Technische Universität Wien
Institut für Diskrete Mathematik und Geometrie

Interactive Freeform Architectural Design with Nearly Developables and Cold Bent Glass

DISSERTATION

von

Konstantinos Gavriil

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines

Doktors der technischen Wissenschaften

eingereicht an der

Technischen Universität Wien
Fakultät für Mathematik und Geoinformation

Betreuer:

Univ. Prof. Dr. **Helmut Pottmann**

Gutachter:

Univ. Prof. Dr. **Niloy J. Mitra**

Dr. **Tor Dokken**

Wien, August 2020

Vienna University of Technology
Institute of Discrete Mathematics and Geometry

Interactive Freeform Architectural Design with Nearly Developables and Cold Bent Glass

DOCTORAL THESIS

by

Konstantinos Gavriil

submitted in partial fulfillment of the requirements for the degree of

Doctor of Technical Sciences

to the

Faculty of Mathematics and Geoinformation
Vienna University of Technology

Advisor:

Univ. Prof. Dr. **Helmut Pottmann**

External reviewers:

Univ. Prof. Dr. **Niloy J. Mitra**

Dr. **Tor Dokken**

Vienna, August 2020

"Any sufficiently advanced technology is indistinguishable from magic."
— Arthur C. Clarke, *Profiles of the Future*

Abstract

Interactive design of freeform architectural surface panelizations is at the core of this PhD thesis. We provide the computational framework for dealing with two important types of paneling elements. Specifically, we focus on certain types of developable surfaces and cold bent glass panels, all relevant to contemporary freeform architecture.

To this end, we initially present a novel method for increasing the developability of a B-spline surface. We use the property that the Gauss image of a developable surface is 1-dimensional and can be locally well approximated by circles. This is cast into an algorithm for thinning the Gauss image by increasing the planarity of the Gauss images of appropriate neighborhoods. A variation of the main method allows us to tackle the problem of paneling a freeform architectural surface with developable panels, in particular enforcing rotational cylindrical, rotational conical and planar panels, which are the main preferred types of developable panels in architecture due to the reduced cost of manufacturing. We are interested in near developability, rather than exact developability, so the optimization approach is sufficient. The motivation behind this is the fact that most materials allow for a little bit of stretching and therefore developability needs not be satisfied to a high degree.

One such material is glass which is the main focus of the second panelization problem of this thesis. Toughened glass can withstand higher stresses, and therefore allows initially planar glass panels to be elastically bent and fixed at ambient temperatures to a curved frame. This process is called cold bending and it produces panels that can exhibit double curvature, providing a cost- and energy-efficient alternative of higher optical quality than traditional hot bent glass panels. However, it is very challenging to navigate the design space of cold bent glass panels due to the fragility of the material, which impedes the form-finding for practically feasible and aesthetically pleasing cold bent glass façades. We present an interactive, data-driven approach for designing cold bent glass façades that can be seamlessly integrated into a typical architectural design pipeline. Our method allows non-expert users to interactively edit a parametric surface while providing real-time feedback on the deformed shape and maximum stress of cold bent glass panels. Designs are automatically refined to minimize several fairness criteria while maximal stresses are kept within glass limits. We achieve interactive frame rates by using a differentiable mixture density network trained from more than a million simulations. Given a curved boundary, our regression model is capable of handling multistable configurations and accurately predicting the equilibrium shape of the panel and its corresponding maximal stress. We show predictions are highly accurate and validate our results with a physical realization of a cold bent glass surface. For both applications explored in this work, a plethora of results and examples are provided.

Acknowledgements

I thank first and foremost my supervisor Helmut Pottmann for introducing me to architectural geometry and supporting me through this doctoral journey. His invaluable guidance helped me work on my flaws and develop my strengths. His intellect, extensive knowledge, and exceptional research quality were and will be an example for my work. I thank him for having me as a student.

I thank Ioannis Emiris, my MSc thesis advisor and project coordinator for the ARCADES network, as well as Christos Konaxis, technical coordinator of ARCADES. I am grateful for their support during my first research steps. I also extend my gratitude to everyone I met through the ARCADES network and particularly Laurent Busé, Bernard Mourrain, Carlos D'Andrea, Evelyne Hubert, Josef Schicho, Tor Dokken, Georg Muntingh, Oliver J.D. Barrowclough, and all the fellow early stage researchers—Clément Laroche, Yairon Cid Ruiz, Fatmanur Yildirim, Ahmed Blidia, Alvaro Fuentes Suárez, Jan Legerský, Francesco Patrizi, Andrea Raffo, Evangelos Bartzos, Theofanis Katsoulis, Sotirios Chouliaras, and Michael Jimenez.

A big thank you goes to the former Evolute GmbH team—Alexander Schiftner, Mathieu Huard, Mathias Höbinger, and Heinz Pottmann—for the very friendly and productive work environment during the first years of my PhD, as well as the wealth of real-life practical experience they shared with me.

I also thank all of my colleagues at TU Wien—Christian Müller, Martin Kilian, Martin Peternell, Przemyslaw Musialski, Udo Hertrich-Jeromin, Doris Hotz, Birgit Slama, Ronald Haidvogl, Hui Wang, Ildar Gilmutdinov, Stefan Pillwein, Kurt Leimer, Michael Birsak, Christian Hafner, Felix Dellinger, Ulrike Grill, Ruzica Mijic, Klara Mundilova, Dino Rossegger, Mason Pember, Gudrun Szewieczek, María Lara Miró, Aditya Kapilavai, Heinz Schmiedhofer, and collaborators Florian Rist and Davide Pellis.

A special thank you goes to Bernd Bickel, Ruslan Guseinov, Jesús Pérez, and Paul Henderson for our successful collaboration.

I am very grateful to Tor Dokken and Niloy Mitra for agreeing to be in my thesis commitee.

I thank Sophie Pennetier and Corinna Datsiou for their expert input on the industrial and practical applications of cold bent glass. I also thank Zaha Hadid Architects and Waagner Biro for providing the architectural datasets.

I thank Lefteris Kirousis for properly introducing me to research and Dimitrios Thilikos for supporting me during my undergraduate studies. For the same reason, I am also thankful to Andreas Nikolaidis and Spyridoula Kanta. I thank my high school teacher Nicos Hiotelis for inspiring me to pursue mathematics.

I am grateful to my parents, brother, and grandparents. This achievement is the result of their unconditional and continuous support, their sacrifices, encouragement and advice. And of course I thank Pixel, a certain Irish terrier whose company kept

me sane. He has been a constant reminder of the importance of being playful and going on long walks.

Most importantly, I thank Lydia Simantiraki for being by my side during this challenging time, for her love, patience, encouragement and understanding.

Thank you all!

Vienna, August 2020
ϰɣ

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant Agreement No. 675789.

Contents

Introduction 1

The recent advances in computational design tools, manufacturing techniques, and structural materials have provided architects and designers the freedom to challenge the boundaries of contemporary architecture and achieve truly ambitious and beautiful designs. Freeform architecture in particular is one example of such design philosophy in which the architectural space is enclosed by a collection of freeform surfaces, usually called *façades* (see Figure 1.1). Many interesting challenges arise when attempting to realize such a freeform design, and these constitute the main focus of a relatively recent research area called *Architectural Geometry*.

In particular, the realization of such freeform designs usually involves the covering of the design surface by smaller simpler elements called *panels*. This process is called *panelization* and a lot of problems revolve around it. These include but are not limited to the choice of combinatorics of the panelization, the choice of panel geometry, i.e. shape, scale, curvature etc., and adhering to a number of constraints, such as those imposed by the chosen material, and the project budget.

