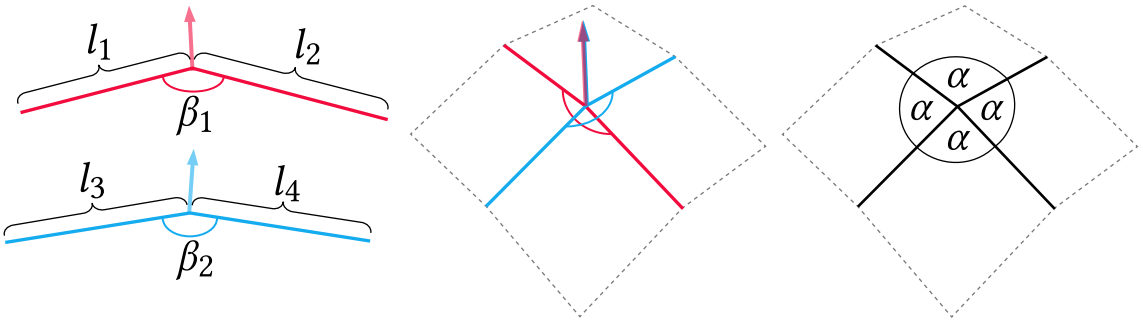


Figure 5.7: *Parameterization of a DOG star. By Theorem 21, a single proper DOG star is determined up to rigid motion by its 4 edge lengths $l_j, j = 1..4$ and 2 curve coordinate angles β_1, β_2 (left). Reconstructing the star from these parameters requires translating and rotating one of the discrete 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.*

5.2.6 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 22. *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. 5.8. We

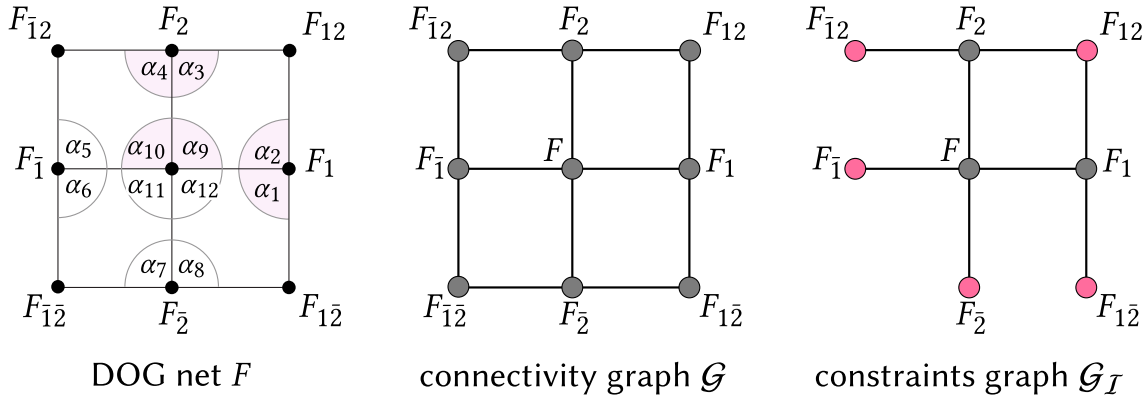


Figure 5.8: *A DOG net F with a 3×3 grid connectivity graph $\mathcal{G}$ and a constraints graph $\mathcal{G}_{\mathcal{I}}$. The notation for the angles in this example is chosen such that adjacent constrained angles have consecutive indices. On the right we show the constraints graph $\mathcal{G}_{\mathcal{I}}$ 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.*

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 $j = 1, 3, 5, 7, 9, 10, 11$, 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

$$J_F = \begin{pmatrix} \frac{\partial \phi_1}{\partial \mathbf{F}} \\ \frac{\partial \phi_2}{\partial \mathbf{F}} \\ \vdots \\ \frac{\partial \phi_7}{\partial \mathbf{F}} \end{pmatrix} = \begin{pmatrix} \frac{\partial \phi_1}{\partial F_{\bar{1}\bar{2}}} & \frac{\partial \phi_1}{\partial F_{\bar{2}}} \cdots \frac{\partial \phi_1}{\partial F_{12}} \\ \frac{\partial \phi_2}{\partial F_{\bar{1}\bar{2}}} & \frac{\partial \phi_2}{\partial F_{\bar{2}}} \cdots \frac{\partial \phi_2}{\partial F_{12}} \\ & \vdots \\ \frac{\partial \phi_7}{\partial F_{\bar{1}\bar{2}}} & \frac{\partial \phi_7}{\partial F_{\bar{2}}} \cdots \frac{\partial \phi_7}{\partial F_{12}} \end{pmatrix}. \tag{5.2}$$

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 23. We use the angle notation of Fig. 3.2. 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. 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. □

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 21, Theorem 22 and Lemma 23 we get the following:

Theorem 24. The constraints in Eq. (7.15) 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 $\frac{\partial \phi_i}{\partial \mathbf{F}} = \frac{\partial\big(\cos(\alpha_{j_1}) - \cos(\alpha_{j_2})\big)}{\partial \mathbf{F}} \neq \vec{0}$ as a result of Lemma 23 and the fact that each cosine term involves a vertex that the other does not. Therefore by Theorem 22, $\mathcal{M}^i$ 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 21, $\mathcal{M}^s$ 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 23.

5.2.7 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 $\mathcal{C}_\mathcal{I}$ the $|\mathcal{I}| \times n$ *constraint matrix* formed by taking the rows of J corresponding to those indices.

Definition 7. 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 $\mathcal{C}_\mathcal{I}$ 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 $\mathcal{C}_\mathcal{I}$.

The rows of $\mathcal{C}_\mathcal{I}$ 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 $\mathcal{C}_\mathcal{I}$ (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 25. 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 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 $\mathcal{C}_\mathcal{I}$ (for the x, y, z coordinates) are not all zeros. By Lemma 25, 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. 5.8 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 26. 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 $\mathcal{C}_\mathcal{I}$ 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 $\mathcal{C}_\mathcal{I}$ to cancel a given row, w.l.o.g.

the first row. By Lemma 23 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 9. 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 10. 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. 5.8.

Lemma 27. (See Fig. 5.9, 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

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

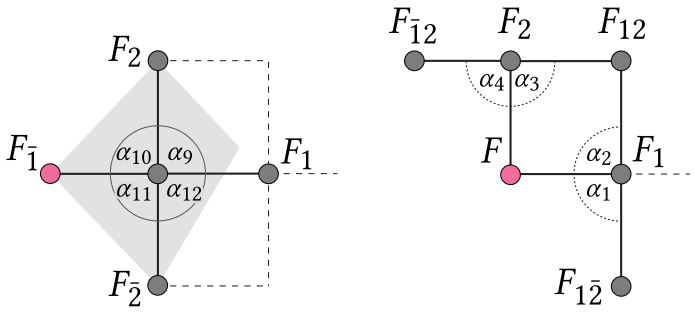
There exist a_1, a_2 that are not both zero, such that

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

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

$$a_1\frac{\partial\left(\cos(\alpha_{11})-\cos(\alpha_{10})\right)}{\partial F_{\bar{1}}}=a_2\frac{\partial\cos(\alpha_{11})}{\partial F_{\bar{1}}}. \tag{5.3}$$

By Lemma 23, 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. (5.3) also implies $\frac{\partial\cos(\alpha_{10})}{\partial F_{\bar{1}}}$ is parallel to $\frac{\partial\cos(\alpha_{11})}{\partial F_{\bar{1}}}$. By Lemma 23, 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 27 Lemma 28

Figure 5.9: *Illustrations for lemmas about leaves and corners in a constraints graph $\mathcal{G}_\mathcal{I}$ of a circuit. If $F_{\bar{1}}$ is a leaf vertex connected to F, then all neighbors of F are in $\mathcal{G}_\mathcal{I}$, 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 $\mathcal{G}_\mathcal{I}$.*

Lemma 28. Let F be a corner in a circuit's constraints graph $\mathcal{G}_\mathcal{I}$ of a proper net. The neighbors of F are not corners in $\mathcal{G}_\mathcal{I}$. 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 $\mathcal{G}_\mathcal{I}$.

Proof. See Fig. 5.9 (right) for notation. As F is a corner in $\mathcal{G}_\mathcal{I}$ there are rows in $\mathcal{C}_I$ whose partial derivative w.r.t. to the coordinates of F do not vanish. By Lemma 25 these are linear combination of gradients constraining angles of nearby vertices. As $\mathcal{G}_\mathcal{I}$ 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 23, 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 $\mathcal{G}_\mathcal{I}$, 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 $\mathcal{G}_\mathcal{I}$. □

Lemma 29. Let $\mathcal{G}_\mathcal{I}$ be a circuit's constraints graph of a proper net, then $\mathcal{G}_\mathcal{I}$ 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 $\mathcal{G}_\mathcal{I}$, $V_L = \min_j \{F(j,h) \in \mathcal{G}_\mathcal{I}\}$. 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. 5.10. □

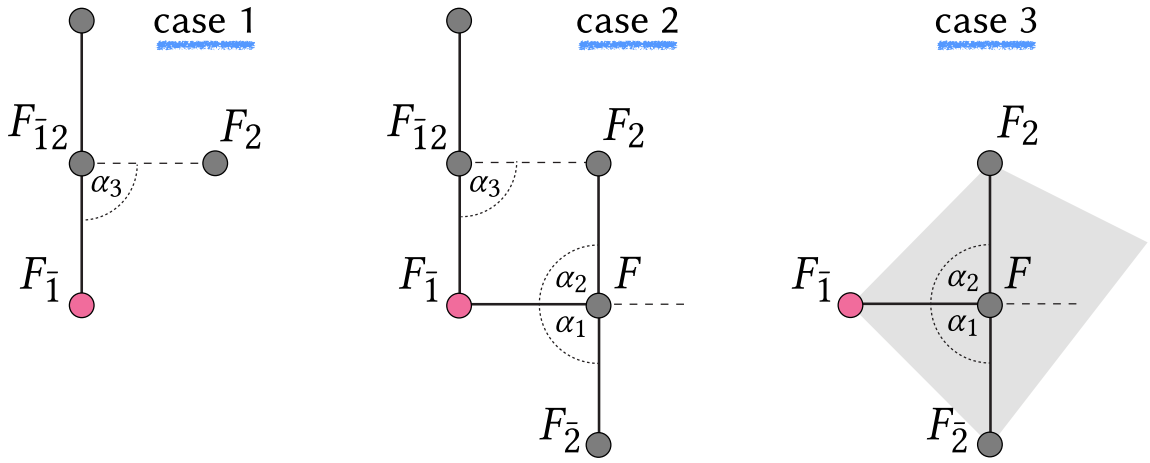


Figure 5.10: *Lemma 29. If $F_{\bar{1}}$ is the lowest vertex among all "leftmost" vertices, there are three cases for the neighborhood of $F_{\bar{1}}$ in the constraints graph $\mathcal{G}_{\mathcal{I}}$, as depicted above. Case 1 is not possible for a circuit constraints graph, since by Lemma 23, $\frac{\partial \cos(\alpha_3)}{\partial F_{\bar{1}}} \neq 0$. Case 2 is not possible due to Lemma 28, 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 27, $F_{\bar{1}}, F, F_2, F_{\bar{2}}$ are all in $\mathcal{G}_{\mathcal{I}}$ and lie on the same plane.*

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 $\mathcal{G}_{\mathcal{I}}$ to the geometry of DOG nets.

Lemma 30. Let $F_{\bar{1}}$ be a leaf connected to the right in a circuit's constraints graph $\mathcal{G}_{\mathcal{I}}$ of a DOG net, then $F, F_2, F_{\bar{2}}$ are collinear.

Proof. By Lemma 27, the neighbors of F are in $\mathcal{G}_{\mathcal{I}}$ 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. 2018a] (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. □

5.2.8 The shape space of DOG nets

Theorem 31. 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 29 and Lemma 30, 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 22. Note that the tangent space dimension $k = n - m$ is $O(|\partial F|)$, the number of boundary vertices, because n

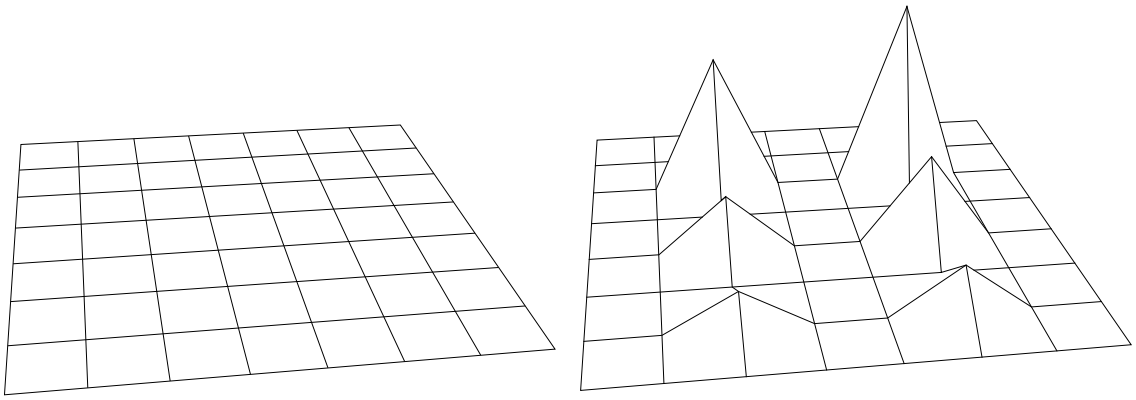


Figure 5.11: *A planar DOG net has more flows than a general typical DOG net, and it admits non-smooth flows, since one can push inner vertices in the z-direction independently from other vertices. This freedom is a discrete artifact that exists only on such singular nets, since typically DOG nets do not admit local deformations.*

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 32. $\mathcal{M}$ 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 ν_i. See Fig. 5.11. □

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 $\mathcal{M}_G$. It is not surprising that the existence of planar parts, once again, leads to singularities in the shape space, just as in the smooth case they add combinatorial degrees of freedom to the shape space in the form of possible sudden changes to the developable surface combinatorial decomposition.

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

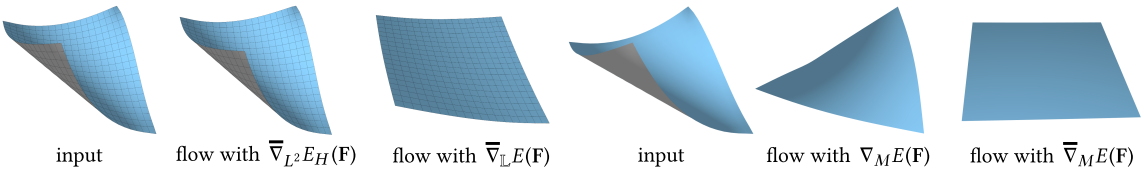


Figure 5.12: *The effect of shape space metric M (left) and gradient projection (right) on a flow minimizing squared mean curvature (as defined in Def. 5). Left: we show the input model and the results of 200 iterations of the flow (taking 1.4 seconds on a 20×20 grid), using the standard L^2 as metric 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.*

Theorem 33. 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 29 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 27, then the Jacobian of F^* has full row rank. □

5.2.9 Discretizing DOG Flows

A corollary of Theorem 31 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 the previous section. The resulting algorithm is summarized in Algorithm 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 the DOG Laplacian matrix $\mathbb{L}$ as the metric M, which we derived at Sec. 4.3. A comparison between using this metric vs. the standard L_2 metric is shown in Fig. 5.12 (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 $\overline{\nabla}_M E(\mathbf{F})$:

$$\overline{\nabla}_M E(\mathbf{F}) := \arg\min_{\mathbf{t}} \|\nabla_{\mathbf{M}} \mathbf{E}(\mathbf{F}) - \mathbf{t}\|_M, \quad \mathbf{t} \in T\mathcal{M}. \tag{5.4}$$

A comparison between a Willmore flow with and without projecting the gradients onto $T\mathcal{M}$ is shown in Fig. 5.12 (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. 5.2. 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:

$$\begin{aligned} K \begin{pmatrix} \overline{\nabla}_M E(\mathbf{F}) \\ \lambda \end{pmatrix} &= \mathbf{b} \\ K = \begin{pmatrix} M & J^\mathsf{T} \\ J & 0 \end{pmatrix}, \quad \mathbf{b} &= \begin{pmatrix} \nabla E(\mathbf{F}) \\ \mathbf{0} \end{pmatrix}. \end{aligned} \tag{5.5}$$

By Theorem 31, J has full row rank for a generic net. This implies that the system in Eq. (5.5) 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. (5.5), 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 $\phi(\mathbf{F}) = \boldsymbol{\varepsilon}$, with a small norm $\|\boldsymbol{\varepsilon}\| < \epsilon$. We therefore

solve the following system:

$$K \begin{pmatrix} \overline{\nabla}_M E(\mathbf{F}) \\ \lambda \end{pmatrix} = \mathbf{b}$$
$$K = \begin{pmatrix} M & J^\mathsf{T} \\ J & 0 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \nabla E(\mathbf{F}) \\ \boldsymbol{\varepsilon} \end{pmatrix}. \tag{5.6}$$

Handling singularities

The shape space $\mathcal{M}$ is not always a manifold locally, as shown in Theorem 32. 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 32). By Theorem 22, this also implies that the system in Eq. (5.5) is singular for planar $\mathbf{F}$.

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 33 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. (5.5) on the net F^*, which, by Theorem 33, 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. 5.11), 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 in [Rabinovich et al. 2018b] for various examples of editing starting from a singularity, as well as Fig. 5.13.

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. 2018a] 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. 5.14 and supplementary video in [Rabinovich et al. 2018b]). 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 section 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. (5.5) 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. 2018a] 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. 5.13).

5.2.10 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.