The designer's toolset nowadays is of course mostly digital and the majority of

Figure 1.1: The Guggenheim Museum Bilbao (Bilbao, Spain) is a famous example of freeform architecture, designed by Frank Gehry, a significant representative of this architectural movement.

the design workflow is done on *computer-aided design* (CAD) software. While traditional digital tools allow for modeling ambitious forms, these might not be necessarily realizable due to a number of constraints. These constraints can derive from aesthetic ambition, structural requirements, material limitations, and economic restrictions. It is clear that the incorporation of these requirements to available design tools is of great practical importance.

Two developments that emerged from this need are *constraint-based surface modeling* which embeds the design constraints, usually geometric in nature, into the modeling stage, and *fabrication-aware design* which involves design tools that generate buildable structures of a certain type. A special case of the latter is *material-aware design* which allows for modeling geometry that is manufacturable by a specific material, such as glass. In this research work, we are interested in both types of geometry modeling tools.

1.1 Motivation

The main motivation of this thesis is the facilitation of the design creative process via the improvement of the modeling tools at the designer's disposal. To this end, we provide a computational framework for two important instances of panelization problems, namely realizing a freeform design with (i) developable surfaces and (ii) cold bent glass panels. We present in this section the importance of these contributions.

A *developable surface* is a surface that can be flattened onto a plane without distortion. Panels of this type are popular in freeform architecture since they can be achieved from initially planar elements only by bending, making them a cost-effective choice. The bending process varies depending on the material choice, which includes metal, wood, and glass.

Furthermore, glass processing methods have allowed glass to be used as more than a decorative or functional material. In particular, the introduction of a thin layer of residual compression to the glass panel surface during manufacturing leads to the production of *toughened glass*. This type of glass can withstand larger amounts of stress, allowing larger amounts of deformation than traditional glass, supporting

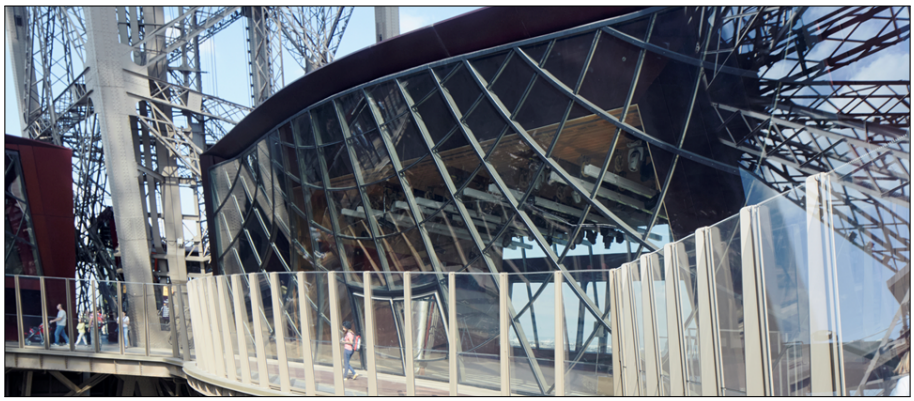


Figure 1.2: The façades of the Eiffel tower pavilions (Paris, France) are an example of a freeform surface being approximated by simpler elements. Specifically, cylindrical hot bent glass panels were used in this case to cover a non-developable surface. Photo by Evolute GmbH.

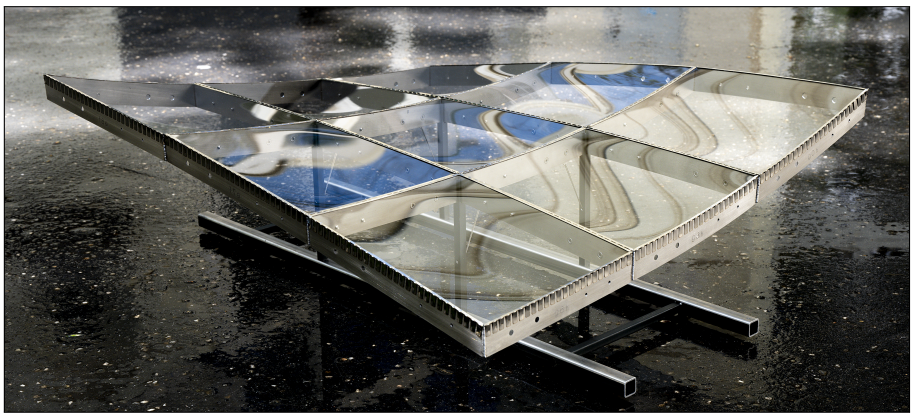


Figure 1.3: Proof-of-concept prototype of a cold bent glass panelization consisting of 3×3 panels at model scale.

its usage as a structural element. The additional deformation freedom also allows for an initially flat glass panel to be elastically bent, while staying within the material safety limits, to a desired curved shape. This process is called *cold bending*; in contrast to *hot bending* which involves the glass panel to be heated to transition temperature before molding it into the desired shape. The flat glass panel is bent to a pre-manufactured frame, usually with the help of clamps or presses, before being fixed into place with the use of mechanical fixings or structural adhesives. Interestingly, cold bent glass panels are not limited to single-curved (developable) surfaces, but also allow for double-curved shapes.

Freeform architecture has recently witnessed the increased use of this type of cold bent glass paneling elements (see Figure 1.3). They present several advantages making them an appealing alternative to the hot bending process and justifying their usage in freeform panelization. First of all, cold bent glass is much more cost- and energy-efficient to produce. The initial glass panel is always flat which removes the need to pre-bend it using furnaces and special molds that provide shape to the panel. Secondly, the use of molds during the hot bending process introduces minor visual artifacts to the surface, meaning that the alternative cold bent glass panels result to superior visual quality. Moreover, the cold bending process can be optionally carried out in situ allowing for the panels to be more easily transported in their flat state.

1.2 Contributions

Motivated by the arguments presented above, this thesis provides the following core contributions:

- We present a novel optimization method for increasing the developability of an arbitrary surface. It is based on local approximations of the surface by developable surfaces with planar and thus circular Gauss images.
- We employ the above methodology to the problem of paneling a freeform surface with (rotational) cylindrical, (rotational) conical and planar panels, which are the main preferred types of developable panels in architecture due to the reduced cost of manufacturing.

- We provide a computational framework for interactively designing a panelization of a freeform surface with cold bent glass panels. The interactive design tool employs a data-driven model, which was trained on a large number of costly simulations, that sufficiently predicts the shape and the maximal stress of a cold bent glass panel configuration.

1.3 Publications

The contributions of this thesis have been presented in the following publications:

- **Optimizing B-spline surfaces for developability and paneling architectural freeform surfaces.**

  Konstantinos Gavriil, Alexander Schiftner, and Helmut Pottmann.

  Computer-Aided Design, *111*, 29-43, 2019.

- **Computational Design of Cold Bent Glass Façades.**

  Konstantinos Gavriil, Ruslan Guseinov, Jesús Pérez, Davide Pellis, Paul Henderson, Florian Rist, Helmut Pottmann, and Bernd Bickel.

  ACM Trans. Graph., *39*(6), 208:1-208:16, 2020. Proc. *SIGGRAPH Asia 2020*.

While the extent of the research work conducted during this PhD is not limited to the above published material, we refer to only that relevant to this body of work. The above publications constitute the majority of the PhD research work and represent the main consistent research framework. The abstracts of the excluded publications are provided in Appendix B.

1.4 Overview and organization

In Chapter 2, we present a novel method for increasing the developability of B-spline surfaces, and employ it to the problem of paneling a freeform architectural surface with special types of developable panels that are of interest in architecture. We base the method on the fact that developable surfaces possess 1−dimensional Gauss images and can be locally approximated by surfaces with planar Gauss image. We present the necessary background as well as elaborate on the theory that justifies this approach in Section 2.2 before formulating the problem as an optimization problem in Section 2.3. The paneling problem is presented in Section 2.4 as an adapted optimization problem that employs the previous methodology. Special types of developables such as (rotational) cylindrical, (rotational) conical and planar panels are naturally handled by the method. Section 2.5 presents several results and experiments of this approach, as well as a short discussion on the advantages and disadvantages of the method.

In Chapter 3, we present the complete computational framework for designing panelizations with cold bent glass panels. The method is multidisciplinary, borrowing concepts and methods from material simulation, geometry optimization, machine learning, and interactive design. We present a brief overview of the distinct parts of the method in Section 3.1.3 before describing each part in detail. A description of the mechanical model used for the glass panel simulations is presented in Section 3.3. We present an appropriate geometry representation that is sufficient for

our needs in Section 3.2. In Section 3.4 we demonstrate the construction of the data-driven model that is at the core of the method. Multiple simulations were performed for randomized (within reasonable bounds) curved glass panel frames. A database was populated with the resulting shapes and maximal stress values for each simulation, which was later used to train a machine learning model. Specifically, we trained a *mixture density network* (MDN), capable of capturing the multimodality present in the dataset. In Section 3.5 we describe how we incorporated the MDN to an interactive design tool. Several applications and results, as well as validation experiments, are presented in Section 3.6.

Previous work for each of the problems is presented at the beginning of each respective chapter. Finally, Chapter 4 concludes this work and provides an outlook to future work.

Increasing Developability 2

2.1 Introduction

Developable surfaces can be locally mapped to a planar domain without distortion. Since they can be constructed from an initial planar state without stretching or tearing, only by bending, they represent the shapes obtainable with thin materials like sheet metal or paper which do not stretch. These surfaces are of great interest to many applications. Areas like architecture, manufacturing and design take advantage of the cost-reduced manufacturing process that developables have.

Developable surfaces have been well studied in classical differential geometry. Developable, twice differentiable surfaces are *single curved*, meaning one of the principal curvatures is zero. Thus, the Gauss curvature vanishes at every point. They are composed of special *ruled* surfaces with a constant tangent plane at all points of a ruling. As the surface normal vectors along a ruling agree, the Gauss image of a developable surface is 1-dimensional, i.e. a curve.

We base the main method of the chapter on this property of the Gauss image. However, our focus is not on exact developability, but rather on *nearly developable* surfaces which we characterize by nearly curve-like Gauss images. The motivation for our research is the fact that most materials allow for a little bit of stretching and therefore developability needs not be satisfied to a high degree in a variety of applications. In particular, we are interested in applications in architecture where various kinds of tolerances can be exploited to reduce the production cost of freeform skins. Our work fits into a larger research program on novel digital tools which consider key aspects of function and fabrication, including material behavior, already in the early design and digital modeling phase.

2.1.1 Related work

There is a vast amount of literature on developable surfaces, on their theory, their computational design using various types of representations and on their appearance in numerous applications. We limit this discussion to three main areas which are most closely related to our work: (i) developable Bezier and B-spline surfaces, (ii) discrete representations and nearly developable surfaces and (iii) their importance in paneling architectural surfaces.

Developable Bézier and B-spline surfaces. Lang and Röschel [1992] expressed developability of rational, in particular polynomial Bézier surfaces in a system of cubic equations. In general, this system cannot be solved in a simple way, but in various special cases, explicit solutions have been derived [Aumann, 1991, 2003; Chu & Chen, 2004; Chu & Séquin, 2002]. One can avoid these nonlinear constraints by using the projectively dual representation, where a developable is represented as the envelope of its tangent planes. For details, we refer to [Pottmann & Wallner, 2001, Section 6.2], but note that the dual representation is not sufficiently intuitive to be suitable for interactive design. Moreover, it is difficult to control singularities. A combination of the primal and the dual representation has been successfully employed for interactive design of developable NURBS surfaces by Tang et al. [2016].

Discrete representations and nearly developable surfaces. There are numerous papers which model developable surfaces with triangle meshes; we just refer to a few of them [Frey, 2004; Mitani & Suzuki, 2004; Rose et al., 2007; C. Wang & Tang, 2004]. Jung et al. [2015] improve on Decaudin et al. [2006] method that locally approximates neighborhoods around each mesh triangle with a cone. Liu, Pottmann, Wallner, et al. [2006] treat developable surfaces as a limit case of meshes from planar quads. Solomon et al. [2012] use a mesh approach to flexibly model the shapes achievable by bending and folding a given planar domain without stretching or tearing. An elegant discrete model of developable surfaces is provided by special quad meshes which discretize orthogonal nets of geodesics [Rabinovich et al., 2018a, 2018b].

Nearly developable surfaces appear in connection with specific applications, e.g. modeling ship hulls [Pérez & Suárez, 2007] and clothing [M. Chen & Tang, 2010] or segmenting meshes in geometry processing [Julius et al., 2005; Yamauchi et al., 2005]. Narain et al. [2013] go beyond developability and present a technique for simulating plastic deformation in sheets of thin materials, such as crumpled paper, dented metal, and wrinkled cloth. Closely related to our work is a paper by C. C. L. Wang et al. [2004] on increasing developability of a trimmed NURBS surface, but our approach and applications differ significantly.

Another very recent work with a strong connection to our research is the developable surface flow by Stein et al. [2018]. This flow is a gradient flow on the energy $\int_M \kappa_1^2 \, dA$, κ_1 being the smallest principal curvature. It constructs piecewise developable rather than globally developable surfaces as minimizers. The discrete model is based on triangulations whose vertex stars dominantly lie in pairs of planes. One could say that the surface is locally approximated by a pair of planes, their intersection representing the ruling direction. In a similar spirit, our local approximations are of higher order, as discussed below. Note that Stein et al. [2018] generate *piecewise* developable surfaces, where the arising pattern of developable patches is a result of the geometric flow and depends on the initial triangulation. We can increase developability of a single smooth surface without the introduction of tangent discontinuities. We can also allow for piecewise developable surfaces through an appropriate selection of knots and their multiplicities in the underlying B-spline surface, but our arrangements of developable patches are more restricted (and at the same time more controlled) than the ones by Stein et al. [2018].

Paneling architectural surfaces. Architectural surfaces need to be decomposed into panels, which is a key process and largely responsible for a cost effective solution. For an overview of the problems in this field we refer to [Pottmann et al., 2015].

In particular, we point to the paneling solution of Eigensatz et al. [2010]. It exploits various tolerances at seams and a cost model for the production of panels of different geometric types to suggest solutions within an optimization framework. The user provides the design surface and a suggested network of panel boundary curves, while the algorithm slightly adapts the design surface and network and optimally fills it with panels (patches). Our work can be considered as an extension in the sense that the panel boundaries are also subject to optimization with the overall goal of increasing developability of the individual panels. For developable and nearly developable surfaces in architecture, we further point to [Pottmann et al., 2008; Schiftner et al., 2013; Schneider & Mehrtens, 2013; Shelden, 2002].

2.1.2 Contributions

The main contributions covered in this chapter are as follows:

- We present a novel optimization method for increasing the developability of an arbitrary surface. It is based on local approximations of the surface by developable surfaces with planar and thus circular Gauss images. While we could also use other representations within our framework, we prefer B-splines in order to have simple access to smoothness of patches. Moreover, we naturally obtain a patchwork of regular quad combinatorics, which is a preferred arrangement in many architectural projects.