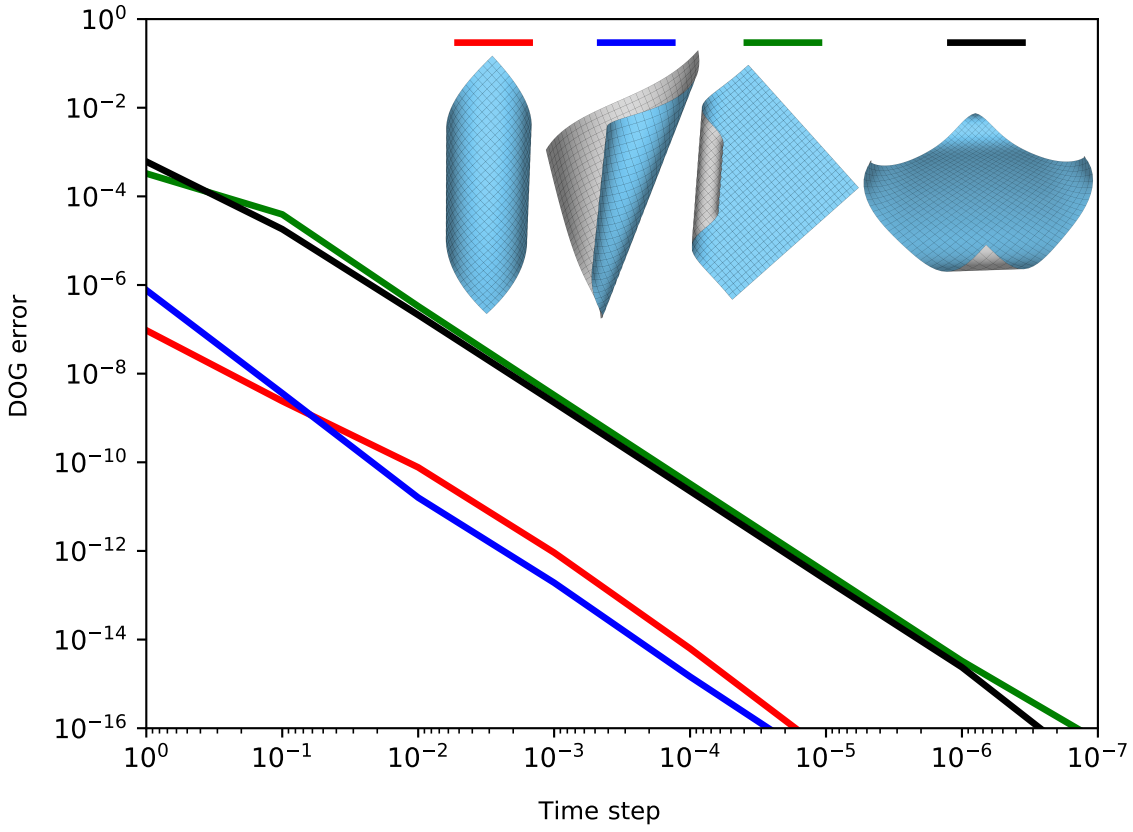


Figure 5.13: *Convergence plot of generic nets (red and blue) and singular nets (green and black) under Willmore flow as the flow time step goes to zero. We run Willmore flow until each mesh becomes planar. The vertical axis ("DOG error") represents the maximal value of the sum of squared errors of the DOG constraints, taken over the entire flow.*

Objective functions

Our first objective function is a bending minimization term defined as

$$E_H(\mathbf{F}) = \sum_{i \text{ vertex in } F} A_i \, H_i^2. \tag{5.7}$$

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. 5.12), 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:

$$E_{\text{iso}}(\mathbf{F}) = \sum_{e \text{ edge in } F} (l_e - l_e^0)^2. \tag{5.8}$$

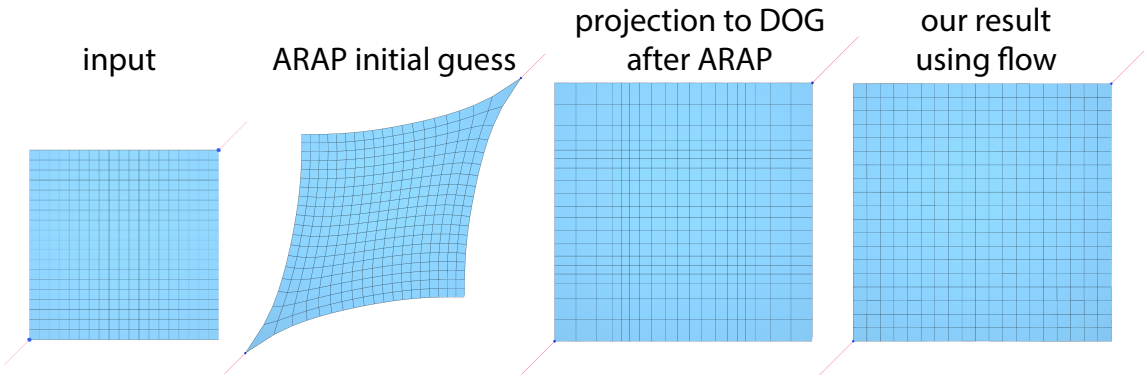


Figure 5.14: *Stretching a planar orthogonal grid along the diagonal by moving point handles at two diagonally opposing corners. The algorithm presented in Chapter 3 computes an initial guess with ARAP, and then projects the guess to a nearby DOG. Strongly stretching the surface causes a distorted ARAP initial guess lying far away from the DOG shape space, leading to a jittery editing experience (see the accompanying video at [Rabinovich et al. 2018b]), and also a 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* $w_H = 1$, $w_{\textit{uni-all}} = 0$, $w_{iso} = w_{\textit{uni-opp}} = 0$, $w_p = 0.5$ *in the flow objective.*

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. 5.14 and in the accompanying video for [Rabinovich et al. 2018b], that our algorithm naturally handles stretching, as opposed to the work of [Rabinovich et al. 2018a], 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. 5.14, 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 tangential deformations:

$$\begin{aligned} E_{\text{uni-all}}(\mathbf{F}) &= \sum_{\text{star in } F} (l_1 - l_2)^2 + (l_3 - l_4)^2 + (l_1 - l_3)^2 + (l_2 - l_4)^2, \\ E_{\text{uni-opp}}(\mathbf{F}) &= \sum_{\text{star in } F} (l_1 - l_3)^2 + (l_2 - l_4)^2, \end{aligned} \tag{5.9}$$

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

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

, 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.

Smoothing and subdivision

The definition of a DOG net does not encode smoothness, hence a DOG net can be jaggy, as in Fig. 5.11. This is common in discrete differential geometry [**?**], 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. 5.15 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 for [Rabinovich et al. 2018b]. We note that smoothing operations are not possible in the optimization framework of [Rabinovich et al. 2018a], 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.

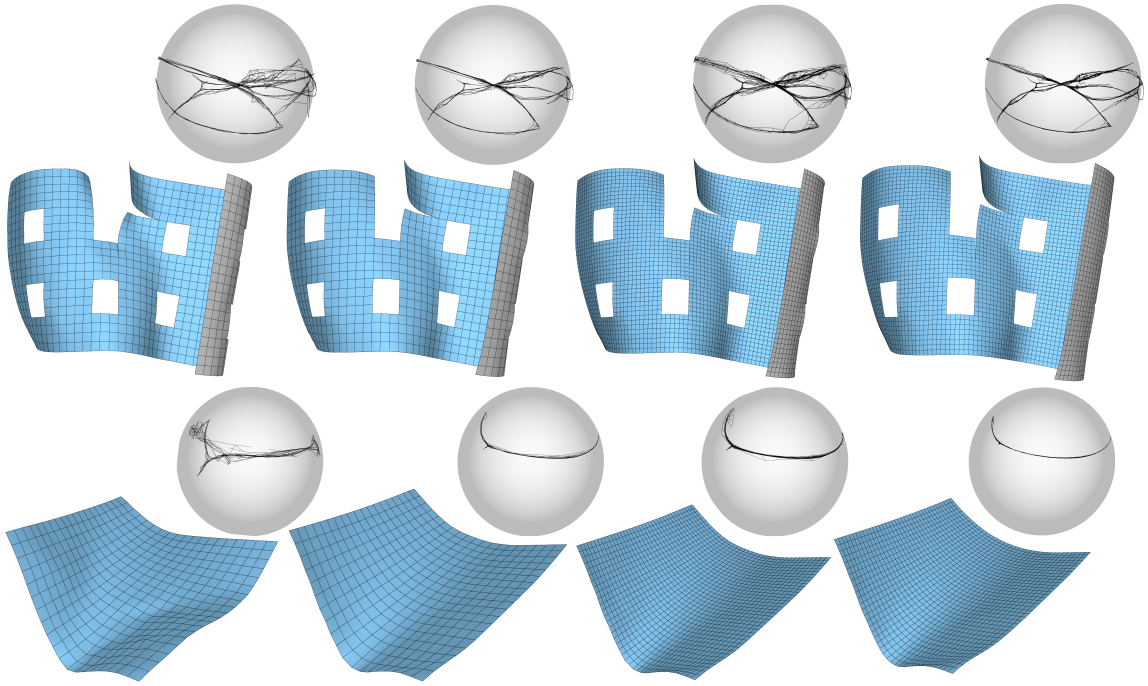


Figure 5.15: *DOG nets and their discrete Gauss maps. The image of the Gauss map of a smooth orthogonal geodesic net is locally a point or a curve, and Gauss maps of smoother DOG nets are closer to being one dimensional, although this is not directly optimized for. From left to right: An input DOG net; 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.*

Curve-constraining flow

Our framework offers an interesting variant on deformations with positional constraints: flows induced by constraining whole curves on the surface. 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 $F_1 - F$:

$$\kappa = \frac{2\sin(\beta_1)}{\|F_1 - F_{\bar{1}}\|}, \quad \tau = \frac{\sin(\theta)}{\|F_1 - F\|}, \tag{5.10}$$

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. 5.16).

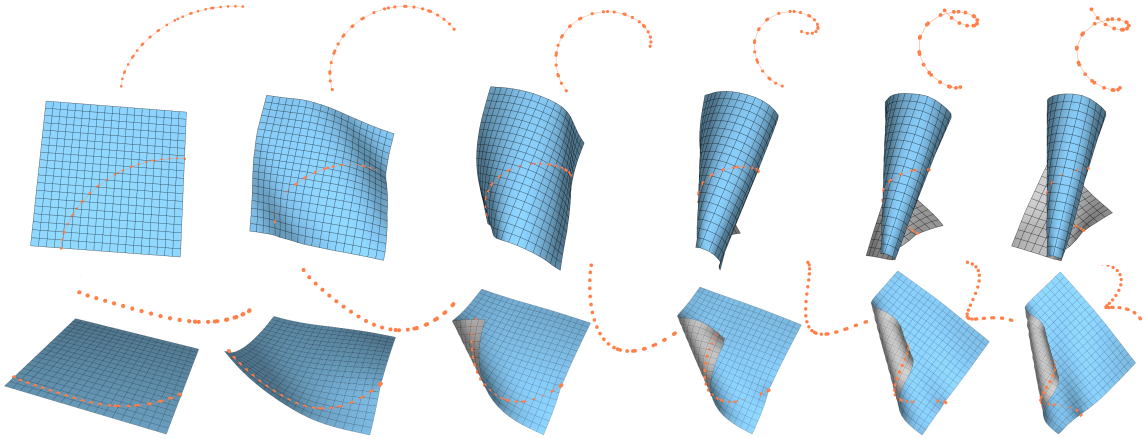
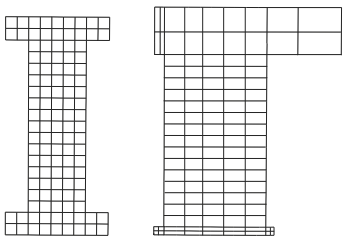


Figure 5.16: *Curve-constraining flows induced by fixing whole curves on the surface using positional constraints. We first represent a curve intrinsically as a set of points on edges, then design target curve positions and compute an interpolation between the curve's initial and target state. Finally, 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.*

CHAPTER 6

Isometries on discrete orthogonal geodesic nets

So far we have defined a model for discrete developable surfaces, but we have not touched upon the subject of their *discrete isometries*. The concept of isometry lies at the core of the study of surfaces, but is notoriously hard to discretize, even for planar surfaces, as current models suffer from locking (see Sec. 2.1 or [Chapelle and Bathe 1998; Alessio 2012]). In this section we will devise two approaches for modeling isometries of discrete planar surfaces based on smooth orthogonal geodesic nets.



DOGs can describe a variety of surfaces with different scales, shapes and lengths (see inset for two orthogonal geodesic nets with the same connectivity). Here, we are looking for a definition of discrete isometry that specifies when two nets are "the same" in a precise manner. Two smooth surfaces S_1, S_2 are said to be isometric, denoted $S_1 \cong S_2$, if there exists an isometry map $\phi : S_1 \to S_2$, i.e., a bijective map that preserves distances on the surfaces, or equivalently the lengths of all geodesics.

6.1 Global isometry for disc topology DOGs

In the special case of two developable surfaces with disc topology, one can test whether they are isometric by looking at their boundaries, as justified by the following lemma.

Lemma 34. Let S_1 and S_2 be two smooth developable surfaces with *disc topology* and equal-length boundaries. Let $\gamma_1(s)$, $\gamma_2(s)$ be their closed boundary curves in arc length parameterization and $\kappa_{g1}(s)$, $\kappa_{g2}(s)$ the geodesic curvatures of these curves on S_1 and S_2, respectively. Then $S_1 \cong S_2 \iff \kappa_{g1}(s) = \kappa_{g2}(s)$.

Proof. Every developable surface is *locally* isometric to a planar surface. By [Tang et al. 2016], a simply connected developable surface is (globally) isometric to a planar surface. Hence, disc topology developable surfaces S_1, S_2 are isometric to some planar surfaces $\hat{S}_1$, $\hat{S}_2$. As geodesic curvature is invariant to isometries, the curvatures of the boundary curves of $\hat{S}_1$, $\hat{S}_2$ are $\kappa_{g1}(s), \kappa_{g2}(s)$. By the fundamental theorem of planar curves, the planar boundary curves differ by a rigid motion (meaning that $\hat{S}_1, \hat{S}_2$ are exactly the same planar shape up to rigid motion) if and only if $\kappa_{g1}(s) = \kappa_{g2}(s)$, hence if and only if $S_1 \cong \hat{S}_1 \cong \hat{S}_2 \cong S_2$. □

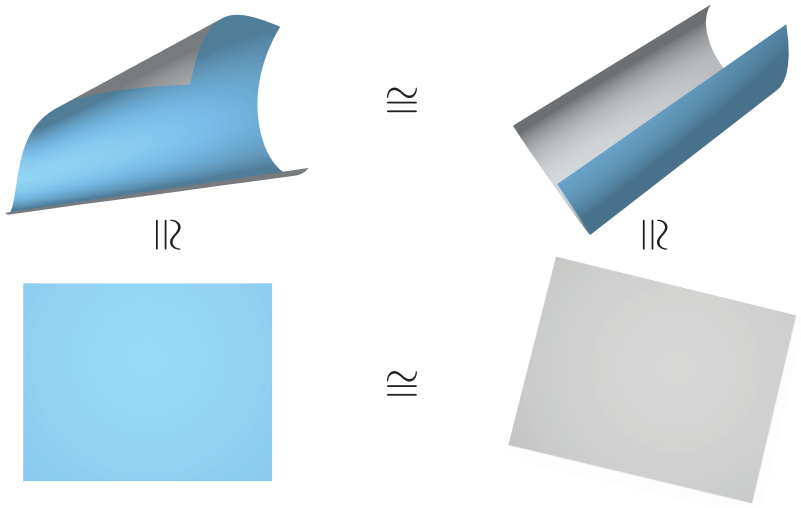
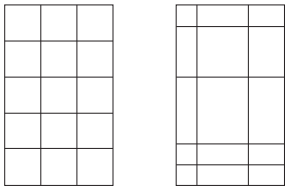


Figure 6.1: *An application of Lemma 34: A developable surface with disc topology and piecewise geodesic boundary with $\frac{\pi}{2}$-corners is isometric to a flat rectangular shape on the plane. Two such surfaces with equal lengths of the boundary pieces are isometric, as their flattened shapes are isometric.*

This lemma can be extended to the case of piecewise geodesic boundary, where the lengths of matching boundary pieces on the two surfaces are equal and the angles of the turns (or "corners") match as well, see Fig. 6.1. This is simple to discretize: two discrete developable nets F_1 and F_2 with disc topology and piecewise geodesic boundaries can be considered isometric if each matching pair of boundary pieces have equal lengths and the matching corners' angles agree.

Such a global definition of isometry cannot be easily generalized to non-disc topologies and it does not provide us with the isometry map in the discrete case. One can easily find a situation where two discrete nets F_1, F_2 with the same con-

nectivity are deemed isometric by the global definition above, but there is no *vertex-to-vertex* map $\Phi : F_1 \to F_2$ that we can reasonably call an isometry. For example, the inset shows a case of two isometric rectangles represented by two different discrete orthogonal geodesic nets, where the discrete mapping Φ that matches corresponding vertices does not preserve any edge lengths. Consequently, a smaller piece $F_1' \subset F_1$ of the first surface is not isometric to the corresponding piece $\Phi(F_1') \subset F_2$ of the second surface. In practical terms, this means that the global criterion is too limited for the purposes of isometric shape modeling, and we need a *local* definition of isometry that tells us when a mapping between two discrete nets is isometric.



6.2 Constraining edge lengths

Defining a notion for a DOG isometry is equivalent for defining a notion of flattening of DOGs to planar DOGs. Indeed two DOGs F^1, F^2 could be considered isometric if their corresponded isometrically flattened DOGs F^1_P, F^2_P have corresponding equal edge lengths. As discussed in Sec. 3.5, a DOG star can be isometrically flattened, but a DOG net generally cannot be. Furthermore, constraining edge lengths on a DOG to be unit length will allow us to only model a very small subset of developables (see Fig. 6.2).

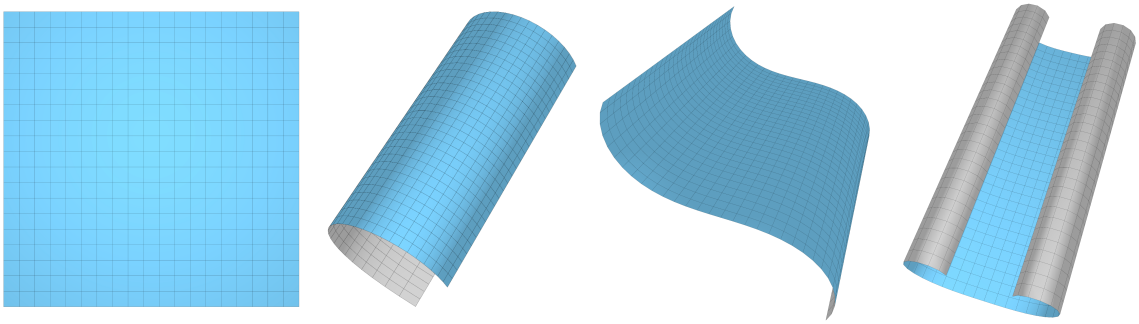


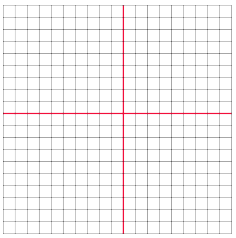
Figure 6.2: *DOGs with unit length edges corresponds to smooth developables parameterized by a family of orthogonal geodesics with the same torsion along the entire net, such as planar and cylindrical nets (see Theorem 13 and Sec. 4.4).*

This "failure" in flattening corresponds to the fact that a general DOG is not a discrete Chebyshev net, though a smooth orthogonal geodesic net is (Sec. 3.5). In practice DOGs representing smooth shapes are very close to being Chebyshev, and the denser the DOG representing a smooth shape is, the closer we could expect it to being numerically Chebyshev.

The surface extension analysis in Sec. 5.1.1 also implies that we cannot add many

edge constraints to our net. Fig. 5.3 depicts how the cone-ray intersection given by constraints propagates and determines a whole quad strip, leaving us solely one edge length and one angle per vertical strip as degrees of freedom, after specifying the coordinates of a single horizontal curve.

The fact that we can only exactly constrain the edge lengths of a single vertical and a single horizontal curve is not surprising, as on a discrete Chebyshev net, made of parallelograms as quadrilaterals, this is equivalent to constraining all edge lengths (see inset);



By the rigidity analysis of Sec. 5.1.1, one can constrain such a set of edges on a DOG without over-constraining the net: the shape space of a rectangular patch of $l \dots r$ vertices of a generic DOG $F \in \mathcal{M}_G$ is a smooth manifold of dimension $3(l + r) - 4$. As smooth DOGs are numerically Chebyshev, modeling smooth shapes with these constraints results in shapes that are numerically isometric, but not exactly. This minor flexibility is required to prevent locking into a partial set of developables (see Fig. 6.2).

In the next section we will look at a model that is similar to DOGs, but allows for an *exact* notion of local isometry, not only at the smooth limit.

6.3 4Q orthogonal geodesic nets

In the previous section we saw how defining a local notion of per-quad isometry on DOGs, for instance by constraining the length of each quad to unit length, results in an over-constrained model that is prone to locking. Here we expand our notion of local neighborhood on discrete nets and loosen the developable net definition somewhat. We define a new class of nets called *4Q orthogonal geodesic nets*, composed of *4Q orthogonal patches*, defined as follows:

Definition 11. An orthogonal 4Q patch is a composition of four quads (see Fig. 6.3), such that:

1. Odd vertices have discrete orthogonal geodesic stars (Def. 1);
2. Even vertices have discrete geodesic stars (Def. 2);
3. The lengths of opposing sides (each a sum of two edges) of the 4Q patch are equal.

Conditions (1) and (2) imply that an orthogonal 4Q patch can be seen as discrete developable, since its boundary can be interpreted as a set of four geodesic curves intersecting orthogonally, resulting in a vanishing integrated Gaussian curvature in

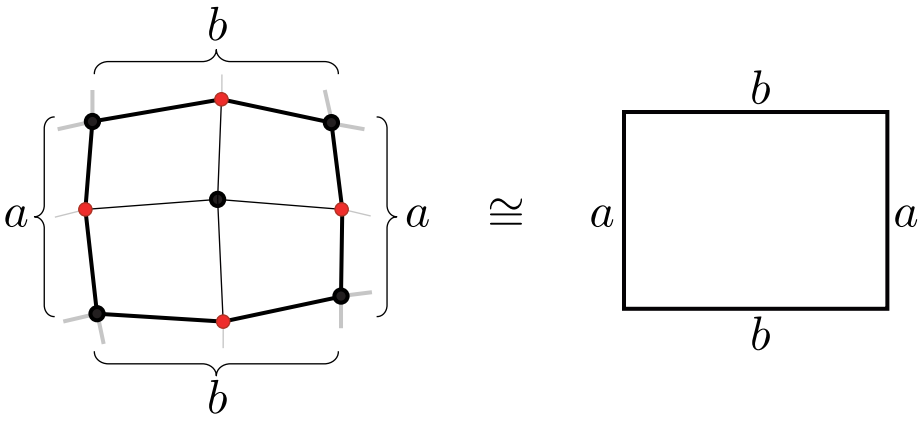


Figure 6.3: *An orthogonal 4Q patch. Odd (black) vertices are discrete* orthogonal *geodesic vertices, even (red) are discrete geodesic vertices. The lengths of opposing sides of the 4Q patch are equal. An orthogonal 4Q patch is seen as isometric to a rectangle in the plane with the same side lengths.*

the interior of the patch. Condition (3) implies that the 4Q patch can be seen as isometric to a rectangle, in the sense of the extension of Lemma 34 discussed above. In the same spirit, we can model (global) isometries of the 4Q patch by requiring the conservation of the lengths of its sides.

An orthogonal 4Q geodesic net F is a discrete net composed of orthogonal 4Q patches. Two orthogonal 4Q geodesic nets are isometric if there exists a one-to-one correspondence between their 4Q patches, such that for each pair of matching patches, the corresponding side lengths are equal. Modeling isometric deformations on an orthogonal 4Q net amounts to keeping these lengths fixed, enabling us to model isometries on a wide range of surfaces, unconstrained by their topology.

6.3.1 Rigidity analysis

Here we analyze the rigidity of orthogonal 4Q nets by looking at the construction of a 4Q net from a single strip, similarly to the analysis of orthogonal geodesic nets in Sec. 5.1.1. We observe that orthogonal 4Q nets have a similar rigid structure, which implies that while these nets do offer us additional degrees of freedom to incorporate local length constraints, they are not too permissive and still reasonably represent the space of developable surfaces.

Analogously to our analysis in Sec. 5.1.1, we show how orthogonal 4Q net constraints propagate from a given horizontal strip, leaving only a few degrees of freedom, and in practice, for nets representing smooth shapes, almost none. Recall that we denote by black vertices the centers of discrete orthogonal geodesic stars, while red vertices are centers of discrete geodesic stars that are not necessarily orthogonal; opposite sums

of edges in every 4Q quad are equal. We start by noting that a vertex and three of its neighbors can be generally completed to a geodesic star by a point located on a unique ray (Fig. 6.4), and we refer to this as *direction propagation*; this is analogous to the plane reflection Lemma 19 that refers to the special case of orthogonal geodesic stars, and is a direct result of Lemma 5. We analyze the most constrained case, where one 4Q quad is already given that extends our horizontal strip. This is similar to the choice of one edge length and angle for discrete orthogonal geodesic nets in Sec. 5.1.1, but with a few more degrees of freedom (Fig. 6.4).

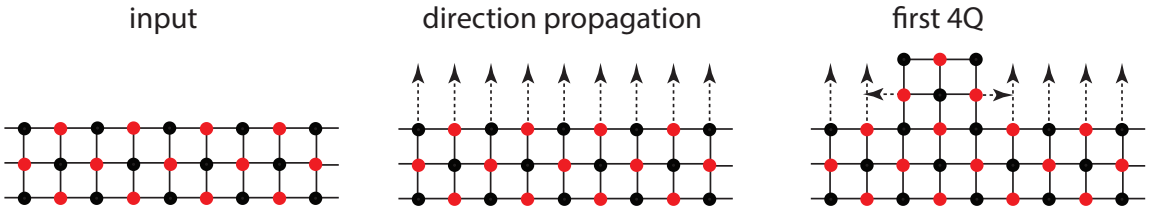


Figure 6.4: *Left: A given 4Q strip. Center: By direction propagation, any vertex with its three neighbors can be generally completed to a geodesic star by a point on a unique ray. Right: The first extension 4Q quad must be such that the horizontal rays emanating from its middle, determined by the direction propagation, intersect the two neighboring vertical rays emanating from the strip, such that valid vertices can be formed at the intersection points.*

By direction propagation, this first extension 4Q quad must be such that the two horizontal rays emanating from its middle intersect the two neighboring vertical rays from the strip (Fig. 6.4), so that valid vertices can be formed at the intersection points.

We continue observing how the entire strip propagates by the orthogonal 4Q geodesic net constraints. We refer the reader to Fig. 6.5, where we note that by the previous constraint on the first extending 4Q quad, two rays intersect at a new vertex. The rest of the figure shows repeated application of Lemma 20, a sequence of cone-ray intersections. Note that this lemma is also valid when only one of the vertices is a geodesic, as evident in its proof.

The cone intersection propagation determines all vertices of a neighboring 4Q quad but one. This vertex must fulfill two conditions:

1. The sums of edge lengths of the opposing vertical sides of the 4Q quad are equal;
2. The sums of edge lengths of the opposing horizontal sides of the 4Q quad are equal.

As all edges except one are already determined, this means that the missing vertex should lie in a fixed distance from two different points, or equivalently on an

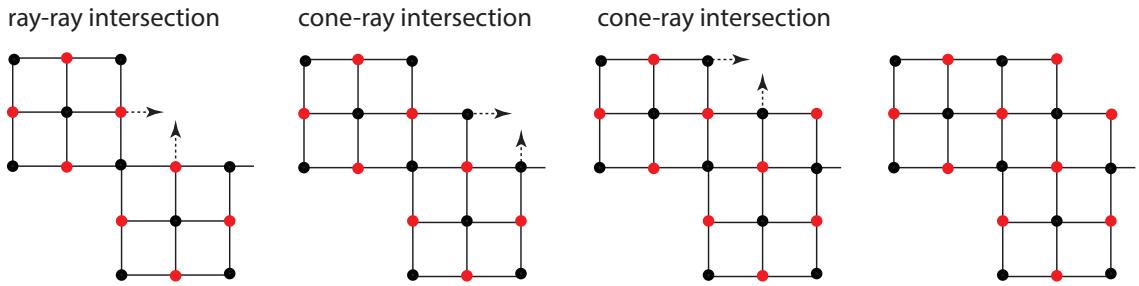


Figure 6.5: *A ray-ray intersection and repeated application of Lemma 20 determines the location of all but one vertices of a neighboring 4Q quad.*

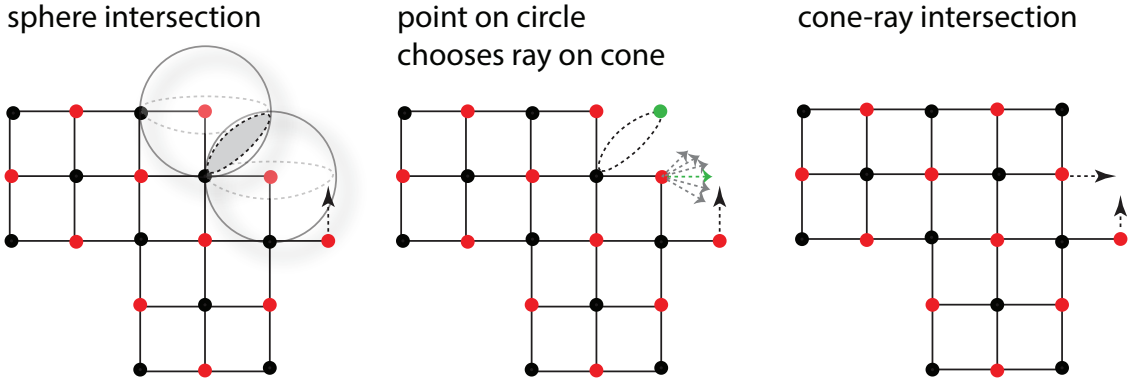


Figure 6.6: *Left: The upper right corner vertex lies on an intersection of two spheres. Center: These spheres generally intersect in a circle. Every point on this circle determines a unique ray by direction propagation, and all these directions together form a cone. Right: This cone intersects with a given vertical ray, and the upper right corner vertex is a point on a circle that propagates the direction of the intersecting ray on the cone.*

intersection of two spheres (Fig. 6.6). If the spheres intersect, they either intersect in a point or a circle; in practice for a smooth enough net, this generally results in a circle. Every point on this circle satisfies the length constraint, but does not in general create a direction that intersects with a given vertical direction for the net. The set of all of these directions generates a cone, and so the last 4Q vertex lies on the intersection of this cone with a given vertical ray (see Fig. 6.6). This process repeats to reveal the entire extension strip, as the next vertex of a neighboring 4Q quad is given by a ray-ray intersection.

6.3.2 Results

The length constraints (3) in Def. 11 can be written as the difference of the sums of the respective edge lengths. Incorporating these constraints allows us to isometrically

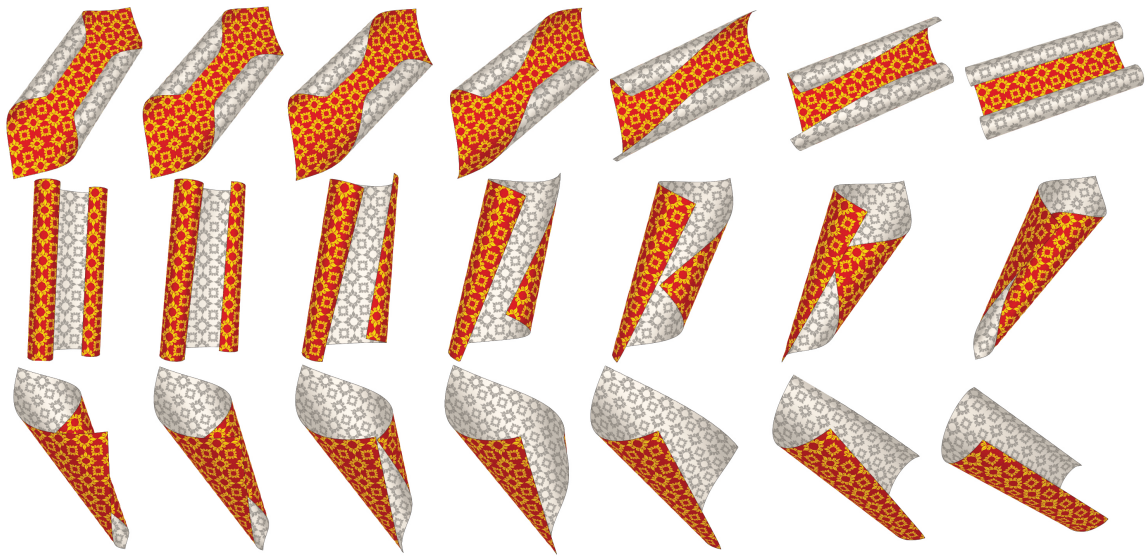


Figure 6.7: *An interpolation sequence of isometric orthogonal 4Q nets. Note that in all cases, the deformations alter the rulings directions and their combinatorial structure. See the accompanying video at [Rabinovich et al. 2018a] for the entire sequence.*

edit orthogonal 4Q nets. Fig. 6.8 demonstrates an editing operation that includes bending, gluing and cutting of a strip, all done while maintaining the orthogonal 4Q patches isometric to the reference state.

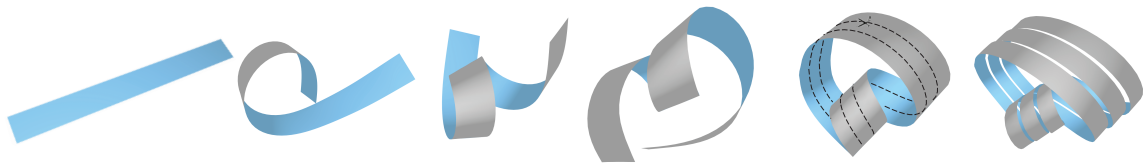


Figure 6.8: *Our local isometry model allows us to deform, glue, and cut a surface while maintaining an exact discrete notion of local isometry. In this figure we twist a long strip twice, glue it, and then cut it along two horizontal sections, creating three interleaved knotted surfaces.*

As an experiment showing empirically the flexibility of the 4Q model, we show that our constraints can be used in combination with a shape interpolation algorithm such as [Lipman et al. 2005; Fröhlich and Botsch 2011]. In Fig. 6.7 we compute a sequence of isometric shapes, morphing a source shape into an (isometric) target, thereby simulating isometric bending of developable surfaces that generally happens *not* along their rulings. An initial guess for each interpolation frame is first computed with [Fröhlich and Botsch 2011], followed by the optimization of (i.e., projection onto) our constraints, similar to the one at Sec. 3.6.

6.4 Discussion

We have discussed two models for modeling isometries, based on smooth orthogonal geodesic nets, both complemented with a theoretical foundation showing that they do not suffer from locking. These are, as far as the authors know, the only discrete models with this property.

The first model (Sec. 6.2) is based on discrete orthogonal geodesic nets, on the rigidity analysis at Sec. 5.1.1 and on the fact that smooth orthogonal geodesic nets are Chebyshev (Sec. 3.5). This model requires choosing of an arbitrary edge length to constrain for every row or column, though by the Chebyshev property of smooth orthogonal geodesic nets this choice should have very little affect. This arbitrary choice however, have deterred the author of this thesis to use it in practice, and the isometric modeling at the paper [Rabinovich et al. 2019] is motivated by this model, but instead use soft constraints using a penalty on all edge lengths. This removes the choice of an arbitrary edge per row/column and is simpler to implement, but is theoretically less sound. This seems to work well in practice, requiring a reasonable penalty parameter and allow for interactive modeling of isometries with numerical very little stress and no locking (see Sec. 7.6 for implementation details). Unlike works on thin shell simulations [Burgoon et al. 2006], the exactly enforced DOG constraints here imply that a minor stretch, at worst, means we model a slightly stretched *developable surface*, rather than a doubly curved surface. We do note however that it might be useful, in terms of speed as well as quality, to implement the exact isometry model here rather than our penalized version.

The second model, $4Q$ orthogonal geodesic nets, is a model for smooth orthogonal geodesic nets that is an alternative to DOGs, and unlike DOGs has a Chebyshev-like property but at the price of a model that is more complicated and slightly less local. We believe that models enforcing local constraints in a slightly bigger neighbourhood, could be used to discretize other geometries, and to prevent locking. The key to devise them is understanding the constraints via analysis similar to Sec. 5.1.1 and Sec. 6.3.1.

CHAPTER 7

Freeform modeling of curved folded surfaces with discrete orthogonal geodesic nets

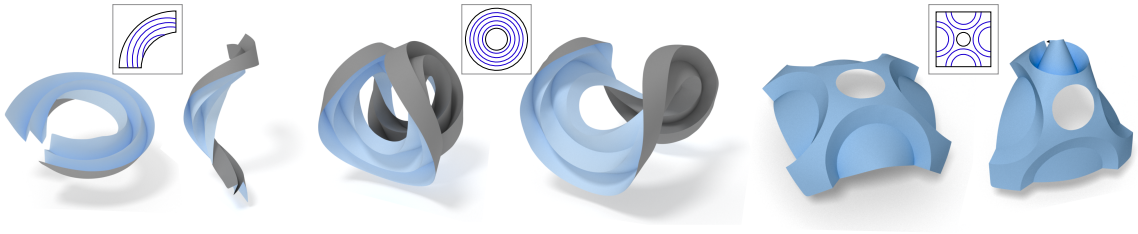


Figure 7.1: *Curved folded surfaces modeled with our method, along with their crease patterns. Our deformation algorithm is able to simultaneously bend and fold complicated crease patterns using only positional constraints, while automatically finding a valid mountain/valley assignment along the creases. Our framework is suitable for freeform editing and exploration of new curved folded surfaces.*

There are infinitely many ways to deform a planar sheet without stretching or tearing it. One can either bend it, form sharp creases by folding it, or combine the two. Folding and bending isometries are different by nature, and historically there has been a dichotomy in the study of the two. Smooth bending deformations are typically studied in differential geometry [do Carmo 1976], whereas straight folds are explored in the field of computational origami [Demaine and O'Rourke 2007]. Curved folded surfaces [Huffman 1976] (Fig. 7.1) can be viewed as a combination of the two, since folding an inextensible sheet along a curve necessitates global bending around the

crease. These elegant geometries have garnered the attention of architects, artists, and industrial designers [Pottmann et al. 2015; Tachi 2013, 2011; Buri et al. 2011; Gramazio and Kohler 2014; Demaine et al. 2011c].

The design of a curved folded surface is manual and time consuming and is usually done using an empirical trial and error approach [Demaine et al. 2011c,b]. The known theory on curved folds is confined to a narrow set of folds, and contrary to classical origami, bending and folding instructions are hard to write down and multiple creases must be folded simultaneously [Kilian et al. 2017]. Artists generally pre-crease the paper using a ball burnisher or a CNC plotter before carefully folding and bending, making the process of shape exploration even slower.

Although manual and slow, playing with paper is still the predominant approach for curved folded surface design. Existing works on modeling such surfaces are either limited to previously discovered surfaces [Kilian et al. 2008, 2017] or model a small, partial set of folded surfaces generated by reflections or rotational sweeps [Mitani and Igarashi 2011; Mitani 2009]. Modeling the folding process of novel forms remains a challenge [Demaine et al. 2011c].

In this chapter we set out to develop the basic tools for freeform modeling of curved folds, with the objective of aiding the exploration, analysis and study of new curved folded surfaces. Our work builds upon discrete orthogonal geodesic nets (DOGs) as a discrete model for C^2 developable surfaces. DOGs do not suffer from locking of various deformation modes [Chapelle and Bathe 1998; Alessio 2012; Grinspun et al. 2003], are not limited by an initial choice of meshing or rulings [Tang et al. 2016; Kilian et al. 2008; Stein et al. 2018; Solomon et al. 2012] (see Fig. 7.2) and do not require remeshing while deforming the surface [Kilian et al. 2017; Schreck et al. 2017; Narain et al. 2013]. Therefore DOGs are particularly suited for modeling curved folds.

The result of this chapter is a computational framework for interactive design and exploration of curved folded surfaces. We first show how to model curved folds as a collection of DOGs, and then devise an algorithm to simultaneously fold creases and smoothly bend planar sheets. We complement our algorithm with essential building blocks for curved folding deformations: objectives to control dihedral angles and mountain-valley assignments. At the end of this chapter, we apply our machinery to build the first interactive freeform editing tool capable of modeling bending and folding of complicated crease patterns.