- We provide a justification of our approach in two ways: We discuss local approximations of developable surfaces, especially with those being characterized by a planar Gauss image. Moreover, we study the implications of a nearly curve-like Gauss image on the underlying surface, thus supporting our claim of achieving near developability through Gauss image thinning.

- We introduce a variation of the main method presented to tackle the problem of paneling a freeform surface with (rotational) cylindrical, (rotational) conical and planar panels, which are the main preferred types of developable panels in architecture due to the reduced cost of manufacturing.

- We provide results that illustrate the power of the proposed approach and outline potential directions for future research.

2.1.3 Overview

This chapter is organized as follows. In Section 2.2, we outline some important fundamentals for our work and, in section 2.3 present the main optimization algorithm step by step. Section 2.4 focuses on a variation of the main optimization algorithm which is designed for paneling a freeform surface with panels that are special cases of developable surfaces. We present the differences with the main algorithm and introduce any necessary new tools. In Section 2.5, we provide results on various data sets, including ones from real architectural projects. Moreover, we discuss advantages and shortcomings of our approach and outline future work.

2.2 Fundamentals

2.2.1 Local approximations of developable surfaces

We are interested in smooth or piecewise smooth developable surfaces S. They are composed of C^2 surface patches which fall into one of the following four categories: planes, general cylinders, general cones and tangent surfaces of space curves. Their *Gauss images* C, i.e. sets of unit normals viewed as points on the unit sphere S^2, are composed of *curves*. The junction points of C where more than two curve segments meet, correspond to planar patches on S. In the following, we discuss only the three non-trivial basic types: These are ruled surfaces with a constant tangent plane along each ruling. In other words, they are envelopes of a one-parameter family of planes.

We are interested in second order local approximations of these basic types. The following result is well-known (see, e.g. [Pottmann & Wallner, 2001, Theorem 6.1.4]) and closely related to the simple fact that the Gauss image of a developable surface is a spherical curve, which has an osculating circle at each of its regular points.

Lemma 2.2.1. *Along each ruling r, a nonplanar developable ruled surface S has second order contact with a rotational cone Γ (osculating cone). The vertex of this cone is the singular point of r (regression point). Γ is a rotational cylinder for a cylindrical ruling r (regression point at infinity) and it degenerates to a plane if r is an inflection ruling.*

Let us add a bit more detail for the generic case where S is the tangent surface of a space curve, $S: \ \mathbf{x}(u,v) = \mathbf{c}(u) + v\dot{\mathbf{c}}(u)$. This so-called regression curve $\mathbf{c}(u)$ is a singular curve on S. The osculating plane at $\mathbf{c}(u)$, spanned by $\dot{\mathbf{c}}, \ddot{\mathbf{c}}$, is the constant tangent plane of S along a ruling (isoparameter line $u = const$). If u is an arc length parameter, then the Frenet frame at $\mathbf{c}(u)$ is given by the tangent vector $\mathbf{e}_1 = \dot{\mathbf{c}}$, principal normal $\mathbf{e}_2 = \ddot{\mathbf{c}}/\kappa$ (with curvature $\kappa = \|\ddot{\mathbf{c}}\|$), and the binormal vector $\mathbf{e}_3 = \mathbf{e}_1 \times \mathbf{e}_2$. The Frenet equations can then be written in the form $\dot{\mathbf{e}}_i = \mathbf{d} \times \mathbf{e}_i$. Here $\mathbf{d} = \tau\mathbf{e}_1 + \kappa\mathbf{e}_3$ is the so-called Darboux vector, where τ denotes the torsion. The Darboux vector is the direction vector of the osculating cone Γ. This means that the angle ϕ between cone axis and ruling satisfies $\cot\phi = \tau/\kappa =: k$, a value which is called *conical curvature* of the developable surface at the ruling.

The Gauss image of a rotational cone Γ is a circle C on S^2 which becomes a great circle if Γ is a cylinder and degenerates to a point for a plane Γ. So all 2nd order local approximations addressed above have a *planar Gauss image* curve C. However, a planar Gauss image C of a surface Γ does not yet imply that Γ is a cone, while Γ must be a cylinder if C is a great circle and a plane if C is just a point. So let us discuss the case of a small circle C as Gauss image of a surface. These surfaces are well studied in classical differential geometry and known as *surfaces of constant slope*. They are the tangent surfaces of curves c of constant slope. Their tangents form a constant angle with a certain direction in space, which is obviously the rotational axis of the circle C. For a detailed study of these surfaces, we refer to [Pottmann & Wallner, 2001, Section 6.3]. The increased degrees of freedom compared to the osculating cone allow us to increase the local approximation of an arbitrary developable surface by one with a planar Gauss image:

Theorem 2.2.2. *At each regular point p of a developable ruled surface S, there is a developable surface Γ with a planar Gauss image, which has second order contact with S along the entire ruling through p and interpolates a curve $a \subset S$ through p.*

Proof. We omit the cases where S is a plane or a cylinder, since these surfaces already have a planar Gauss image curve. So we are left with cones and tangent surfaces S. We pick the osculating cone Γ_p of S along the ruling r_p through p and intersect S with the plane A through p which is orthogonal to the axis of Γ_p. This yields the curve a. Note that the plane A intersects the cone Γ_p in a circle, which is the osculating circle of a at p. The construction of the developable surface Γ proceeds as follows: Through each tangent of a we compute the two planes which form the same angle with the axis of Γ_p as Γ_p does. Among these two planes, we select the one which is closer to the corresponding tangent plane of S. Then, the envelope of this family of planes is the desired developable surface Γ with a planar Gauss image described in the theorem. By construction, Γ and S share the osculating cone Γ_p and thus have second order contact along the ruling through p. We could choose another curve $a \subset S$ which lies transversal to the rulings of S, but leave it with this special choice as it simplifies the further analysis.

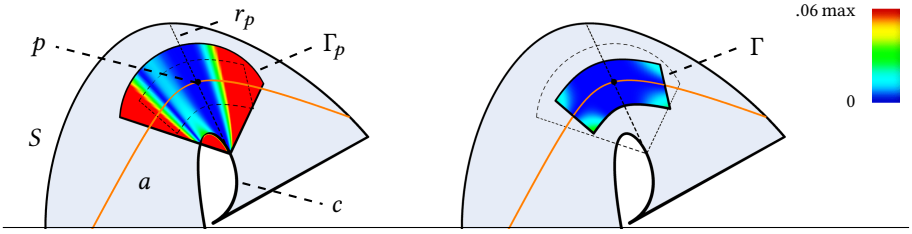


Figure 2.1: Local approximations of a developable surface S, which is the tangent surface of a space curve c. Left: The osculating cone Γ_p at a point $p \in S$ approximates S to 2nd order along the entire ruling r_p. Right: A developable surface Γ as in Theorem 2.2.2 approximates S even better, as is seen from the color coding of Γ and Γ_p according to their orthogonal distance to S.

For that, we use a local (x, y, z) coordinate system with $A : z = 0$ and describe the curve a by its support function $h(u)$. This means that we view a as envelope of its tangent lines

$$L(u) : \; x \cos u + y \sin u + h(u) = 0,$$

which form the angle u with the y-axis and possess the signed distance $h(u)$ from the origin (if the positive side of L is determined by the normal vector $(\cos u, \sin u)$). The derivative with respect to u is the curve normal, $\dot{L}(u) : \; -x \sin u + y \cos u + \dot{h}(u) = 0$. Intersecting the two lines $L, \dot{L}$, we obtain a parameterization of the curve a as

$$\mathbf{a}(u) : \; x = -h \cos u + \dot{h} \sin u, \;\; y = -h \sin u - \dot{h} \cos u.$$

Differentiating again yields the curvature centers (evolute) of $\mathbf{a}(u)$ as $\mathbf{a}^*(u) = \dot{L} \cap \ddot{L}$,

$$\mathbf{a}^*(u) : \; x = \dot{h} \sin u + \ddot{h} \cos u, \;\; y = -\dot{h} \cos u + \ddot{h} \sin u.$$

Thus, the signed curvature radius of $\mathbf{a}(u)$ is $\rho(u) = h(u) + \ddot{h}(u)$.

Let p be the point $\mathbf{a}(0) = (-h(0), -\dot{h}(0), 0)$. To shorten notation, we use the notation $h(0) =: h_0$ and likewise for the derivatives. Then the z-parallel line through the curvature center $\mathbf{a}^*(0) = (\ddot{h}_0, -\dot{h}_0, 0)$ is the axis of the osculating cone Γ_p. With k as conical curvature of Γ_p and of S at $u = 0$, the vertex of Γ_p has z-coordinate $z = (h_0 + \ddot{h}_0)/k = \rho_0/k$. Planes $P(u)$ through the tangents of $\mathbf{a}$ and with the same inclination against the z-axis as Γ_p have the equations

$$P(u) : \; x \cos u + y \sin u - kz + h(u) = 0. \tag{2.1}$$

Their envelope is the desired approximation Γ of S at p with a planar Gauss image and through $\mathbf{a}$. Differentiating with respect to u yields planes $\dot{P}, \ddot{P}$ whose equations agree with those of $\dot{L}, \ddot{L}$ and are therefore z-parallel planes through these lines. Recall that rulings of Γ are obtained as intersections $P \cap \dot{P}$ and the regression curve is found as $P \cap \dot{P} \cap \ddot{P}$. As discussed in more detail in [Pottmann & Wallner, 2001, Section 6.3], the regression curve of Γ lies in the z-parallel cylinder through $\mathbf{a}^*$ and the intersections of Γ with planes $z = const$ are translated offsets of $\mathbf{a}$. The intersection curve $\mathbf{a}_1$ of Γ with the plane $z = 1$ is a translated version of the offset of $\mathbf{a}$ at distance k and therefore has a support function $h(u) - k$. The ruling vectors $\mathbf{r}_1 = \mathbf{a}_1 - \mathbf{a}$ of Γ are $\mathbf{r}_1(u) = (k \cos u, k \sin u, 1)$.

The intersection curve $\bar{\mathbf{a}}$ of S with $z = 1$ has a support function $\bar{h}(u) = h(u) - k + f(u)$. Due to the 2nd order contact at $u = 0$, we have $f(0) = \dot{f}(0) = \ddot{f}(0) = 0$. Then, the tangent planes of S are

$$T(u) : \; x \cos u + y \sin u + (f(u) - k)z + h(u) = 0, \tag{2.2}$$

and the ruling vectors of S are $\mathbf{r} = \bar{\mathbf{a}} - \mathbf{a}$,

$$\mathbf{r}(u) = ((k - f) \cos u + \dot{f} \sin u, (k - f) \sin u - \dot{f} \cos u, 1).$$

Now we have parameterizations of S as $\mathbf{s}(u, v) = \mathbf{a}(u) + v\mathbf{r}(u)$ and of Γ as $\mathbf{g}(u, v) = \mathbf{a}(u) + v\mathbf{r}_1(u)$, which concludes the proof. □

However, we want to go beyond that and estimate the distance between S and its approximation Γ, and compare it to the distance between S and the osculating cone Γ_p.

We over-estimate the distances by measuring them in planes $z = const = v$ and there between points with parallel tangents. This means that we measure distances between points of the two surfaces which have the same parameter values (u, v). This distance $\delta(u, v)$ between S and Γ is given by

$$\delta(u, v) = |v| \|\mathbf{r}_1(u) - \mathbf{r}(u)\| = |v| \sqrt{f(u)^2 + \dot{f}(u)^2}. \tag{2.3}$$

We can also look at distances $\bar{\delta}$ between the parallel tangents directly, which are in view of equations (2.1) and (2.2),

$$\bar{\delta}(u, v) = |v f(u)|.$$

For $u = 0$ we get the ruling r_p through p and of course $\delta, \bar{\delta} = 0$.

Let us compare this with the approximation of S by the osculating cone Γ_p. The cone is given by (2.1) where h is replaced by the support function h_c of the osculating circle $\mathbf{c}_o$ of $\mathbf{a}$ at $p = \mathbf{a}(0)$,

$$h_c(u) = \rho_0 + \dot{h}_0 \sin u - \ddot{h}_0 \cos u.$$

The parameterization of the osculating circle is

$$\mathbf{c}_o(u) = (\ddot{h}_0 - \rho_0 \cos u, -\dot{h}_0 - \rho_0 \sin u, 0).$$

Thus, a parameterization of Γ_p is given by $\mathbf{c}_o(u) + v\mathbf{r}_1(u)$, and the two errors $\delta_p, \bar{\delta}_p$ between S and Γ_p become

$$\delta_p(u, v) = \|\mathbf{c}_o(u) - \mathbf{a}(u) + v(\mathbf{r}_1(u) - \mathbf{r}(u))\|, \; \bar{\delta}_p(u, v) = |v f(u) + h(u) - h_c(u)|.$$

To get better insight into the behavior of the errors, we insert Taylor expansions at $u = 0$,

$$f(u) = a_3u^3 + \ldots,\ h(u) = h_0 + \dot{h}_0u + \frac{\ddot{h}_0}{2}u^2 + \frac{\dddot{h}_0}{3}u^3 + \ldots.$$

The error vector between $\mathbf{a}$ and $\mathbf{c}_0$ now reads

$$\mathbf{c}_o(u) - \mathbf{a}(u) = (-\frac{\dot{\rho}_0}{3}u^3 + \ldots, \frac{\dot{\rho}_0}{2}u^2 + \frac{\ddot{h}_0}{6}u^3 + \ldots, 0).$$

Note that the quadratic term in the error vector is in tangential direction at p, and thus confirms the 2nd order contact between $\mathbf{c}_o(u)$ and $\mathbf{a}(u)$ at p. For the errors, we find the following expansions,

$$\delta(u,v) = |3a_3u^2v + \ldots|,\ \bar{\delta}(u,v) = |a_3u^3v + \ldots|,$$

and

$$\delta_p(u,v) = |\frac{\dot{\rho}_0}{2}u^2 + 3a_3u^2v + \ldots|,\ \bar{\delta}_p(u,v) = |\frac{\dot{\rho}_0}{6}u^3 + a_3u^3v + \ldots|.$$

As expected, the approximation of S by the osculating cone Γ_p is not as good as with Γ, since the deviation in the base plane $z = 0$ ($v = 0$) adds to the error everywhere. The appearance of the derivative $\dot{\rho}_0$ of the curvature radius $\rho(u)$ at $u = 0$ in the lowest order term is no surprise, as for $\dot{\rho}_0 = 0$ the osculating circle $\mathbf{c}_o$ has 3rd order contact with $\mathbf{a}$ and S at p.

There is one exception which we did not cover here, namely if the ruling r_p through p is an *inflection ruling*. In that case, Γ_p degenerates to the tangent plane, and one cannot parameterize directly via the tangent directional angle u. Instead, one can use another parameter t, and work with a parameterization in support coordinates $(u(t), h(t))$, as in [Pottmann & Wallner, 2001, pp. 362-363].