7.1 Related work

Curved folded sculptures are beautiful works of art almost a hundred years old, dating back to the 1920's works of Josef Albers in the Bauhaus art school [Adler 2004], and

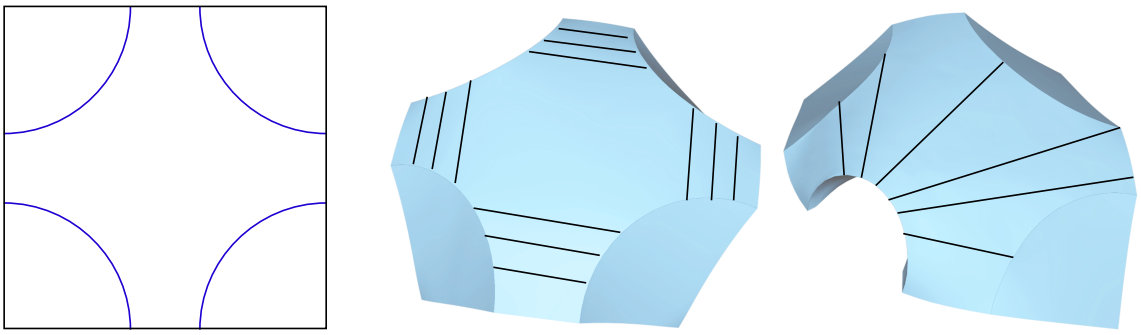


Figure 7.2: *Folding and bending the same curved crease pattern (left) into two different surfaces, with some of their rulings plotted. Methods that model the rulings explicitly must remesh in order to model these surfaces, since the rulings can change drastically, connecting vertices of different patches to one another. Representing the curved folded surface as a piecewise DOG avoids the need to remesh because a DOG is parameterized by intrinsic invariants – orthogonal geodesics – and does not explicitly encode the developable rulings in its mesh.*

continuing with the investigations of David Huffman and Ron Resch in the 1970's [Huffman 1976; Resch 1974]. This direction in art, though intimately linked to the mathematics of developable surfaces, is mostly driven by physical experiments with paper [Demaine et al. 2011c]. As opposed to smooth developable surfaces [do Carmo 1976] or straight fold origami [Demaine and O'Rourke 2007], the mathematics of curved folding is lagging behind the manual craft and mostly concerns the local behavior of a single folded curved crease [Duncan and Duncan 1982a; Fuchs and Tabachnikov 2007; Demaine et al. 2011c]. Notable crease patterns, such as the Huffmann Tower [Wertheim 2004; Demaine et al. 2011b], are not yet understood [Demaine et al. 2018] and the known mathematics on the folding and bending of multiple curved creases is limited to few particular cases, coupled with a specific folding movement and guided by fixed rulings [Demaine et al. 2015, 2018; Mundilova 2019]. In essence, we do not know much about which crease patterns can fold, and we do not know in which ways they can fold. Unlike straight origami, there are often infinitely many ways of folding and bending a curved crease pattern by varying the dihedral angles as well as the developable rulings along the different creases.

Curved folding was introduced to the geometry processing community by the work of Kilian and colleagues [Kilian et al. 2008], where the authors devised an algorithm to reconstruct scanned paper curved folded surfaces by estimating their ruling directions, resulting in a mesh with a fixed torsal/planar patch decomposition and mountain/valley assignments. Our work directly deals with the difficulty of folding starting from a flat configuration (Fig. 7.10) and our folding characterization and algorithm (Sec. 7.4 can also be applied to the model of [Kilian et al. 2008], possibly combined with remeshing to accommodate the locking issue (Fig. 7.2).

Several works on modeling curved folded surfaces focus on a given subset of folding deformations, such as planar creases generated by reflection [Mitani 2012; Mitani and Igarashi 2011], surfaces generated by rotational sweeps [Mitani 2009] or surface folded with a fixed ruling pattern [Tang et al. 2016]. In [Kilian et al. 2017] the authors simplify the process of fabrication for a wide range of curved folded models using a network of strings, solving the problem of which surface points to pull in order to actuate a folding movement. To model the folding deformation the authors of [Kilian et al. 2017] employ the model of [Botsch et al. 2006] with the remeshing algorithm in [Narain et al. 2012]. Their deformation is guided by mountain/valley assignments of all creases, which are provided as input, as well as prescribed soft constraints on folding angles, as well as a bending objective. We note that prescribing dihedral angles on a single curve results in an undetermined system, while prescribing the folding angles of multiple creases often results in an overdetermined system. There are infinitely many ways to fold a surface with the same prescribed folding angles [Fuchs and Tabachnikov 1999; Duncan and Duncan 1982a], however dihedral angles across multiple folds must be compatible with each other [Demaine et al. 2018].

The tools presented in this chapter are applied to develop the first freeform handle based system for curved folding deformations. Our point handle based editing system ensures folding of the creases of a given pattern rather than smoothly ignoring them, all without requiring the user to specify dihedral angles or mountain/valley assignments (see Fig. 7.10). In cases where more explicit control is desired, we also derive simple quadratic constraints to control dihedral angles along folds, as well as mountain/valley assignments – a degree of freedom that is often only available along one curved crease (see Fig. 7.17 and Fig. 7.18).

7.2 Setup

7.2.1 Definitions

Throughout the chapter, we use the following definition for a curved folded surface:

Definition 11. A surface S is called a curved folded surface if it is locally isometric to the plane and can be written as a finite union $S = \bigcup S_i$ where each S_i is a C^2 developable surface termed a *patch*, and the intersections of different patches $S_i \cap S_j$ are either empty or are C^2 curves.

By this definition, a smooth developable surface is a curved folded surface with a single patch. The definition is suitable for various topologies, such as a cylinder, but throughout this chapter we work with surfaces that are isometric to a subset of $\mathbb{R}^2$. Borrowing terms from [Demaine and O'Rourke 2007; Demaine et al. 2011a], we often refer to the surface *crease pattern* as the planar domain P isometric to S, subdivided

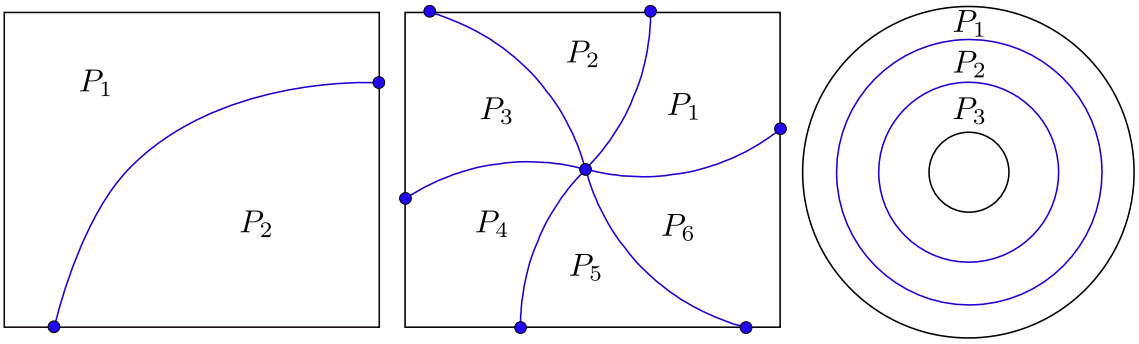


Figure 7.3: *Curved crease patterns, decomposing a pattern into multiple components P_i and intersecting at crease vertices. Boundary curves in black, crease curves in blue.*

into patches P_i (flattened S_i), whose intersection curves are the flattened creases (see Fig. 7.3). Flattened creases with nonzero curvature are said to be curved, while those with vanishing curvature are straight. A crease might be partly curved and partly straight, or curved almost everywhere but with inflection points where the curvature vanishes. The flattened domain boundaries together with the flattened crease curves form a planar arrangement [Grünbaum 1972], inducing a planar graph that decomposes P into the planar faces P_i. The vertices of this graph are the intersection points of the curves with each other or the boundary curves, which we call *crease vertices*. The *edges* of this graph are the pairwise intersections of the various patches, and we refer to the inner points of these curves as *crease points*, i.e., the points on these curves that are not *crease vertices*. We say that S is *folded* at a crease point p if at that point the patches S_i, S_j sharing it have a tangential discontinuity.

7.2.2 Desiderata

Our goal is to develop tools for the exploration of curved folded shapes on top of piecewise DOGs by means of deformations. Our choices are guided by the two following ground rules for deforming DOGs: (1) Perform homotopy based optimization, and (2) Minimally constraining the DOGs.

Homotopy based optimization is motivated both theoretically and empirically: Modeling DOGs requires solving highly constrained and nonlinear optimization problems, yet the theory of DOGs guarantees the existence of nearby solutions if one starts at a feasible point. As seen in Chapter 5, generally the shape space of DOGs is a smooth manifold. This observation is useful in practice: DOGs exploration performs well using smooth flows or homotopy based optimization methods both for

handle based deformation tasks as well as more complicated deformations such as curve-constraining flows as in Sec. 5.2.

Figure 7.4: *Homotopy based optimization for curved folding modeling. A discretization of a* curve-constraining flow *on curved crease surfaces, composed of two DOG patches connected by boundary constraints. This folding and bending flow is smooth, when viewed on the patches separately. connected along used in the above animation.*

Minimally constraining the DOGs. Since DOGs are already heavily constrained objects, one needs to carefully choose which quantities to fix by hard constraints, and which to optimize using soft constraints. This is essential in order to avoid locking or ill-posed problems in case the constraint gradients are linearly independent Sec. 5.2. In particular, the rigidity analysis in Sec. 5.1 demonstrates that one cannot fix all edge lengths, or likewise demand a DOG to also be a Chebyshev net (Sec. 3.5). We note however that this can be done approximately and to a low tolerance, because a smooth orthogonal geodesic net is Chebyshev, and it admits a rich set of exact isometries. Our model in Sec. 7.3 is a minimal model for curved folded surfaces, modeling intersecting developable surfaces that can be flattened along their intersection as well. Our folding constraints in Sec. 7.4 are chosen such that they can be satisfied *exactly*. They capture an important characteristic of curved folded surfaces: a folded crease point remains folded under small deformations.

7.3 A model for curved folded surfaces

Up to this point our discussion has been limited to discretizing smooth deformations (Sec. 5.2) on developable surfaces, minimizing smooth objective functions defined at Chapter 4. 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. Following Def. 11, we would like to extend the set of deformations to model *piecewise* smooth developable surfaces, allowing tangent discontinuities along arbitrary curves on the surface.

7.3.1 Crease curves

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. 7.5, left).

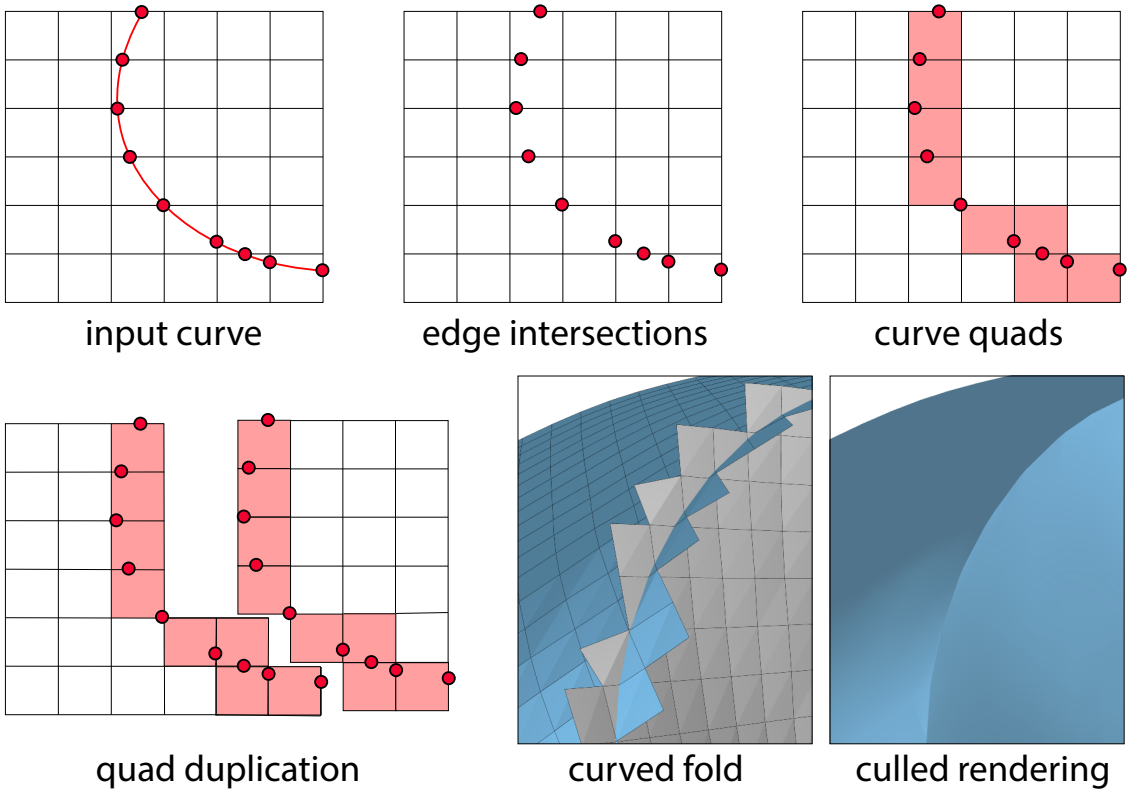


Figure 7.5: *Crease curves on DOGs. A curve on the surface is represented by its intersections with mesh edges. To model a sharp crease, we duplicate the faces intersected by the curve and split the mesh, adding C^0 continuity constraints. For curved folds, we also incorporate flattenability conditions along the curve. For rendering purposes, extraneous parts of the duplicated faces are culled.*

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 $0 \leq t \leq 1$ and i_1, i_2 are the indices of the edge vertices.

7.3.2 Discrete curved folded surfaces

We represent a curved folded model as a collection of DOGs surfaces, denoted by S_i, satisfying a set of constraints on their shared boundaries. Our input is an arrangement of curves representing our crease pattern. On top of this arrangement we place an orthogonal grid while ensuring that every vertex of the arrangement lies on a grid line. We then split the grid into overlapping patches, sharing the faces where curves pass. Finally, we compute the intersections of the curves with the grid edges and represent the resulting curve points as linear combinations, one for each patch sharing that curve point. See Fig. 7.6 for an illustration.

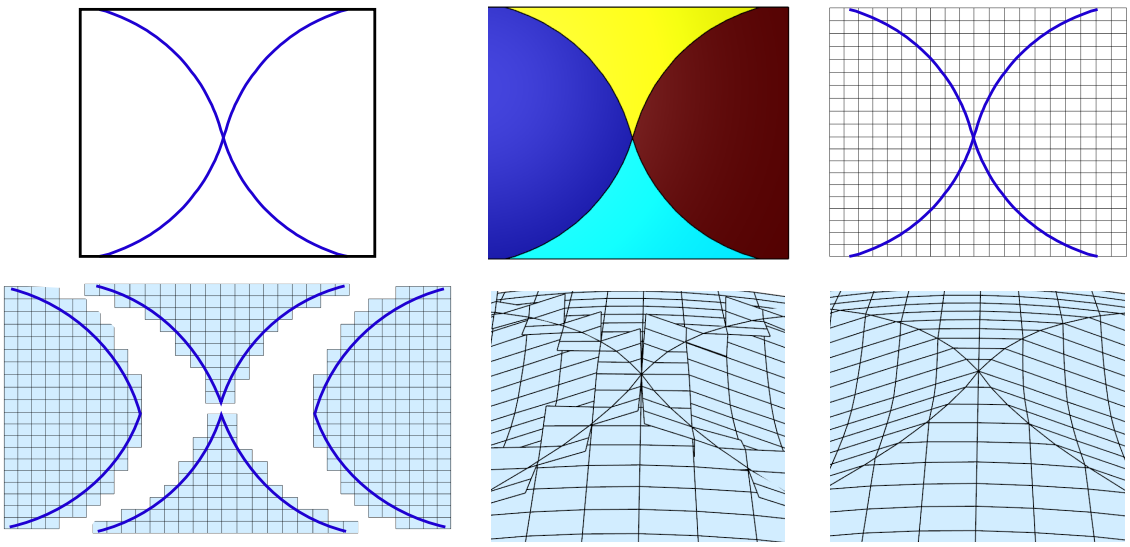


Figure 7.6: *Representation of a discrete curved folded surface. Top left: A given curve arrangement, representing the domain via its boundary (in black), and two curves intersecting at their inflection point in the center of the domain (in blue). Top center: The curves segment the domain into four patches, each intersecting with two other patches along a segment of a crease curve. Top right: Placing an orthogonal grid on top of the crease pattern. Bottom left: We model the surface by a separate DOG for each patch, where faces intersecting the crease curves are duplicated for the different patches. Each pair of intersecting DOG patches satisfies continuity constraints. Bottom center: A closeup on a deformation of the planar model. Bottom right: Culling the extraneous parts of the duplicated faces.*

We maintain *continuity* of curve points along edges while penalizing deviation of the edge lengths across duplicated quads. The first constraint corresponds means the surfaces S_i indeed intersect, while the latter corresponds to making sure they can be flattened to single planar patch along their creases. In this chapter we enforce flattenability across patches by strictly modeling isometries of the DOGs from a reference planar surface, where the grids are aligned. However, one can also allow for non-isometric modeling of curved folded surfaces (as in [Rabinovich et al. 2018b]) by enforcing equal edge lengths across quadrilaterals, without any direct reference to an isometric planar shape.

This model can be directly plugged in an editing system (Fig. 7.7), or used in a variety of design tasks with curved and straight creases (Fig. 7.4, Fig. 7.8, Fig. 7.9).

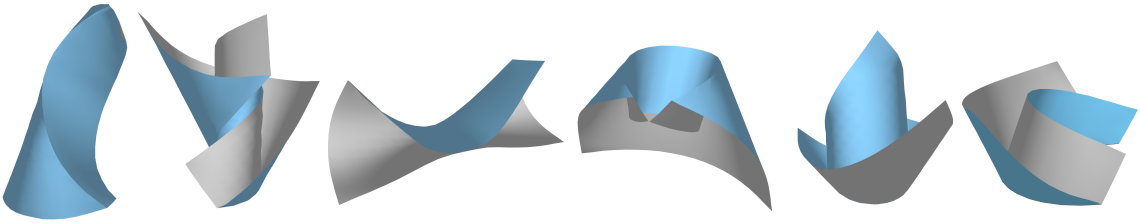


Figure 7.7: *Curved folded surfaces created using our point-handle based editing system.*

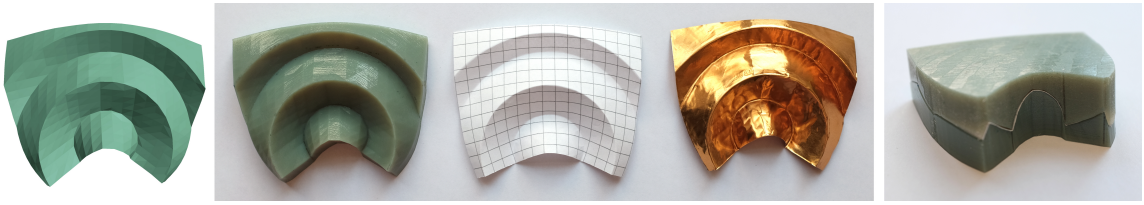


Figure 7.8: *Design and fabrication of a surface with curved folds (four concentric circles). From left to right: the mesh, triangulated for rendering and 3D printing purposes; 3D printed "mold"; rectangular sheets of paper and foil, manually formed by the mold; fixating the shape of the paper in a 3D 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.*

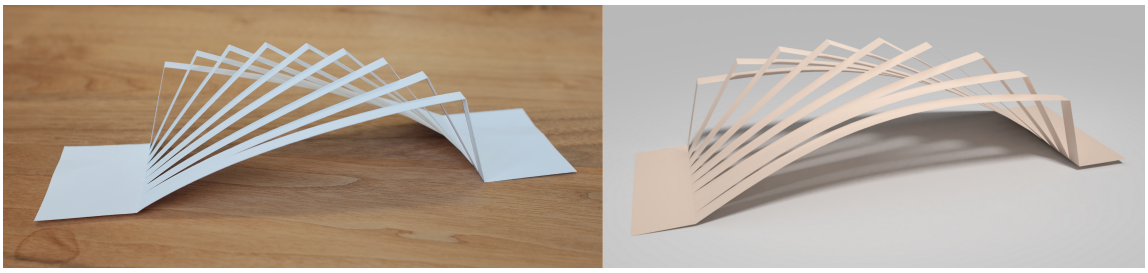


Figure 7.9: *A kirigami surface folded from a planar sheet after applying multiple cuts and straight creases. On the left we show our physical mockup made of thick paper, and on the right is a physically based rendering of the corresponding DOG mesh designed in our system.*

7.4 Folding crease patterns

In practice, deforming the model at Sec. 7.3 does not usually result in a model that is folded along all creases (see Fig. 7.10), but instead in a surface that is smooth along most of the creases.