Knowing that surfaces with a planar Gauss image approximate developable surfaces at each point so well, we can increase developability by enforcing local approximations of this type through an optimization algorithm (see section 2.3).

2.2.2 Surfaces with a thin Gauss image

Our method will try to make the Gauss image of a B-spline surface thinner. After that, it will lie in a region R_ε on the sphere which has at most geodesic distance ε to a curve $C \subset S^2$. Let us briefly discuss the implications on a surface S which has a Gauss image in such an ε-strip R_ε. For that, we pick a part of the surface without an umbilic; there the principal curvature lines form a quadrilateral curve network without singularities. For simplicity, let us just consider a patch $\mathcal{P} \subset S$ in this region which is bounded by four principal curvature lines and does not contain parabolic points. Moreover, we select a square-like patch $\mathcal{P}$, meaning that the average length of the two pairs of opposite boundary curves is the same. The Gauss image $\sigma(\mathcal{P})$ of that principal patch $\mathcal{P}$ is a principal patch on S; corresponding curves on P and $\sigma(\mathcal{P})$ have parallel tangents at corresponding points, as they are principal directions and thus eigendirections of the derivative of the Gauss map. As we exclude parabolic points in $\mathcal{P}$, the Gauss map is regular everywhere and thus locally injective.

The Gauss image $\sigma(\mathcal{P})$ of $\mathcal{P}$ is squeezed into the thin region R_ε. Being contained in R_ε, at least one family F_1 of principal curvature lines on $\mathcal{P}$ must be mapped to very short curves in R_ε. If this is not true for the other family F_2 of principal curvature lines; the Gauss image curves of that family must be nearly parallel to the central

curve C of R_ε. Thus, the Gauss images of curves in F_1 will be nearly orthogonal to C (see Figures 2.11, 2.12). Their length can be bounded depending on the width variation of $\sigma(\mathcal{P})$. The shortening of curves in F_1 through the Gauss map to a length $\approx \varepsilon$ implies that the curves themselves will be close to straight lines. A surface with one family of straight principal curvature lines is exactly developable; our surface is only an approximation of that. A more thorough investigation of the geometric implications of a thin Gauss is left for future research.

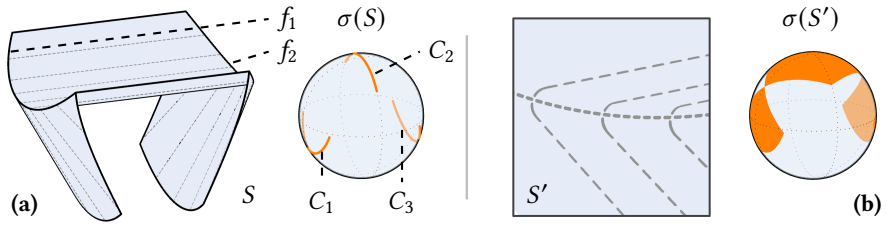


Figure 2.2: (a) Example of a developable shape S with curved folds f_1, f_2, and its Gauss image $\sigma(S) = C_1 \cup C_2 \cup C_3$. **(b)** Rounding the fold curves of S, leads to shape S' with Gauss image $\sigma(S')$ which is not thin.

Due to our focus on architectural geometry, we can exclude surfaces with wrinkles or folds appearing for example in cloth. These wrinkles are close to curves formed by parabolic points and have one very high principal curvature. They are not of interest in the present work, and are not characterized by thin Gauss images. Some insight into the geometry of these folds can be obtained as follows: Consider a planar sheet of material, mark a fold curve on it and bend it into a 3D shape S, leading to a developable surface with a curved crease (for the local geometry of such curved folds, see e.g. [Pottmann & Wallner, 2001, Section 6.5]. The two developable surfaces on either side of the fold curve f have curves C_1, C_2 as Gauss images. Now let us add a thin smooth blend to round off the fold curve f. The Gauss image of that blend surface will connect the two curves C_1, C_2 to a region which needs not be thin at all. With a sufficiently small blending radius the shape S can be arbitrarily close to an exact developable surface and thus be nearly developable, but the Gauss image will not be thin (see Figure 2.2).

Therefore, our approach of thinning the Gauss image implies the construction of nearly developable surfaces, but the converse is not true. A nearly developable surface needs not have a thin Gauss image, due to the phenomenon of wrinkles. For materials which allow only very little stretching, these wrinkles appear to be smoothed versions of developable surfaces with curved folds, as indicated above. There is interesting research on this phenomenon, combining geometry and physics; see e.g. [Cerda et al., 2004]. However, we are not aware of any differential geometric characterization of nearly developable surfaces which does not use the planar unfolding.

2.2.3 Developable bicubic surfaces

We will use bicubic B-spline surfaces and thus it is appropriate to justify this choice. When it comes to modeling nearly developable surfaces, our choice is natural due to the approximation power of splines. The condition of one family of nearly straight principal curvature lines is sufficiently soft to be modeled nicely with these splines.

However, especially in our architectural application, we will model panel arrangements also by bicubic B-spline surfaces, with knots of multiplicity three, which are just C^0 patchworks of bicubic polynomial patches. We want these polynomial patches to be close to developable surfaces, in particular to right circular cones or cylinders. Thus, we briefly discuss *developable bicubic surfaces*.

Bicubic patches on tangent surfaces. The tangent surface of a polynomial cubic $\mathbf{c}(u)$ can be parameterized as

$$\mathbf{x}(u,v) = \mathbf{c}(u) + v\dot{\mathbf{c}}(u),$$

and it is therefore a bicubic surface. In this form, the rulings are v-isoparameter curves and an axis aligned rectangle in the parameter domain represents a patch on the surface bounded by two rulings. There are other bicubic patches on that surface, which are obtained as images of arbitrary parallelograms in the (u,v)-plane. Equivalently, one can obtain them as images of the unit square $[0,1]^2$ in a $(\bar{u},\bar{v})$ parameter plane via an affine parameter change,

$$u = a_0 + a_1\bar{u} + a_2\bar{v},\ v = b_0 + b_1\bar{u} + b_2\bar{v}.$$

Furthermore, special bilinear re-parameterizations where the first equation remains and the second one reads

$$v = b_0 + b_1\bar{u} + b_2\bar{v} + b_3\bar{u}\bar{v},$$

also yield bicubic patches on that tangent surface.

Even *the tangent surface of a polynomial quartic* $\mathbf{c}(u)$ *has a bicubic parameterization.* We write $\mathbf{c} = \mathbf{a}_4u^4 + \mathbf{a}_3u^3 + \ldots + \mathbf{a}_0$ in monomial form and parameterize its tangent surface as

$$\mathbf{x}(u,v) = \mathbf{c}(u) + (-u/4 + v)\dot{\mathbf{c}}(u),$$

which is a bicubic representation. A complete classification of all bicubic tangent surfaces is an open problem. For our purposes it suffices to see that tangent surfaces of quartic curves are included in this class of surfaces, which leaves sufficient flexibility for modeling.

Bicubic patches on cones and cylinders. A cone with vertex $\mathbf{v}$ can be written as $\mathbf{x}(u,v) = \mathbf{v} + f(u,v)\mathbf{c}(u)$. To get a bicubic parameterization, we can use a cubic curve $\mathbf{c}(u)$ and a cubic polynomial $f(u,v) = g(v)$ or a quadratic curve (parabola) $\mathbf{c}(u)$ and a function $f(u,v)$ of bi-degree $(1,3)$. In the former case, the cone is in general a cubic surface, while in the latter case one parameterizes quadratic cones.

A cylinder $\mathbf{x}(u,v) = \mathbf{a}(u) + f(u,v)\mathbf{r}$, with a ruling direction $\mathbf{r}$, has a bicubic representation when $\mathbf{a}(u)$ is at most cubic and f any bicubic function.

Developable bicubic patches with a planar Gauss image. This class of surfaces includes all bicubic cylinders. Among the cones, only rotational cones are possible. We can generate them from the special cone $x^2 + y^2 = z^2$, and then apply uniform scaling in z-direction and a rigid body motion. The special cone is parameterized by a Pythagorean triple of bicubic functions $x(u,v), y(u,v), z(u,v)$ of the form

$$x(u,v) = 2abw,\ y(u,v) = (a^2 - b^2)w,\ z(u,v) = (a^2 + b^2)w,$$

where $a(u,v)$, $b(u,v)$, $w(u,v)$ are bilinear functions. Bicubic tangent surfaces with a planar Gauss image have a regression curve $\mathbf{c}(u)$ of constant slope. It follows from our considerations above that the tangent surface of a polynomial curve $\mathbf{c}(u)$ of constant slope and degree ≤ 4 is such a surface. These curves $\mathbf{c}(u)$ are exactly the *spatial Pythagorean hodograph curves* of degree ≤ 4. For their generation and degrees of freedom, we point to the monograph by Farouki [2008, Chapter 21].

We have already mentioned rotational cones and note that rotational cylinders do not possess an exact bicubic parameterization. This is due to the fact that a rotational cylinder cannot carry a polynomial curve transversal to the rulings as it would project onto a circle. While a circle does not have an exact polynomial parameterization, it is possible to achieve good approximations with cubics (see [Vavpetič & Žagar, 2019] and the references therein). This is sufficient for our purposes.

Developable B-spline surfaces. If two algebraic developable surface patches meet with C^1 continuity at a common curve (different from a ruling), their set of tangent planes agrees there. Due to the algebraic nature, agreement of the set of tangent planes along a curve segment is sufficient for the agreement of the set of tangent planes everywhere and for agreement of the two algebraic surfaces. Therefore, any developable B-spline surface with C^1 continuity represents a single polynomial developable surface, unless the patches are joined along rulings. This latter case is used in [Tang et al., 2016]. The former case is useful to represent appropriate trimmed patches on polynomial developable surfaces, but not for increasing the flexibility in modeling the surfaces themselves.

A regular bicubic surface $\mathbf{S}$ parameterized by parameters u, v is developable when the Gaussian curvature vanishes at every point $(u,v) \in D$ of the surface. Based on this definition of developable surfaces, we can compute the algebraic complexity of the developability property for $\mathbf{S}$. Since the Gaussian curvature is the ratio of the determinants of the second and first fundamental forms, it is sufficient for the following equation to hold

$$\det(\mathbf{II}) = 0 \Leftrightarrow [\mathbf{S}_{uu}, \mathbf{S}_u, \mathbf{S}_v][\mathbf{S}_{vv}, \mathbf{S}_u, \mathbf{S}_v] - [\mathbf{S}_{uv}, \mathbf{S}_u, \mathbf{S}_v]^2 = 0, \quad \forall (u,v) \in D$$

where $[\mathbf{a}, \mathbf{b}, \mathbf{c}]$ denotes the triple product of vectors $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c} \in \mathbb{R}^3$. Expanding and grouping with respect to monomials in parameters u, v we get a polynomial $f \in \mathbb{R}[x_{00}, y_{00}, z_{00}, \ldots, x_{33}, y_{33}, z_{33}][u,v]$, where $(x_{ij}, y_{ij}, z_{ij}) \in \mathbb{R}^3$, are the coordinates of control point $\mathbf{P}_{i,j}$ of surface $\mathbf{S}$. Following this grouping, we count that polynomial f has 191 coefficients $g_k \in \mathbb{R}[x_{00}, y_{00}, z_{00}, \ldots, x_{33}, y_{33}, z_{33}]$, where $k = 1, \ldots, 191$.

The requirement that polynomial f vanishes for all values $(u,v) \in D$ is satisfied if f is identically the zero polynomial, or equivalently all coefficient polynomials g_k vanish. This means that, if we need to guarantee these conditions precisely by evaluating f at different points on the surface, we would require a minimum of 191 points in a general position, namely points that would generate linearly independent combinations of g_k . In practice, since $\deg_u(f) = \deg_v(f) = 13$ we would define a 14×14 regular grid over D to acquire 196 evaluation points.

Alternatively, we can examine the algebraic variety $V(I)$ of the ideal $I = \langle g_1, \ldots, g_{191} \rangle$ generated by the coefficient polynomials g_k. Again, these are 191 homogeneous polynomials in 48 variables with $\deg(g_k) = 6$. Computing a reduced Gröbner basis in an attempt to work with a minimal number of generators $h_m \in \mathbb{R}[x_{00}, y_{00}, z_{00}, \ldots, x_{33}, y_{33}, z_{33}]$, with $m \leq 191$, for the ideal I is computationally expensive, and is expected to produce generators that have increasingly higher degrees [Dubé, 1990].

These observations only demonstrate that if we wish to increase interactivity in the design process with developable surfaces, we need to avoid the computational complexity of exact satisfiability and instead sufficiently approximate the developability property in an efficient way.

2.3 Increasing developability

Motivated by Theorem 2.2.2, we can try to increase the developability of a surface S by ensuring that the Gauss images of well chosen regions on S are nearly planar. Using this basic idea, we now discuss the details of an optimization algorithm which iteratively deforms a bicubic B-spline surface towards a nearly developable one.

2.3.1 Optimization setup

Surface. Let us consider a bicubic B-spline surface $\mathbf{S} : \mathbb{R}^2 \to \mathbb{R}^3$,

$$\mathbf{S}(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_{i,3}(u) B_{j,3}(v) \mathbf{P}_{i,j}, \tag{2.4}$$

where $u, v \in [0,1]$ and $B_{i,3}(u)$, $B_{j,3}(v)$ are cubic B-spline basis functions defined on uniform knot sequences in both directions. $\{\mathbf{P}_{i,j}\} \in \mathbb{R}^3$ are the control points of the surface $\mathbf{S}$, where $0 \le i \le n$, $0 \le i \le m$ and $n, m \ge 3$. For more information on B-spline surfaces and NURBS surfaces in general, we direct the reader to [Piegl & Tiller, 1997, Section 4.4].

Surface $\mathbf{S}$ serves as the central object of study in this work. A generic surface of the above form is non-developable and we aim to increase its developability by modifying the coordinates of its control points in a "minimal" way that will be defined in the following sections.

We point out that surface $\mathbf{S}$ could be defined as any NURBS surface as long as the weights of the control points and the knot vectors are fixed and are not considered variables in the optimization process. This simplifies and accelerates the optimization procedure while not sacrificing the quality of our results in the sense that B-spline surfaces are adequate approximations of more general NURBS surfaces. For readability, we define $\mathbf{S}$ as an elementary B-spline surface while keeping in mind that the following applies to more general surfaces.

Sampling the surface. We begin by sampling $\mathbf{S}$, the surface that is to be optimized, at a set of evaluation points $\{\mathbf{p}_k\} \subset \mathbb{R}^3$, which we will call *sample points*.