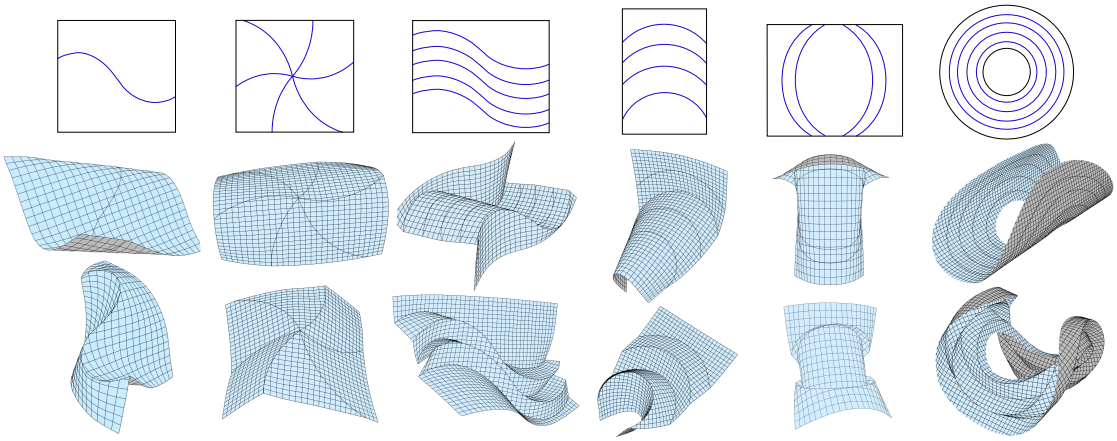


Figure 7.10: *Comparison of the same deformation objective with and without our folding algorithm. Up: Crease patterns given as input. Center: Applying a positional based deformation objective of the crease patterns without our folding algorithm results in most crease curves being ignored, i.e. not folded. At this stage, one cannot bend these creases without first flattening the surface. Down: Result of applying the same deformation with our folding algorithm. Our folding algorithm simultaneously folds all crease points while deforming a surface by adding a bias that effectively push flat points towards a folded configuration while not affecting already folded points. The crease mountain/valley assignments, which in these examples are in fact fixed given one choice, are determined automatically without any input. The objective of all of these deformations were the same positional constraints defined on mesh vertices or on a part of a crease curve based on a curve-constraining flow.*

Part of the difficulty of modeling these deformations stems from the need to fold all curves simultaneously starting from a flat configuration. The primary goal of this chapter is to deal with this difficulty. We start by exploring the different ways one can fold a given crease pattern, and then derive a discrete combinatorial characterization for the local existence of a fold in a piecewise DOG. This characterization will be instrumental to our folding optimization algorithm (Sec. 7.6).

7.4.1 The smooth and combinatorial degrees of freedom around a single curved crease

Straight creases are rather boring, mathematically speaking. Straight lines can only be folded as in classical origami, i.e., by keeping them straight [Demaine et al. 2015], unless one first folds a crease by 180 degrees, such that the two incident sheets coincide. Hence a folding of a single straight crease can be described by a single real number representing the constant dihedral angle between the incident planes. There are infinitely many ways, or degrees of freedom, to fold a curved crease. If S is a

surface with a folded crease, and P is its flattened isometric reference, then one can locally deform the curved surface S by freely deforming the crease curve, as long as the absolute value of the crease curvature stays greater than its flattened curvature in P [Fuchs and Tabachnikov 1999]. Up to a rigid motion, a curve is defined by its curvature and torsion functions.

One can flip this point of view: Given a planar domain P, a curve on the domain $\gamma(t)$ and a deformed, isometric space curve $\Gamma(t)$ with greater absolute curvature than that of $\gamma(t)$, there are only two smooth surfaces isometric to P passing through $\Gamma(t)$ such that the unfolding of the surface to the plane maps $\Gamma(t)$ to $\gamma(t)$ [Fuchs and Tabachnikov 2007] (see Fig. 7.11 left and center).

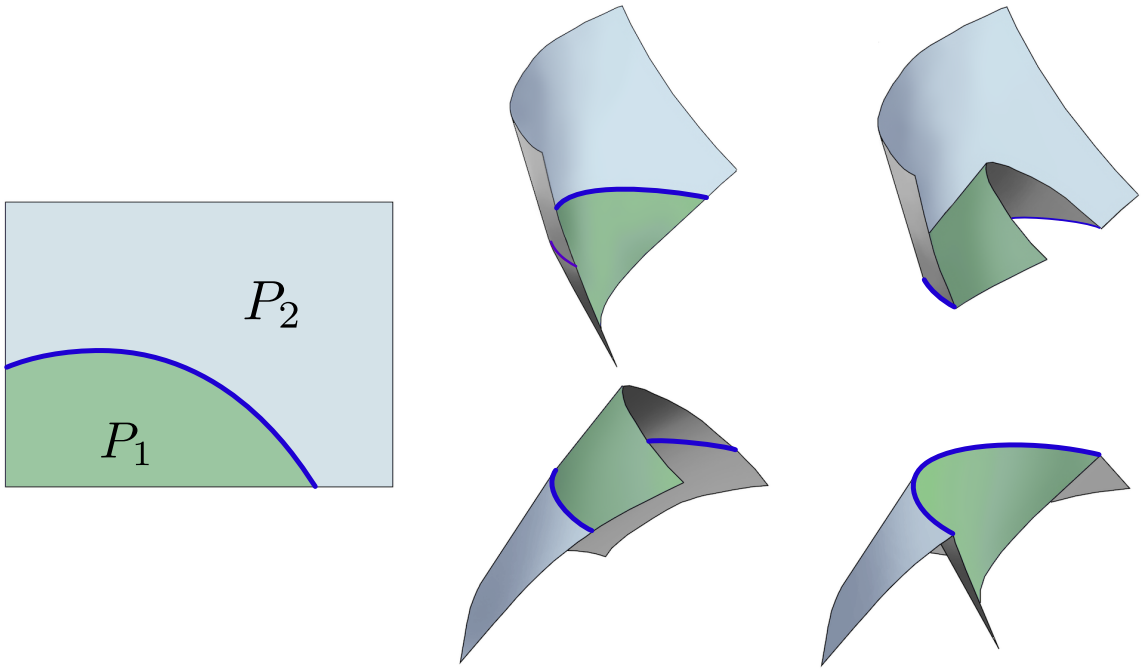


Figure 7.11: *Illustration of the combinatorial degrees of freedom in curved folding. If a crease pattern of a curved folded surface (left) is isometrically folded such that a given curve lies in some configuration in $\mathbb{R}^3$, then there are only two smooth surfaces that isometrically flatten into the crease pattern (center). One can also choose a different surface for each patch P_1, P_2, resulting in two other, curved folded surfaces (right).*

If one permits the surface S to have a fold along $\Gamma(t)$ but remain smooth around it, then there are four possible surfaces: two of them are smooth, while the other two are folded along the curve (see Fig. 7.11). If $S = S_1 \cup S_2$ is folded along $\Gamma(t) = S_1 \cap S_2$ then the angle between the tangent planes of S_1, S_2 along the curve is called the folding angle, which we denote by $\theta(t) > 0$. Unlike the case of a straight crease, $\theta(t)$ often varies along the curve. The bigger the curvature of $\Gamma(t)$, the bigger the folding angle. If $\kappa(t)$ is the curvature of the space curve $\Gamma(t)$, $\kappa_g(t)$ is its geodesic

curvature, which is also the curvature of $\gamma(t)$, then $\kappa_g(t) = \kappa(t)\cos\frac{\theta(t)}{2}$, implying that the osculating plane of the crease $\Gamma(t)$ bisects the tangent planes of the smooth patches intersecting at $\Gamma(t)$ [Kilian et al. 2008; Duncan and Duncan 1982a]. The folding angle does not dictate the shape of the surface, as different surfaces can be generated with the same $\theta(t)$ by varying the torsion of $\Gamma(t)$, thereby changing the ruling pattern of each developable patch. The connection between $\theta(t)$, the curvature and the torsion of $\Gamma(t)$, the curvature of $\gamma(t)$ and the rulings of each incident patch is further detailed in [Demaine et al. 2018].

To summarize, the shape of a curved folded surface S with a single curved crease $\Gamma(t)$ can be locally described by two real functions for the curvature and torsion of $\Gamma(t)$ under the condition of sufficient absolute curvature, as well as an additional combinatorial parameter distinguishing between four possible surfaces, two of which have a fold.

7.4.2 The combinatorial parameters of multiple creases

The previous analysis explains the local behavior of curved folding around a single curve. Understanding crease patterns globally still remains a challenge. In essence, deforming one patch propagates a global deformation of the patch on the other side of the crease, a process that depends on the locations of the creases and the possibly changing ruling lines along the developable. When there are multiple creases, the propagation dictates the shape of other patches. The process becomes more complicated when some creases intersect, due to compatibility constraints (see Fig. 7.12).

Generally speaking, one may be able to choose between four different configurations of the surface at one crease, but this choice already fixes the patch shape for nearby creases. The combinatorial degrees of freedom that remain are whether each crease is folded or not (see Fig. 7.10). The difficulty in modeling folding of a planar surface stems from the fact that these combinatorial choices often need to be enforced at the beginning of the folding process, as explained by the following theorem.

Theorem 35. Let $S(t)$ be a curved folding flow and let $p(t_k)$ be a point on a curved crease of $S(t)$ lying on two patches $S_1(t), S_2(t)$ at a given time in the flow, $t = t_k$. If $p(t_k)$ is not a planar point on $S_1(t_k)$ (or equivalently on $S_2(t_k)$), then there exists an $\epsilon > 0$ such that one of the following holds:

1. $S(t)$ is folded at $p(t)$ for every $t \in (t_k - \epsilon,\ t_k + \epsilon)$;
2. $S(t)$ is *not* folded at $p(t)$ for every $t \in (t_k - \epsilon,\ t_k + \epsilon)$.

Proof. Let $\kappa(p(t))$ be the crease curvature at the point $p(t)$, and let $\kappa_g(p(t)) = \kappa(p(t))\cos\frac{\theta(p(t))}{2}$ be the crease's flattened (geodesic) curvature. If $\theta(p(t)) \neq 0$, the

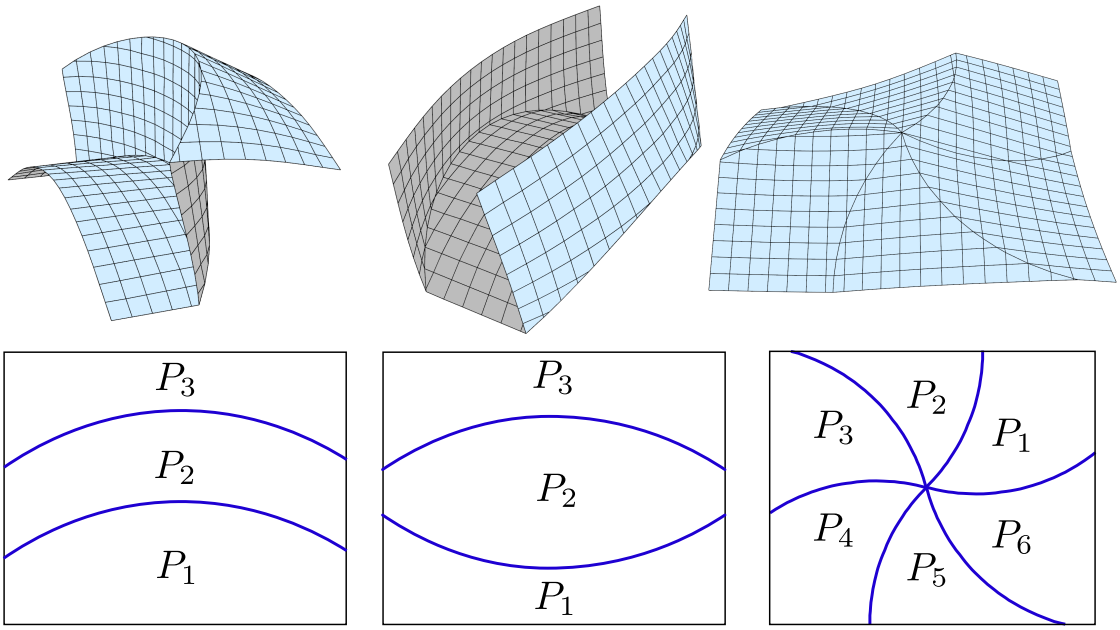


Figure 7.12: *Propagation of constraints in crease patterns. Bottom row: Crease patterns. Top row: Curved folding of the crease patterns. Left and center columns: A deformation in the patch P_1 dictates most of the shape of the patch P_2, which in turn dictates most of the patch P_3. The propagation of deformation is generally global, and depends on the directions of the rulings. In these cases one can choose a mountain/valley assignment for one fold, which already determines the M/V assignment of the next fold (left: valley-mountain, center: valley-valley). Right column: In more complicated crease patterns, for instance those with a vertex, the process is more involved, as there are also some compatibility conditions the patches must satisfy.*

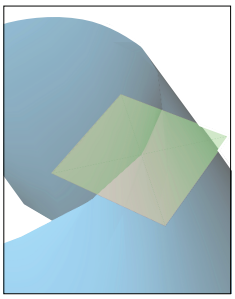
claim follows from the discontinuity of the two tangent planes at $p(t)$: a folding corresponds to a different choice of the tangent planes, forming an angle of $\theta(p(t))$ with each other, and any small continuous deformation cannot move from a folded to a non-folded configuration or vice-versa. Finally, the non-planarity of $p(t_k)$ implies $\theta(p(t_k)) \neq 0$, since $\theta(p(t_k)) = 0$ would mean that the normal curvature of the crease curve is 0, and therefore the tangent of the crease curve is parallel to the ruling direction. But by Lemma 12 and Corollary 16 in [Demaine et al. 2015] this implies that the curve has a kink at $p(t_k)$, contradicting the fact that $S(t)$ is C^2 when restricted to the patches $S_1(t), S_2(t)$. □

Therefore it is impossible to fold a crease point that is non-planar. For a non-planar point that is not folded, any small deformation keeps it that way. Folding can only happen after flattening the point, and if the surface is already folded, any small deformation keeps it folded. Thus, the decision whether to fold or not can only be

done when the crease points are planar, and no extra care needs be taken if the crease is already folded. With this in mind, we note the following observation.

Theorem 36. A non-planar curved crease point p on a curved folded surface S is folded if and only if the osculating plane of the crease curve at p is locally a supporting plane for the patches S_1, S_2 intersecting at p.

This follows directly from the fact that the tangent planes on both sides of a crease curve coincide if the surface is smooth there, but along a folded crease they are reflections of one another through the crease curve's osculating plane. Planar crease points along a curved crease, while not folded, still satisfy this constraint, since around these points the tangent planes are exactly the same as the osculating plane of the curve, though even the slightest surface deformation might change that.



7.4.3 Discretization

We saw that folding happens exactly when both sides of the surface around a crease are in the same half-space of the osculating plane of the crease curve. We discretize this condition by constraining the tangents of the discrete parametric (grid) lines of the two DOG patches to be on the same side of the crease curve's discrete osculating plane. See Fig. 7.13 for the notation. The DOG edges intersecting the crease can be considered as discrete surface tangents originating at the crease points. In the notation of Fig. 7.13, each of the two patches has its own duplicate of the edge (e_1 or e_2) intersecting the crease curve. In the starting, flat configuration the two edges coincide, but a folding movement creates a discontinuity between them. We denote the discrete surface tangents on both sides of the crease curve as $t_1 = \frac{e_1}{\|e_1\|}, t_2 = \frac{e_2}{\|e_2\|}$. The binormal of the crease curve, i.e., the normal of its osculating plane, is $B = \frac{e_f \times e_b}{\|e_f \times e_b\|}$, noting that e_f, e_b always coincide for both patches.

The supporting plane constraint can be written as

$$\operatorname{sgn}(\langle t_1, B\rangle) + sgn(\langle t_2, B\rangle) = 0 \tag{7.1}$$

with

$$\operatorname{sgn}(x) = \begin{cases} -1 : \text{if } x < 0, \\ 0 : \text{if } x = 0, \\ 1 : \text{if } x > 0. \end{cases} \tag{7.2}$$

Constraint (7.1) can be simplified by replacing B with the cross product $e_f \times e_b$. Furthermore, if one assumes an isometric deformation, it is possible to replace t_1, t_2 by the non-normalized edges e_1, e_2, arriving at:

$$\operatorname{sgn}(\langle e_1, e_f \times e_b\rangle) + sgn(\langle e_2, e_f \times e_b\rangle) = 0. \tag{7.3}$$

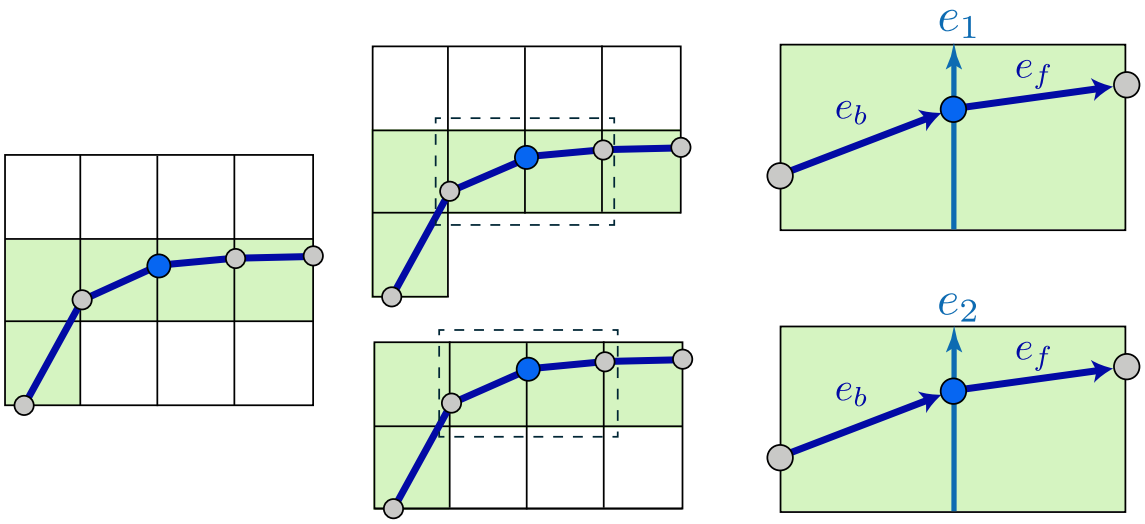


Figure 7.13: *Notation for edges in a discrete crease pattern. Left: a flat DOG with a discrete crease curve. Center: We represent creases by duplicating the quads that contain the crease curve, which results in different connected components, or patches. The positions of the vertices on the curve, i.e., its intersection points with the grid edges, are constrained to match on both patches. Right: Notation for a duplicated grid edge (e_1, e_2) intersecting the crease curve at the blue point, and the two crease edges e_f, e_b.*

In Sec. 7.6 we show how to plug this constraint into an optimization framework to model folding and bending of curved folded DOGs (see Fig. 7.10).

7.4.4 Discussion

There are multiple equivalent characterizations for a folded crease over a curved folded surface. We now point out some key properties of our chosen discretization (7.1) and briefly discuss how these properties are not satisfied by other possible constraint choices.

Suitable for homotopy based optimization methods Our constraint is satisfied on a flat mesh. In this sense we consider a point along a curve with zero normal curvature as both folded and not folded.

Minimal and generally non-intrusive Once a curved crease on a piecewise DOG is folded, one no longer needs to take explicit care for it to stay folded. The effect of Eq. (7.3) on an already folded surface is null. The tangent discontinuity caused by the folding implies that a folded crease remains folded under local deformations,

and in order for it to become unfolded, one needs to first flatten it, as is the case for a piecewise smooth curved folded surface. We also capture the converse: a discrete curved crease can only be folded when starting from a planar point (Theorem 35).

An alternative constraint for folding could be e.g. enforcing discontinuities along the tangents t_1, t_2, but this results in losing the feasibility of the flat models. Moreover, in the discrete case a minor discontinuity can still arise even though there is no fold, i.e., where Eq. (7.1) is not satisfied, thus numerically giving the impression of a fold when visually there is none. Another option is to define folded configurations as those satisfying a similar but simpler smooth constraint:

$$\langle t_1, B \rangle + \langle t_2, B \rangle = 0. \tag{7.4}$$

This condition is satisfied exactly in flat models and in any piecewise smooth curved folded surface, since tangent planes along a folded creased curve are reflections of each other w.r.t. the osculating plane of the crease curve. However, this condition is not satisfied *exactly* on every folded piecewise DOG (as evident by all models in this chapter). An exception to this is the class of curved creases with zero torsion, in which case the folds are simply formed as a global plane reflection [Mitani and Igarashi 2011]. Therefore, enforcing constraint (7.4) as a hard constraint is too restrictive in practice, while enforcing it softly creates a condition that, unlike the smooth case, does not vanish once a crease is folded and hence is not minimal in our sense.

7.5 Folding angles and mountain-valley assignments

In this section we propose tools to constrain the folding angles and their direction (mountain or valley) during deformation, in order to provide designers with additional expressive and intuitive control. We first show how to constrain folding angles, i.e., the angles between the tangent planes around a crease. We implement this by constraining DOG tangent angles, resulting in a simple quadratic constraint. We then devise a tool to differentiate between mountain and valley folds. Our derivations work for both curved and straight origami creases.

7.5.1 Folding angle

A folding deformation can be seen as a rotation of the surface patches' tangent planes hinged on the tangent of the crease curve. The tangent of a straight fold is constant, and so is the folding angle, while on a curved crease, the tangent varies and often the folding angles change along the crease. In both cases, if the folding angle at a given point is θ, then the surface tangent vectors on both sides of the crease that

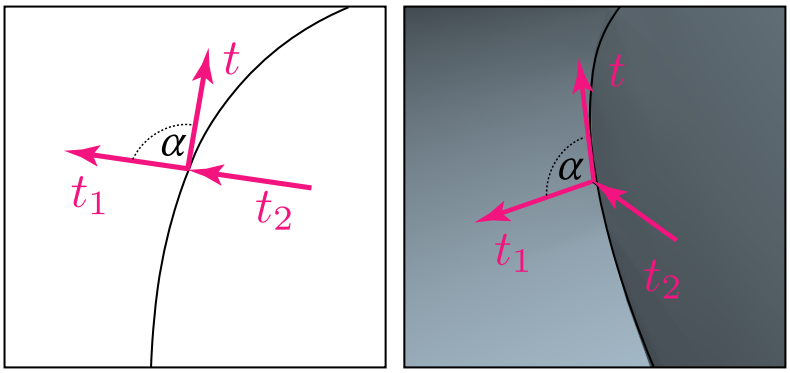


Figure 7.14: *Left: A flattened configuration of a crease curve and its tangent t, forming an angle α with surface tangents t_1, t_2, which are equal in this flat state ($t_1 = t_2$). Right: A folded isometric configuration with a tangent discontinuity $t_1 \neq t_2$. Lemma 37 shows the connection between the angle α and the folding angle θ in the smooth case, stating that $\langle t_1, t_2 \rangle = \cos^2\alpha + \sin^2\alpha \cos\theta$.*

are orthogonal to the crease tangent form an angle of θ, while the surface tangent vectors that are parallel to the crease remain parallel to each other. The following lemma shows the relation between the angle formed by surface tangent vectors that are equal in the flattened configuration and the folding angle (see Fig. 7.14):

Lemma 37. Let t_1, t_2 be surface tangent vectors on two sides of a crease curve at a given point p that are equal to each other in the isometrically flattened state of the developable surface. Let t be the crease curve tangent at p. Assuming the surface went through a curved folding isometric deformation and the folding angle at crease point p is θ, the surface tangent vectors satisfy:

$$\langle t_1, t_2 \rangle = \cos^2\alpha + \sin^2\alpha \cos\theta, \tag{7.5}$$

where α is the angle between t and t_1. Note that this angle is preserved under isometry.

Proof. We denote by t, n, b the vectors of the Frenet frame of the curved crease at p. We wish to express the surface tangent vectors t_1, t_2 in the local coordinates of this Frenet frame. In the flat isometric configuration, t_1, t_2 coincide and can be written as $\cos(\alpha)t + \sin(\alpha)n$. A folding angle of θ means that relative to the Frenet frame of the curve, the surface tangent on one side of the curve was rotated by angle $\frac{\theta}{2}$ about the crease curve tangent t, and the surface tangent on the other side by $-\frac{\theta}{2}$. Thus w.l.o.g.

$$\begin{aligned} t_1 &= \cos(\alpha)t + \sin(\alpha)\left(\cos\left(\tfrac{\theta}{2}\right)n + \sin\left(\tfrac{\theta}{2}\right)b\right), \\ t_2 &= \cos(\alpha)t + \sin(\alpha)\left(\cos\left(\tfrac{\theta}{2}\right)n - \sin\left(\tfrac{\theta}{2}\right)b\right). \end{aligned}$$

The proof is concluded by computing $\langle t_1, t_2 \rangle$ and plugging in the trigonometric identity $\cos\theta = \cos^2\frac{\theta}{2} - \sin^2\frac{\theta}{2}$. □

We discretize Lemma 37 by looking at angles between edges of the DOG emanating from crease points (see Fig. 7.15). Using the notation of Fig. 7.15, we discretize the tangent of the crease curve at a given point by looking at the incident edge vectors e_f, e_b:

$$t = \frac{\|e_b\| e_f + \|e_f\| e_b}{\left\| \|e_b\| e_f + \|e_f\| e_b \right\|}. \tag{7.6}$$

If the two edge vectors e_b, e_f are not collinear, t as above is the tangent at the point to the unique circle passing through the point and its two neighbors.

Under an isometric deformation, t_1, t_2 are linear in the vertex positions, α is constant and Eq. (7.5) is quadratic.

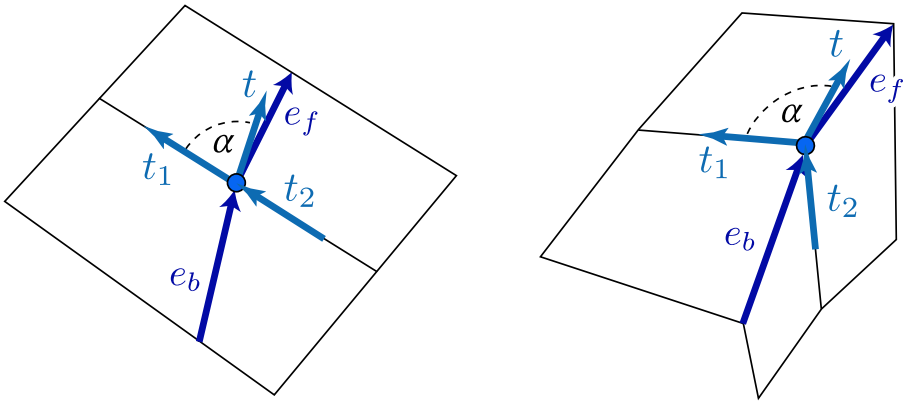


Figure 7.15: *Discretizing Lemma 37 (see Fig. 7.14) at a crease point by using the normalized DOG edges emanating from the crease point as surface tangents.*

7.5.2 Mountain/valley assignments

As mentioned in Sec. 7.4.1, there are a few combinatorial degrees of freedom in choosing the type of fold, i.e., the choice of surfaces on each side of the curve (Fig. 7.11). We follow [Demaine et al. 2015] to distinguish between the two types of folded configurations in Fig. 7.11 by calling one choice a *mountain fold* and the other a *valley fold*. We would like to emphasize that for straight folds, this degree of freedom always exists, but on curved creases it often does not. In fact, in many crease patterns it is often only possible to choose one mountain/valley (M/V) assignment, and the remaining assignments are determined by the propagation of the rulings, leaving only the combinatorial degrees of freedom of whether a crease is folded or not, embodied by Eq. (7.1). We distinguish M/V folds by looking at whether a tangent of one

surface patch is above or below the tangent plane of the second surface patch at the crease point, for a consistent choice of orientation. This can be achieved by the following constraint:

$$\langle t_1, t \times t_2 \rangle \leq 0, \tag{7.7}$$

where t is the tangent of the oriented crease curve and $t \times t_2$ is the normal of the tangent plane of the second surface patch (the one that has t_2 as a tangent vector). The orientation of t determines whether a mountain or a valley fold is chosen. By the cyclic property of the triple product, the left hand side of Eq. (7.7) is also equal to $\langle t_2, t_1 \times t \rangle$. As we are only interested in the sign of the left side of (7.7), we can replace t with the simpler $t^* = \|e_b\| e_f + \|e_f\| e_b$, which is linear under isometry.

To simplify notation for Sec. 7.6, we reformulate our mountain/valley constraint with an equality by using the Heaviside step function:

$$\mathrm{H}(x) = \begin{cases} 0 : \text{if } x \leq 0, \\ 1 : \text{if } x > 0, \end{cases} \tag{7.8}$$

and write the mountain/valley condition as:

$$H(\langle t_1, t^* \times t_2 \rangle) = 0. \tag{7.9}$$

7.6 Freeform deformations of curved folded surfaces

We employ the tools developed in Sec. 7.4 and Sec. 7.5 to devise a simple folding and bending algorithm for deforming piecewise DOGs. The algorithm aims to minimize an objective function while keeping the DOG constraints and ensuring the formation of folds along all crease curves and a specific M/V assignment on a crease when such an assignment is given as input.

7.6.1 Problem setup

We model our curved folded surfaces as a quad mesh, with a separate connected component for each patch. We denote the set of n mesh vertices in $\mathbb{R}^3$ by V, the vertex positions (variables) by $x \in \mathbb{R}^{3n}$, and the quad mesh faces by F. Each connected component is a DOG, i.e., it has the connectivity of a subset of $\mathbb{Z}^2$ and satisfies the DOG angle constraints, which we denote as $\phi_{d_i}(x) = 0, 1 \leq i \leq m$.

We are interested in deformations that fold the surface along all crease curves in a given crease pattern using Theorem 36 and enforcing Eq. (7.3) and optionally the mountain/valley assignment Eq. (7.9). We enforce these constraints on all crease points, which are points on crease curves that are not crease vertices, with the exception of crease points that have the following degeneracies on the flattened mesh (see Fig. 7.16):

1. degenerate osculating plane: crease points with a curvature smaller than a threshold κ_ε;
2. degenerate edge: crease points on an edge, splitting it into two parts where one is shorter than $\varepsilon_r\%$ of the other;
3. degenerate angle with the intersecting DOG tangent: crease points where the tangent directions t_1, t_2 form an angle with one of the edges e_f, e_b that is smaller than ε_α.

We use the constants $\kappa_\epsilon = 1\text{e-}5, \varepsilon_r = 5, \varepsilon_\alpha = 3°$. We denote the set of all folding constraints and the mountain/valley constraints (Eq. (7.9)) by $\phi_{f_j}(x) = 0, 1 \leq f_j \leq n_f$.

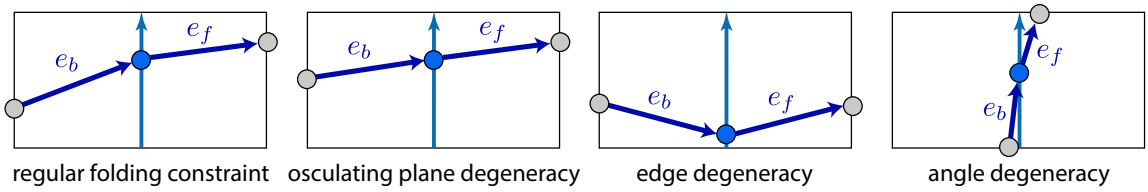


Figure 7.16: *A folding edge constraint defined on a blue crease point splitting the blue edge and degenerate cases where we do not enforce the constraint. From left to right: A regular folding edge constraint, instabilities in the osculating plane's normal as* $\frac{e_b \times e_f}{\|e_b \times e_f\|}$ *caused by* e_b, e_f *being almost collinear, degenerate edges as one part of the edge split by the blue crease point is comparably very short, and lastly a very small angle between* e_b *and the DOG edge crossing the blue point. An angle degeneracy often occurs before or after an edge degeneracy.*

We only enforce the DOG angle constraints and the folding and mountain/valley constraints as hard constraints. The objective function, which we denote by $f(x)$, is composed of a weighted sum of a bending objective, soft positional constraints, soft dihedral constraints and soft patch-continuity constraints.

Isometry is enforced as a soft constraint as advised by the degrees of freedom analysis in Chapter 6, but we emphasize that all our results have an average relative edge stretch that is less than 0.003, and a maximum stretch below 0.004, where our surfaces are normalized to have an average edge length of 1. We also encode the linear continuity constraints between patches as a soft constraint, as we have noticed a significant improvement in the quality and smoothness of crease patterns when these are enforced as a soft penalty with a large weight, and our results have an average continuity deviation of 0.0002 and a maximum of 0.0035. We also note that the constrained shape space analysis in Sec. 5.2 only concerns the DOG angle constraints, and complicated crease patterns give rise to a large set of additional linear constraints.

The problems we solve in this section can be written in the form:

$$\begin{aligned} &\arg\min_x f(x) \\ &\text{subject to} \\ &\phi_{d_i}(x) = 0, \quad i = 1, \ldots, m, \\ &\phi_{f_j}(x) = 0, \quad j = 1, \ldots, n_f, \end{aligned} \tag{7.10}$$

where f is specified in Sec. 7.6.2. We handle the combinatorial constraints ϕ_{f_j} by using a penalty based method (Sec. 7.6.3), and solve our problem with an iterative sequential quadratic programming (SQP) solver with a line search (Sec. 7.6.4). The line search strategy ensures that the DOG angle constraints ϕ_{d_i} are satisfied numerically while the combinatorial constraints ϕ_{f_j} are satisfied exactly.

7.6.2 Objectives and convex Hessian approximations

Our objective f is composed of a weighted sum of various functions measuring bending, stretch, positional constraints and dihedral angles. We use the integrated squared mean curvature bending objective taken from Chapter 4, and we exploit the fact that it is quadratic and convex under isometric deformations [Bergou et al. 2006]:

$$f_{\mathrm{H}}(x) = 0.5x^t(L^t M^{-1} L)x, \tag{7.11}$$

where L is the DOG Laplacian and M is a diagonal mass matrix defined by the DOG vertex area (Chapter 4).

We employ Lemma 37 to constrain the folding angle at a given crease point using the constraint

$$\phi_{\mathrm{d}_{c^i}}(x) := \langle t_1^i, t_2^i \rangle - \cos^2(\alpha^i) - \sin^2(\alpha^i)\cos(\theta^i) = 0, \tag{7.12}$$

where c^i is the index of the crease point along the edge defined as a linear combination of two vertices, t_1^i, t_2^i, α^i are as defined in Sec. 7.5.1 and Fig. 7.15, and θ^i is the desired dihedral angle at the crease point c^i. Under isometry t_1^i, t_2^i are linear in the net vertex locations, α^i is fixed, and the constraint is quadratic.

Let e be an edge on the net mesh, l_e its length and l_e^0 the length in the reference net mesh. We define the following quadratic isometry constraints:

$$\phi_{\mathrm{iso}}(x)_e := l_e^2 - l_e^{0\,2} = 0. \tag{7.13}$$

We maintain continuity along the patches with a set of linear equality constraints on duplicated crease points, which we denote by $\phi_{\mathrm{cont}}(x) = 0$. Lastly, we allow the user to specify positional constraints on vertices or crease edge points, including constraints requiring two points to have the same coordinate, as used in the creation of the ring

at Fig. 7.19 and the annulus at Fig. 3.1 (also see accompanied video for [Rabinovich et al. 2019]). We denote this user defined set of constraints by $\phi_{\text{pos}}(x) = 0$. We enforce the dihedral, positional, isometry and patches continuity constraints in a soft manner by using a penalty on their squared deviation, denoted accordingly by $f_{\text{D}}(x), f_{\text{pos}}(x), f_{\text{iso}}(x), f_{\text{cont}}(x)$. These sum of squared objectives are not convex, and we replace their Hessian in our optimization with their Gauss-Newton's Hessian approximation. We do the same for $\sum \|\phi^*_{f_i}(x)\|_2^2$.

The objective we optimize is then:

$$f(x) = w_{\text{H}} f_{\text{H}} + w_{\text{pos}} f_{\text{pos}} + w_{\text{D}} f_{\text{D}} + w_{\text{iso}} f_{\text{iso}} + w_{\text{cont}} f_{\text{cont}}. \tag{7.14}$$

Throughout the chapter, unless stated otherwise, we use $w_{\text{H}} = 1$, $w_{\text{pos}} = 5$, $w_{\text{D}} = 100$, $w_{\text{iso}} = 20000/|E|$, $w_{\text{cont}} = 1\text{e}4$, where $|E|$ is the number of edges in the net mesh (i.e., for a mesh with 1000 edges $w_{\text{iso}} = 20$). Our meshes are always scaled to have an average edge length of 1 and therefore using a different resolution for the same geometry keeps our bending objective the same, but scales the isometric objective by the number of edges.

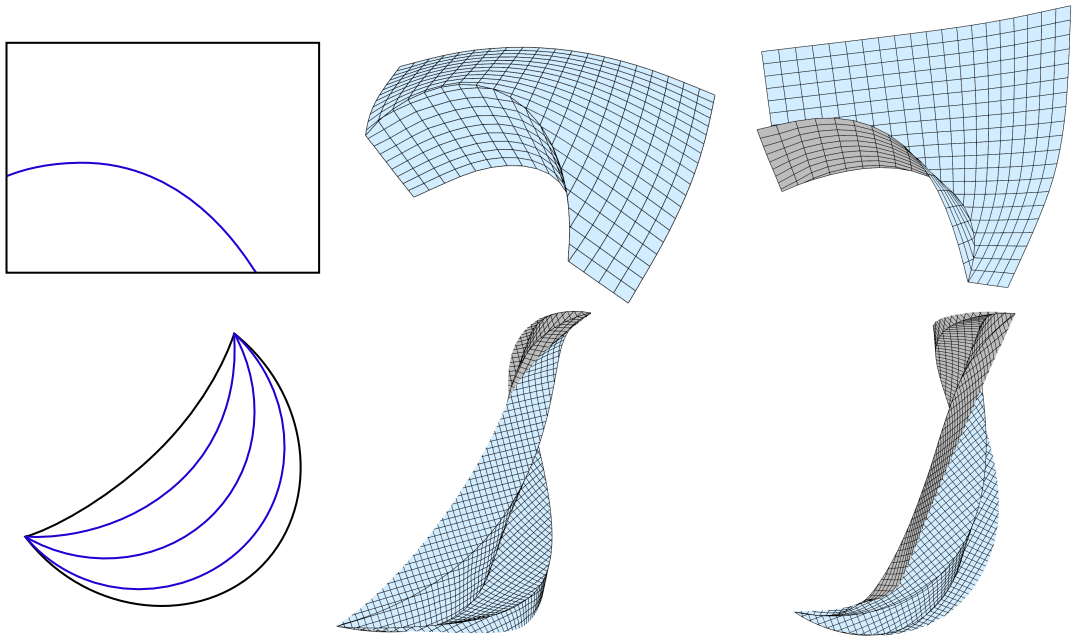


Figure 7.17: *Using the optional mountain/valley assignment input (Sec. 7.5.2) on a single crease curve. Each crease pattern is deformed with the same positional constraints, induced by a curve constrained flow, but with a different mountain/valley assignment along one crease, enforced by Eq.* (7.9)*. In the banana shaped model (bottom row), the rest of the mountain/valley assignments are then uniquely determined.*

7.6.3 Folding constraints

Motivated by the fact that in the smooth case, one cannot move from a folded to a non-folded configuration around a non-planar point, we strive to always satisfy $\phi_{f_j}(x) = 0$ exactly. The common starting point of a flat surface is an interesting case, as it is a bifurcation point between surfaces satisfying Theorem 36 and those that do not, which also holds for the discretization Eq. (7.1). To that end, we solve our problem with an iterative sequential quadratic programming (SQP) solver with a line search, complemented with two simple strategies to handle the constraints $\phi_{f_i}(x)$:

1. a penalty term [Nocedal and Wright 2006] punishing deviation from the constraints;
2. a line search method that backtracks if the resulting mesh does not exactly satisfy $\phi_{f_j}(x) = 0, i = 1, ..., n_f$.

Since the functions $\text{sgn}(x), H(x)$ involved in the constraints $\phi_{f_j}(x)$ are not C^1, we replace them by the approximations:

$$\begin{aligned} &\text{sgn}(x) \approx \tanh(hx) \\ &H(x) \approx \begin{cases} 0 & : \text{if } x \leq 0, \\ \frac{x^2}{x^2+\delta} & : \text{if } x = 0, \delta > 0 \end{cases} \end{aligned} \tag{7.15}$$

using the fixed parameters $h = 1000, \delta =$ 1e-5. Our approximation for $H(x)$ is taken from [Li et al. 2012; Poranne et al. 2017]. The use in a homotopy based optimization necessitates an approximation for $H(x)$ that vanishes on a flat mesh, and therefore we do not use the common approximation for the Heaviside function $H(x) \approx \hat{H}(x) = \frac{1+\tanh(hx)}{2}$ because $\hat{H}(0) = \frac{1}{2}$.

We refer to the approximated constraints as $\phi^*_{f_j}(x)$ and replace the optimization problem (7.10) with the following problem:

$$\begin{aligned} &\arg\min_x f(x) + \omega \sum \|\phi^*_{f_j}(x)\|^2 \\ &\text{subject to} \\ &\phi_{d_i}(x) = 0, \quad i = 1, \ldots, m. \end{aligned} \tag{7.16}$$

Here, $\omega > 0$ is a metaparameter initialized as $\omega_0 = 1$, which doubles its value if the line search cannot find a point satisfying the supporting plane conditions exactly. In practice, the penalty term only affects points that are very close to being planar, while approaching zero very quickly around already folded points.

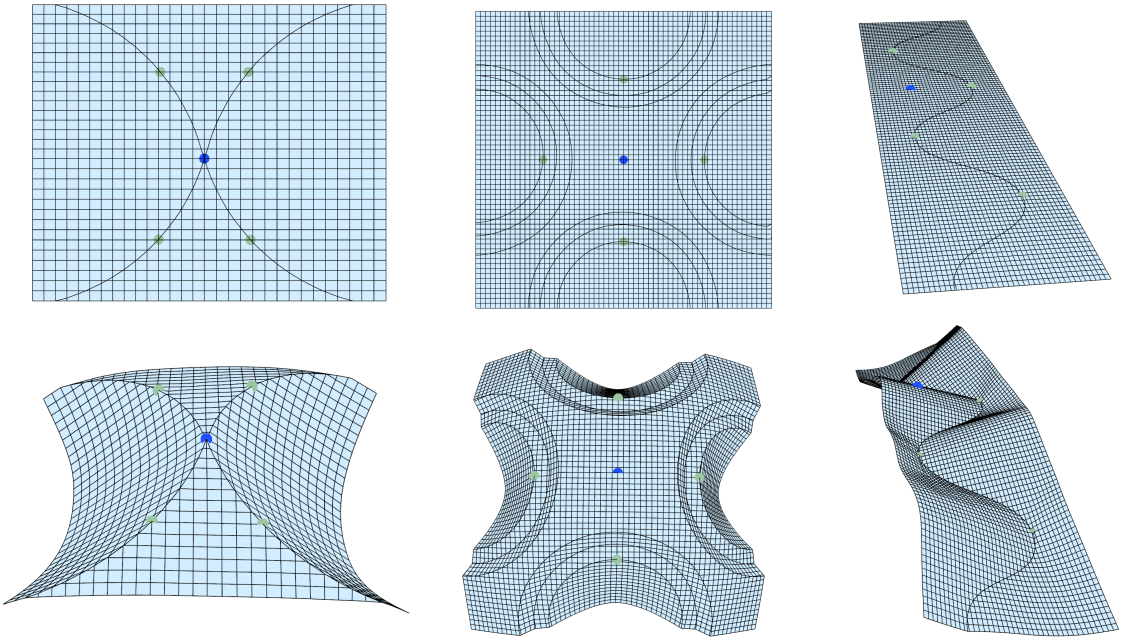


Figure 7.18: *Using the optional folding angle constraints (Sec. 7.5.1) on a sparse set of crease points. These examples are deformed by constraining the folding angle of a set of points (in green), without specifying their folding orientation, and by setting a single positional constraint (in blue).*

7.6.4 Equality constrained SQP

For ease of notation, we use the following to refer to the objective of Eq. (7.16):

$$f_\omega(x) = f(x) + \omega \sum \|\phi^*_{f_j}(x)\|^2. \tag{7.17}$$

We minimize (7.16) using SQP with a line search [Nocedal and Wright 2006]. Given a set of variables at a given iteration x^k and current values of Lagrange multipliers λ^k, a line search equality constrained SQP algorithm iteratively finds the next direction for a line search of Eq. (7.16), by which it sets the next variables x^{k+1} by solving a KKT system of the form:

$$\begin{aligned} &K \begin{pmatrix} d^{k+1} \\ \lambda^{k+1} \end{pmatrix} = b, \text{ where} \\ &K = \begin{pmatrix} \Delta^2_{xx}\mathcal{L}(x^k, \lambda^k) & J^\mathsf{T}(x^k) \\ J(x^k) & 0 \end{pmatrix}, \quad b = \begin{pmatrix} \nabla f_\omega(x^k) \\ -\phi_{d_i}(x^k) \end{pmatrix}, \end{aligned} \tag{7.18}$$

where $J(x)$ is the Jacobian of the equality constraints in Eq. (7.16), $\Delta^2_{xx}\mathcal{L}(x, \lambda) = H_{f_\omega}(x) + \sum \lambda^k_i \Delta^2_{xx}\phi_{d_i}(x)$ is the Hessian of the Lagrangian of the problem and $H_{f_\omega}(x)$ is the Hessian of $f_\omega(x)$.

Following Sec. 5.2, we use a minimally modified Jacobian $J^*(x)$ to deal with singularities in DOGs. We also replace the Hessian of the objective $H_{f_\omega}(x)$ by a convex approximation, which we denote by $H^*_{f_\omega}(x)$, as detailed in Sec. 7.6.2, and thus replace the system (7.18) by:

$$K \begin{pmatrix} d^{k+1} \\ \lambda^{k+1} \end{pmatrix} = b, \text{ where} \tag{7.19}$$
$$K = \begin{pmatrix} H^*_{f_\omega}(x^k) + \sum \lambda_i^k \Delta^2_{xx} \phi_{d_i}(x^k) & J^{*\mathsf{T}}(x^k) \\ J^*(x^k) & 0 \end{pmatrix}, \quad b = \begin{pmatrix} \nabla f_\omega(x^k) \\ -\phi_{d_i}(x^k) \end{pmatrix}.$$

In Sec. 5.2 we discretized Laplacian metric flows by solving a similar system with a Laplacian instead of the Lagrangian's Hessian. However, we found that replacing the Laplacian by the Lagrangian followed by convexifying the Hessian performs significantly better, especially on larger models. As common in SQP algorithms, we use a merit function to guide our line search, defined as a combination of the objective and the constraints. The line search chooses step sizes that reduce the objective and keep the DOG angle constraints numerically feasible, while backtracking if a point does not satisfy the constraints ϕ_{f_j} exactly.

This removes the need for the slower LBFGS constraints' projection used at Chapter 5.2. We use the L_2 merit function [Nocedal and Wright 2006]:

$$\psi(x; \mu) = f_\omega(x) + \mu \sum \|\phi_{d_i}(x)\|_2, \tag{7.20}$$

where we update the parameter μ^k at each iteration using the absolute values of the Lagrange multipliers [Nocedal and Wright 2006]:

$$\mu^k = \max\{c_\mu \cdot \max\{|\lambda_i^k|\},\ \mu_0\}, \tag{7.21}$$

with $c_\mu = 1.1$ and $\mu_0 = 0.05$.

Resulting in a system that is both faster, and also scales better to larger models. We employ the optimization described above (see Eq. (7.14)) in an interactive freeform editing system. The input to our system is the flat domain boundary and the curves of the crease pattern, represented by polylines. These can be easily generated from any standard vector graphics format by sampling the smooth curves therein. Our system computes an arrangement of the input curves using CGAL's arrangement model [The CGAL Project 2019; Wein et al. 2019; Zukerman et al. 2019] and solves the symmetric indefinite linear systems as required by the optimization (Eq. (7.19)) using Pardiso [Kourounis et al. 2018; Verbosio et al. 2017; De Coninck et al. 2016]. We ran our experiments on a 16-core Ryzen Threadripper 1950X clocked at 3.4 GHz. Our editing system supports setting point handle positional constraints, as can be seen in Figures 3.1, 7.12, 7.19, and the second and last model from the left in Fig.

7.10. A stress test for our algorithm is shown in Fig. 7.19, with a crease pattern containing 20 different creases that all automatically bend and fold, driven only by point handle positional constraints. We also support constraining a crease curve by specifying its curvature and torsion in a curve constrained flow, see Figures 7.10 and 7.17, as well as prescribing a sparse set of dihedral angles along crease points, see Fig. 7.18.

Fig. 7.17 is the only case where we supply a mountain/valley assignment as input, while Fig. 7.18 displays the only models designed by constraining dihedral angles. The curve constrained positional constraints, as well as the dihedral angles, are interpolated for improved quality (see Fig. 5.6). To maintain interactive frame rates in handle based editing tasks, we run a fixed number of SQP iterations per frame, which we set to 5. On models with 500, 1000, 2000, 4000 vertices these 5 SQP iterations run on average at more than 39, 19, 9, 4 frames per second. Hence our system is able to handle realtime interaction of meshes with around 2000 vertices. The two concentric circles folds in Fig. 3.1 originate from a mesh with about 5500 vertices. They are designed by simply penalizing the distance of a single pair of vertices while interpolating the penalty weight, and their final forms are reached in about 30 seconds. We refer the reader to our supplementary video for [Rabinovich et al. 2019] for further results, including interactive editing examples.

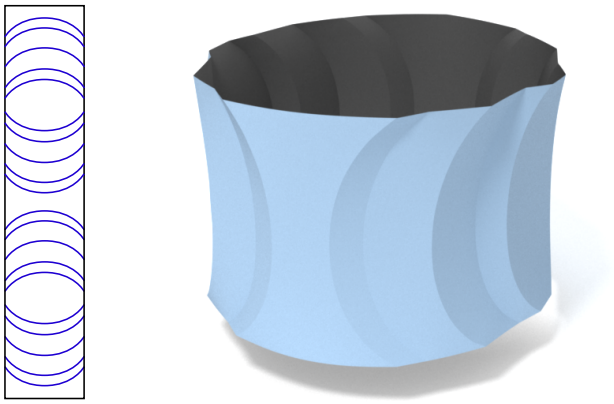


Figure 7.19: *A curved folded ring. The crease pattern is taken from [Mitani 2019] and contains 20 different creases. It was deformed by enforcing our folding constraints (Sec. 7.6), without specifying any folding angles or mountain/valley assignments, together with positional constraints pushing the vertices on the right and left boundary to match such that a loop is formed, while also minimizing an additional bending energy term for the glued area to smooth it.*

CHAPTER 8

Conclusion

In this thesis we developed a discrete theory alongside a computational framework for modeling developable surfaces. Motivated by computational design tasks, and inspired by the rich theory of nets in discrete differential geometry, we strived to develop a discrete theory that respects fundamental aspects of the smooth one. Throughout this thesis we saw how this approach pays off, allowing us to step above the state of the art in modeling developable surfaces. Our theory was developed with an eye towards computation, demonstrating the rich interplay between between geometry processing and discrete differential geometry.

8.1 Recapitulation of core contributions

The core of our contribution lies in Chapter 3, where we introduce DOGs. Our model is simple, yet exact. Using the model of DOGs allows for freeform exploration of developable surfaces in a way not possible before. In particular, the problem of locking, both when modeling isometry and when modeling other developable surface deformations, is a non-issue while modeling with DOGs. The model of DOGs joins the plethora of classes of nets in discrete differential geometry. As demonstrated in Chapter 3, DOGs extend and interacts with previous works in discrete nets such as Voss surfaces [Wunderlich 1951], conjugate nets [Bobenko and Suris 2008], and conical nets [Liu et al. 2006]. We also note here that DOGs are a special case of the newer discrete geodesic parallel coordinates [Wang et al. 2019] model.

The geometric attributes derived on DOGs in Chapter 4 can be used to define accurate as well as simple functionals. A discrete Laplacian operator is one of

the most fundamental operators in geometry processing, and our DOG Laplacian possesses the desired properties of a discrete Laplace operator: it is symmetric, positive semidefinite and linearly precise. Chapter 5 shows that our optimization problems are well defined and well understood, and supplies us with guarantees on the solution space when performing homotopy based optimization. Chapter 6 gives theoretically grounded and practical solutions for modeling isometries on planar surfaces. Chapter 7 then develop tools to model curved folded surfaces with discrete orthogonal geodesic nets.

The editing system developed at the end of Chapter 7 is the first interactive freeform editing tool capable of modeling bending and folding of complicated crease patterns, and uses many of the contributions and observations detailed in all of the previous chapters.

8.2 Reflections and outlook

This thesis is the product of over three years of combined research in both geometry processing and discrete differential geometry (DDG). These are two very different fields, that are nevertheless intimately linked. I often switched back and forth between a practical optimization oriented mindset, and a more theoretical one. Quite often I was surprised to see how both approaches lead to the same solutions. I had the fortune of having mentors covering both aspects, and the chance to give talks and receive feedback on our work from researchers in both fields.

It is here that I would like to emphasize and reiterate that DDG, and in particular the theory of discrete nets, can be used to solve *practical* geometry processing tasks. The simplicity of the models, their minimalistic nature, and the commonplace framework for analyzing their convergence and rigidity, all translate into precise as well as optimization-friendly geometric models. This becomes more and more important as of late, as modern geometry processing research often involves non-linear and non-convex optimization problems, which are hard to solve or analyze. DDG offers tools to understand, analyze and generate guarantees for complicated geometry related optimization tasks [Rabinovich and Sorkine-Hornung 2019]. This work is definitely not the first to use that fact, and in particular the use of nets in DDG is commonplace in works on architectural geometry [Pottmann et al. 2015]. However, this thesis is a prime example of a side by side development of a discrete DDG theory and computational applied tools. The motto of DDG, “Discretize the whole theory, not just the equations” served as a strong guideline. In practice, it forced the practitioner, even the applied one, to build their work on the basis of others while leaving a new groundwork for followups. Thus, each work on DOGs during the time of my PhD was built on top of products of previous ones.

It is my hope that this thesis will further motivate people in the graphics and CAD communities to study and develop the theory of DDG, in order to solve applied geometric problems.

8.3 Future work

Modeling developable surfaces. Inspired by [Stein et al. 2018], we leave it as future research to apply the theory of DOGs for approximation of arbitrary geometry using piecewise developable surfaces. A clear current limitation of our work is the speed for interactive editing, which is restricted to rather coarse models of around 2000 vertices. The current resolution limitation does not allow us to model curved folded crease patterns composed of many patches, or with creases that meet almost in parallel (see Fig. 8.1). We leave scaling of the optimization to future work, possibly by using a multigrid solver and/or domain decomposition algorithms on the DOG grids, or different grid resolutions along different parts of the mesh. In addition, we find that we lack tools and objectives to enforce symmetry of the designed shapes. In particular, we would like to look at folding of curved symmetric plane wallpapers and tessellations [Demaine et al. 2015; Mundilova 2019] (see Fig. 8.2). Finally, we note that our curved folding modeling only allows for deformations of a given fixed input crease pattern. Optimizing and changing an input crease pattern, as done in origami modeling tools [Tachi 2010], could offer new and exciting ways to discover and design curved folded surfaces.

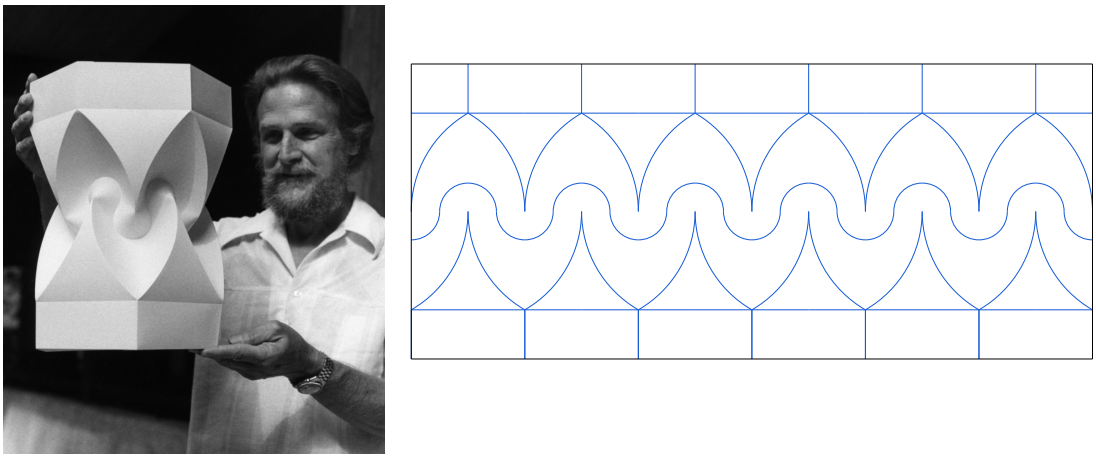


Figure 8.1: *Left: David Huffman and his "hexagonal column with cusps" (photo courtesy of UC Santa Cruz). Right: The corresponding crease pattern, composed of creases that meet almost in parallel, the folding of which cannot be modeled with a coarse mesh.*

Physical simulations of planar sheets. Our work does not take physical reality

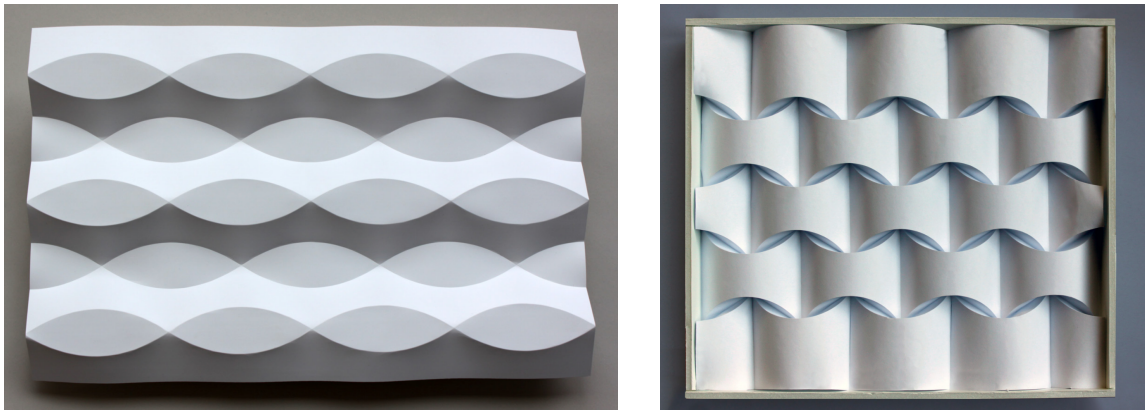


Figure 8.2: *Curved folded tessellations (images taken from [Demaine et al. 2011b]). Left: Huffman's lens tessellation. Right: Huffman's "Arches" design using parabolas and lines.*

constraints into account, such as collisions, material thickness and elasticity properties, making the models created in our system not necessarily realizable. Incorporating our model into a physically accurate design system could potentially alleviate this limitation. We believe that developing a discrete mathematical theory for the formation of creases and their modeling is an interesting avenue for future research.

Laplacians on discrete nets. 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, in particular for other classes of nets with non planar quadrilaterals.

Discrete geodesic nets. The study of non-orthogonal discrete geodesic nets can be beneficial for modeling developable surfaces (Def. 7), as well as deformations and isometries on more general doubly curved surfaces. In particular, it might be possible to define a discrete Gaussian curvature on these nets through angle deficit using Def. 6 in Sec. 3.3.2, as well as to develop a theory for modeling isometries of doubly curved surfaces using geodesic nets.

Orthogonal Chebyshev nets for modeling developable surfaces. We note that in the smooth case, as a consequence of the *formula of Hazzidaki*([Hazzidakis 1879]), an orthogonal Chebyshev net is developable, and is in fact an orthogonal geodesic net. An orthogonal geodesic net is also Chebyshev (Sec. 3.5). Hence, an orthogonal net is Chebyshev if and only if it is orthogonal. Finding a model of orthogonality for an exact discrete Chebyshev net that does not suffer from locking, could be an interesting avenue for further research.

Bibliography

E. D. Adler. *" A New Unity!" The Art and Pedagogy of Josef Albers*. PhD thesis, 2004.

Q. Alessio. *Membrane locking in discrete shell theories*. PhD thesis, Niedersächsische Staats-und Universitätsbibliothek Göttingen, 2012.

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

P. Alliez, G. Ucelli, C. Gotsman, and M. Attene. Recent advances in remeshing of surfaces. In *Shape analysis and structuring*, pages 53–82. Springer, 2008.

M. Bergou, M. Wardetzky, D. Harmon, D. Zorin, and E. Grinspun. A quadratic bending model for inextensible surfaces. In *Symposium on Geometry Processing*, pages 227–230, 2006.

R. L. Bishop. There is more than one way to frame a curve. *The American Mathematical Monthly*, 82(3):246–251, 1975.

P. Bo and W. Wang. Geodesic-controlled developable surfaces for modeling paper bending. *Comput. Graph. Forum*, 26(3):365–374, 2007.

P. Bo, Y. Zheng, X. Jia, and C. Zhang. Multi-strip smooth developable surfaces from sparse design curves. *Computer-Aided Design*, 114, 2019.

A. Bobenko and U. Pinkall. Discrete surfaces with constant negative gaussian curvature and the hirota equation. *J. Differential Geom.*, 43(3):527–611, 1996. URL `http://projecteuclid.org/euclid.jdg/1214458324`.

A. I. Bobenko and U. Pinkall. Discretization of surfaces and integrable systems. *Oxford lecture series in mathematics and its applications*, 16:3–58, 1999.

A. I. Bobenko and P. Schröder. Discrete Willmore flow. In *Proc. Symposium on Geometry Processing*, 2005. ISBN 3-905673-24-X. URL `http://dl.acm.org/citation.cfm?id=1281920.1281937`.

A. I. Bobenko and Y. B. Suris. *Discrete differential geometry: integrable structure*, volume 98 of *Graduate studies in mathematics*. American Mathematical Society, Providence (R.I.), 2008. ISBN 978-0-8218-4700-8.

A. I. Bobenko and Y. B. Suris. Discrete Koenigs nets and discrete isothermic surfaces. *International Mathematics Research Notices*, 2009(11):1976–2012, 2009.

A. I. Bobenko, T. Hoffmann, and B. A. Springborn. Minimal surfaces from circle patterns: Geometry from combinatorics. *Annals of Mathematics*, 164(1):231–264, 2006.

A. I. Bobenko, H. Pottmann, and J. Wallner. A curvature theory for discrete surfaces based on mesh parallelity. *Mathematische Annalen*, 348(1):1–24, 2010.

M. Botsch, M. Pauly, M. H. Gross, and L. Kobbelt. Primo: coupled prisms for intuitive surface modeling. In *Symposium on Geometry Processing*, number CONF, pages 11–20, 2006.

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

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

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

H. U. Buri, I. Stotz, and Y. Weinand. Curved folded plate timber structures. In *IABSE-IASS 2011 London Symposium*, number CONF. IABSE-IASS, 2011.

D. Chapelle and K.-J. Bathe. Fundamental considerations for the finite element analysis of shell structures. *Computers & Structures*, 66(1):19–36, 1998.

H.-Y. Chen, I.-K. Lee, S. Leopoldseder, H. Pottmann, T. Randrup, and J. Wallner. On surface approximation using developable surfaces. *Graphical Models and Image Processing*, 61(2):110 – 124, 1999. ISSN 1077-3169. doi: http://dx.doi.org/10.1006/gmip.1999.0487. URL `http://www.sciencedirect.com/science/article/pii/S1077316999904872`.

K. Crane and M. Wardetzky. A glimpse into discrete differential geometry. *Notices of the American Mathematical Society*, 64:1153–1159, 11 2017.

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

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

A. De Coninck, B. De Baets, D. Kourounis, F. Verbosio, O. Schenk, S. Maenhout, and J. Fostier. Needles: Toward large-scale genomic prediction with marker-by-environment interaction. *Genetics*, 203(1):543–555, 2016. ISSN 0016-6731. doi: 10.1534/genetics.115.179887. URL `http://dx.doi.org/10.1534/genetics.115.179887`.

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

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

E. D. Demaine, M. L. Demaine, and D. Koschitz. Reconstructing david huffman's legacy in curved-crease folding. *Origami*, 5:39–52, 2011b.

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

E. D. Demaine, M. L. Demaine, D. A. Huffman, D. Koschitz, and T. Tachi. Characterization of curved creases and rulings: Design and analysis of lens tessellations. *Origami*, 6:209–230, 2015.

E. D. Demaine, M. L. Demaine, D. A. Huffman, D. Koschitz, and T. Tachi. Conic crease patterns with reflecting rule lines. arXiv preprint arXiv:1812.01167, 2018.

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

M. Desbrun, E. Grinspun, P. Schröder, and M. Wardetzky. Discrete differential geometry: An applied introduction. In *SIGGRAPH Course Notes*, volume 1, 2005.

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

J. P. Duncan and J. Duncan. Folded developables. *Proc. R. Soc. Lond. A*, 383(1784): 191–205, 1982a.

J. P. Duncan and J. L. Duncan. Folded developables. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 383(1784): 191–205, 1982b. ISSN 0080-4630. doi: 10.1098/rspa.1982.0126. URL `http://rspa.royalsocietypublishing.org/content/383/1784/191`.

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

W. H. Frey. Boundary triangulations approximating developable surfaces that interpolate a closed space curve. *International Journal of Foundations of Computer Science*, 13(02):285–302, 2002.

W. H. Frey. Modeling buckled developable surfaces by triangulation. *Computer Aided Design*, 36(4):299–313, 2004.

S. Fröhlich and M. Botsch. Example-driven deformations based on discrete shells. *Comput. Graph. Forum*, 30(8):2246–2257, 2011. ISSN 1467-8659.

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

D. Fuchs and S. Tabachnikov. *Mathematical omnibus: thirty lectures on classic mathematics*. Springer, 2007.

R. Goldenthal, D. Harmon, R. Fattal, M. Bercovier, and E. Grinspun. Efficient simulation of inextensible cloth. *ACM Trans. Graph.*, 26(3):49, 2007.

F. Gramazio and M. Kohler. *Made by robots: challenging architecture at a larger scale*. John Wiley & Sons, 2014.

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

E. Grinspun, A. N. Hirani, M. Desbrun, and P. Schröder. Discrete shells. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 62–67, 2003. ISBN 1-58113-659-5. URL `http://dl.acm.org/citation.cfm?id=846276.846284`.

B. Grünbaum. *Arrangements and spreads*. American Mathematical Society, 1972.

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

J. Hazzidakis. Ueber einige eigenschaften der flächen mit constantem krümmungsmaass. *Journal für die reine und angewandte Mathematik*, 88:68–73, 1879.

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

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

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

T. Hoffmann, A. O. Sageman-Furnas, and M. Wardetzky. A discrete parametrized surface theory in $\mathbb{R}^3$. *International Mathematics Research Notices*, 2017(14): 4217–4258, 2017.

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

H.-D. Hwang and S.-H. Yoon. Constructing developable surfaces by wrapping cones and cylinders. *Computer-Aided Design*, 58:230–235, 2015.

I. Izmestiev. Classification of flexible kokotsakis polyhedra with quadrangular base. *International Mathematics Research Notices*, 2017(3):715–808, 2016.

A. Jacobson, D. Panozzo, et al. libigl: A simple C++ geometry processing library, 2015. http://libigl.github.io/libigl/.

M. Kazhdan, J. Solomon, and M. Ben-Chen. Can mean-curvature flow be modified to be non-singular? *Comput. Graph. Forum*, 31(5):1745–1754, 2012. ISSN 0167-7055.

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

M. Kilian, S. Flöry, Z. Chen, N. J. Mitra, A. Sheffer, and H. Pottmann. Curved folding. *ACM Trans. Graph.*, 27(3):75:1–75:9, 2008. ISSN 0730-0301.

M. Kilian, A. Monszpart, and N. J. Mitra. String actuated curved folded surfaces. *ACM Trans. Graph.*, 36(3):25:1–25:13, 2017. ISSN 0730-0301.

D. Kourounis, A. Fuchs, and O. Schenk. Towards the next generation of multiperiod optimal power flow solvers. *IEEE Transactions on Power Systems*, PP(99):1–10, 2018. ISSN 0885-8950. doi: 10.1109/TPWRS.2017.2789187. URL `https://doi.org/10.1109/TPWRS.2017.2789187`.

S. Lawrence. Developable surfaces: Their history and application. *Nexus Network Journal*, 13(3):701–714, 2011. ISSN 1522-4600. doi: 10.1007/s00004-011-0087-z. URL `http://dx.doi.org/10.1007/s00004-011-0087-z`.

Z. Li, S. Wang, and X. Lin. Variable selection and estimation in generalized linear models with the seamless l_0 penalty. *Canadian Journal of Statistics*, 40(4):745–769, 2012.

Y. Lipman, O. Sorkine, D. Levin, and D. Cohen-Or. Linear rotation-invariant coordinates for meshes. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 479–487, New York, NY, USA, 2005. ACM. doi: 10.1145/1186822.1073217. URL `http://doi.acm.org/10.1145/1186822.1073217`.

Y. Liu, H. Pottmann, J. Wallner, Y.-L. Yang, and W. Wang. Geometric modeling with conical meshes and developable surfaces. *ACM Trans. Graph.*, 25(3), 2006. ISSN 0730-0301.

W. S. Massey. Surfaces of gaussian curvature zero in euclidean 3-space. *Tohoku Mathematical Journal, Second Series*, 14(1):73–79, 1962.

J. Mitani. A design method for 3d origami based on rotational sweep. *Computer-Aided Design and Applications*, 6(1):69–79, 2009.

J. Mitani. Column-shaped origami design based on mirror reflections. *Journal for Geometry and Graphics*, 16(2):185–194, 2012.

J. Mitani. *Curved-Folding Origami Design*. CRC Press, 2019.

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

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

K. Mundilova. On mathematical folding of curved crease origami: Sliding developables and parametrizations of folds into cylinders and cones. *Computer-Aided Design*, 2019. accepted for publication.

R. Narain, A. Samii, and J. F. O'Brien. Adaptive anisotropic remeshing for cloth simulation. *ACM transactions on graphics (TOG)*, 31(6):152, 2012.

R. Narain, T. Pfaff, and J. F. O'Brien. Folding and crumpling adaptive sheets. *ACM Trans. Graph.*, 32(4), 2013.

J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–782, 1980.

J. Nocedal and S. J. Wright. *Sequential quadratic programming*. Springer, 2006.

Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, and L. Liu. Anderson acceleration for geometry optimization and physics simulation. *ACM Trans. Graph.*, 37(4): 42:1–42:14, July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201290. URL `http://doi.acm.org/10.1145/3197517.3201290`.

F. Pérez and J. A. Suárez. Quasi-developable b-spline surfaces in ship hull design. *Computer-Aided Design*, 39(10):853–862, 2007.

U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

K. Polthier and M. Schmies. *Straightest geodesics on polyhedral surfaces*. ACM, 2006.

R. Poranne, M. Tarini, S. Huber, D. Panozzo, and O. Sorkine-Hornung. Autocuts: Simultaneous distortion and cut optimization for uv mapping. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)*, 36(6), 2017.

H. Pottmann and J. Wallner. Approximation algorithms for developable surfaces. *Comput. Aided Geom. Des.*, 16(6):539 – 556, 1999. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/S0167-8396(99)00012-6. URL `http://www.sciencedirect.com/science/article/pii/S0167839699000126`.

H. Pottmann and J. Wallner. *Computational line geometry*. Mathematics and visualization. Springer, Berlin, Heidelberg, New York, 2001. ISBN 3-540-42058-4.

H. Pottmann, M. Eigensatz, A. Vaxman, and J. Wallner. Architectural geometry. *Computers & graphics*, 47:145–164, 2015.

M. Rabinovich and O. Sorkine-Hornung. Nets in discrete differential geometry. Symposium on Geometry Processing Graduate School, 2019.

M. Rabinovich, R. Poranne, D. Panozzo, and O. Sorkine-Hornung. Scalable locally injective mappings. *ACM Transactions on Graphics*, 36(2):16:1–16:16, Apr. 2017.

M. Rabinovich, T. Hoffmann, and O. Sorkine-Hornung. Discrete geodesic nets for modeling developable surfaces. *ACM Transactions on Graphics*, 37(2), 2018a.

M. Rabinovich, T. Hoffmann, and O. Sorkine-Hornung. The shape space of discrete orthogonal geodesic nets. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)*, 37(6), 2018b.

M. Rabinovich, T. Hoffmann, and O. Sorkine-Hornung. Modeling curved folding with freeform deformations. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)*, 38(6), 2019.

R. D. Resch. Portfolio of shaded computer images. *Proceedings of the IEEE*, 62(4): 496–502, 1974.

K. Rose, A. Sheffer, J. Wither, M.-P. Cani, and B. Thibert. Developable surfaces from arbitrary sketched boundaries. In *Proc. Symposium on Geometry Processing*, pages 163–172, 2007. ISBN 978-3-905673-46-3. URL `http://dl.acm.org/citation.cfm?id=1281991.1282014`.

S. Rusinkiewicz. Estimating curvatures and their derivatives on triangle meshes. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pages 486–493. IEEE, 2004.

R. Sauer. *Differenzengeometrie*. Springer, 1970.

W. K. Schief, A. I. Bobenko, and T. Hoffmann. On the integrability of infinitesimal and finite deformations of polyhedral surfaces. In *Discrete Differential Geometry*, pages 67–93, Basel, 2008. Birkhäuser Basel. ISBN 978-3-7643-8621-4. doi: 10.1007/978-3-7643-8621-4_4. URL `http://dx.doi.org/10.1007/978-3-7643-8621-4_4`.

C. Schreck, D. Rohmer, S. Hahmann, M.-P. Cani, S. Jin, C. C. L. Wang, and J.-F. Bloch. Nonsmooth developable geometry for interactively animating paper crumpling. *ACM Trans. Graph.*, 35(1):10, 2015. URL `http://dblp.uni-trier.de/db/journals/tog/tog35.html#SchreckRHCJWB15`.

C. Schreck, D. Rohmer, and S. Hahmann. Interactive paper tearing. *Computer Graphics Forum (Eurographics)*, 36(2):95–106, 2017.

T. W. Sederberg, P. Gao, G. Wang, and H. Mu. 2-D shape blending: an intrinsic solution to the vertex path problem. In *Proc. ACM SIGGRAPH*, pages 15–18, 1993.

D. R. Shelden. *Digital surface representation and the constructibility of Gehry's architecture*. PhD thesis, Massachusetts Institute of Technology, 2002.

J. Solomon, E. Vouga, M. Wardetzky, and E. Grinspun. Flexible developable surfaces. *Comput. Graph. Forum*, 31(5):1567–1576, 2012.

O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Proc. Symposium on Geometry Processing*, pages 109–116, 2007.

M. Spivak. *A Comprehensive introduction to differential geometry, 1.* Publish or Perish, Houston, TX, 1999. ISBN 0-914098-70-5. URL `http://opac.inria.fr/record=b1117511`.

O. Stein, E. Grinspun, and K. Crane. Developability of triangle meshes. *ACM Trans. Graph.*, 37(4), 2018.

G. Sundaramoorthi, A. Yezzi, and A. C. Mennucci. Sobolev active contours. *International Journal of Computer Vision*, 73(3):345–366, 2007.

T. Tachi. Simulation of rigid origami. *Origami*, 4:175–187, 2009.

T. Tachi. Freeform variations of origami. *J. Geom. Graph*, 14(2):203–215, 2010.

T. Tachi. One-dof rigid foldable structures from space curves. In *Proceedings of the IABSE-IASS Symposium*, pages 20–23, 2011.

T. Tachi. Composite rigid-foldable curved origami structure. In *1st International Conference on Transformable Architecture (Transformables 2013), Seville, Spain, Sept*, pages 18–20, 2013.

C. Tang, P. Bo, J. Wallner, and H. Pottmann. Interactive design of developable surfaces. *ACM Trans. Graph.*, 35(2):12:1–12:12, 2016. ISSN 0730-0301.

The CGAL Project. *CGAL User and Reference Manual*. CGAL Editorial Board, 4.14 edition, 2019. URL `https://doc.cgal.org/4.14/Manual/packages.html`.

F. Verbosio, A. D. Coninck, D. Kourounis, and O. Schenk. Enhancing the scalability of selected inversion factorization algorithms in genomic prediction. *Journal of Computational Science*, 22(Supplement C):99 – 108, 2017. ISSN 1877-7503. URL `https://doi.org/10.1016/j.jocs.2017.08.013`.

H. Wang, D. Pellis, F. Rist, H. Pottmann, and C. Müller. Discrete geodesic parallel coordinates. *ACM Trans. Graphics*, 38(6), 2019. Proc. SIGGRAPH ASIA.

W. Wang, J. Wallner, and Y. Liu. An angle criterion for conical mesh vertices. *Journal for Geometry and Graphics*, 11(2):199–208, 2007.

M. Wardetzky. *Discrete differential operators on polyhedral surfaces – convergence and approximation*. PhD thesis, Freie Universität Berlin, 2007.

M. Wardetzky, S. Mathur, F. Kälberer, and E. Grinspun. Discrete laplace operators: no free lunch. In *Proc. Symposium on Geometry Processing*, pages 33–37, 2007.

R. Wein, E. Berberich, E. Fogel, D. Halperin, M. Hemmer, O. Salzman, and B. Zukerman. 2d arrangements. In *CGAL User and Reference Manual*. CGAL Editorial Board, 4.14 edition, 2019. URL `https://doc.cgal.org/4.14/Manual/packages.html#PkgArrangementOnSurface2`.

M. Wertheim. Cones, curves, shells, towers: He made paper jump to life. *The New York Times*, 2(4):5, 2004.

W. Wunderlich. Zur Differenzengeometrie der Flächen konstanter negativer Krümmung. *Sitzungsber. Österr. Akad. Wiss.*, 160:39–77, 1951.

Y.-L. Yang, Y.-J. Yang, H. Pottmann, and N. J. Mitra. Shape space exploration of constrained meshes. *ACM Trans. Graph.*, 30(6), 2011.

B. Zukerman, R. Wein, and E. Fogel. 2d intersection of curves. In *CGAL User and Reference Manual*. CGAL Editorial Board, 4.14 edition, 2019. URL `https://doc.cgal.org/4.14/Manual/packages.html#PkgSurfaceSweep2`.