The approach we took for the sampling was to uniformly sample the parameter space, motivated by the fact that convoluted areas on the surface $\mathbf{S}$, i.e. areas where the control points are concentrated and finer features emerge, would be represented by more evaluation points inherently. We set the number of sample points L_u, L_v along the u, v directions respectively and get a gridded pattern of points $(u,v) \in [0, \frac{1}{L_u+1}, \ldots, 1] \times [0, \frac{1}{L_v+1}, \ldots, 1]$ on the parameter space, which in turn results in the set of required sample points $\{\mathbf{p}_k\}$ on the surface $\mathbf{S}$.

The evaluation of points $\mathbf{p}_k$ is given by formula 2.4, which is linear in the coordinates of the control points with constant coefficients. In practice, these coefficients are precomputed per point and stored. Whenever the control points are updated by the optimization process or user input, we re-evaluate the position of the sample points using the stored coefficients.

Grouping into patches. Next, we consider overlapping neighborhoods on the surface, that we will call *patches*, and that are represented as sets of sample points U_j. We construct the patches in such a way that neighboring patches will have non-empty intersections, i.e. there exists at least one sample point that belongs to both patches. The importance of this property will become clear in a later section.

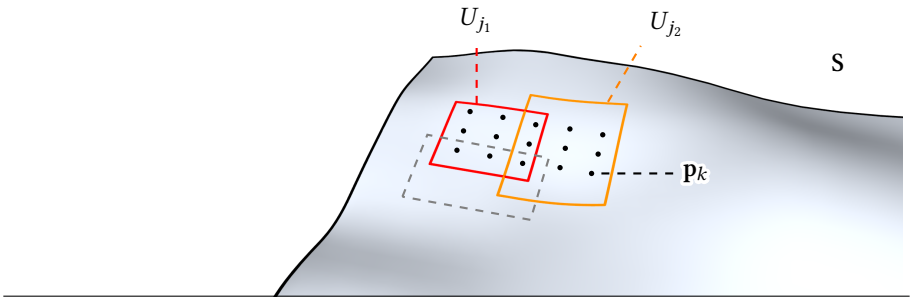


Figure 2.3: Surface $\mathbf{S}$ is sampled at various evaluation points $\mathbf{p}_k$. The sample points are then grouped to overlapping groups. An example of such a grouping are groups U_{j_1} and U_{j_2}.

By uniformly sampling the parameter space we also simplify the process of grouping the sample points. The patches on the surface, as already mentioned, are represented by sets of sample points. By using the grid of points on the parameter space we can determine the patches just by setting the number of sample points in each of the u, v directions that a patch will contain and the number of sample points that will belong in the overlap region for each of the u, v directions. Figure 2.3 focuses on two such patches as an example of a simple grouping.

Normal computation. We associate each sample point $\mathbf{p}_k$ with the unit normal $\mathbf{n}_k$ of the surface at that point. The unit normals define the *Gauss map* σ of the surface. We compute the unit normal $\mathbf{n}_k$ of the surface point $\mathbf{p}_k$ as

$$\mathbf{n}_k := \sigma(\mathbf{p}_k) = \frac{\mathbf{S}_u \times \mathbf{S}_v}{\|\mathbf{S}_u \times \mathbf{S}_v\|},$$

where $\mathbf{S}_u$, $\mathbf{S}_v$ are the partial derivatives of $\mathbf{S}$ with respect to u and v. Note that $\mathbf{S}_u$ and $\mathbf{S}_v$,

$$\mathbf{S}_u(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,3}^{(1)}(u)B_{j,3}(v)\mathbf{P}_{i,j},\ \mathbf{S}_v(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,3}(u)B_{j,3}^{(1)}(v)\mathbf{P}_{i,j},$$

are linear combinations of the control points with coefficients which we precompute and store to accelerate future computations [Piegl & Tiller, 1997, Section 1.5].

Gauss map of a patch. For every patch U_j, we denote by N_j the Gauss image of U_j, i.e. the set of unit normals $\mathbf{n}_k$ corresponding to the sample points $\mathbf{p}_k \in U_j$,

$$N_j = \sigma(U_j) = \sigma(\{\mathbf{p}_k\}) = \{\mathbf{n}_k\}.$$

We associate each patch U_j with a plane $H_j \subset \mathbb{R}^3$ with equation $\mathbf{v}_j \cdot \mathbf{x} + d_j = 0$. Here, $\mathbf{v}_j$ is a unit normal vector of H_j and d_j is the distance of H_j from the origin. H_j serves

as the target plane for N_j. By optimization, we will enforce all normal vectors in N_j to lie on H_j and thus aim at a planar Gauss image of patch U_j.

2.3.2 Initialization

The variables of the optimization are the coordinates of the control points $\mathbf{P}_{i,j}$ and the cutting planes H_j that define the Gauss image circles per patch U_j. In this section, we describe the initialization step of the optimization process.

Control points. We assume that we always have an initial state for the surface that is either user defined or is provided by other means. We initialize the control point coordinates with the values from this initial configuration. Those in turn will be used to initialize H_j for every patch.

Cutting planes. We want to optimize for planarity of the Gauss image N_j of each patch U_j and thus associate with each patch U_j a target plane H_j for N_j. Initializing the target plane H_j for each patch with the best fitting plane to points $\mathbf{n}_k \in S^2$ works in the case that U_j is a developable patch. However, this method does not produce the desired results if the patch is non-developable, as seen in Figure 2.4. To overcome this, we use the following approach.

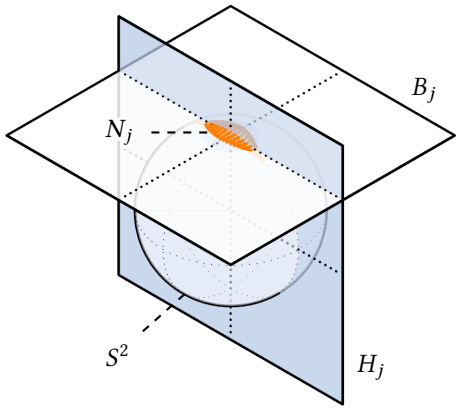


Figure 2.4: Consider the Gauss image N_j of a group U_j. Plane B_j is the best fitting plane to N_j, in the sense that it minimizes the sum of squared distances of points N_j to the plane, and is considered an undesired initialization. Using B_j as a target plane for the points in N_j will degenerate the Gauss image to a single point, meaning patch U_j will be flat. Alternatively, plane H_j is the resulting plane from optimization problem 1 and captures the overall main principal direction of patch U_j. Plane H_j is a better initial target plane, since it will not necessarily lead to a 0-dimensional Gauss image.

Consider the main principal direction $\mathbf{q}_k \in \mathbb{R}^3$ of surface $\mathbf{S}$ at point $\mathbf{p}_k$, i.e. the principal direction corresponding to the principal curvature with the maximum absolute value, that is $\max\{|\kappa_1(\mathbf{p}_k)|, |\kappa_2(\mathbf{p}_k)|\}$ where $\kappa_i : \mathbf{S} \to \mathbb{R}$, $i = 1, 2$, are the principal curvatures of a point on $\mathbf{S}$. The *principal curvatures* and *principal directions* of a surface at a point on the surface are the eigenvalues and corresponding

eigenvectors of the *shape operator* $-d_v\mathbf{N} = -\mathbf{I}^{-1}\mathbf{II}$, where $\mathbf{I}$, $\mathbf{II}$ are the *first* and *second fundamental forms* of the surface. We denote by Q_j the set of main principal directions $\mathbf{q}_k$ corresponding to the points $\mathbf{p}_k \in U_j$.

We initialize H_j as the plane passing through the barycenter of N_j with unit normal in the direction of the vector which is "as orthogonal as possible" to the set Q_j of main principal directions. Intuitively, we wish the initial cutting plane to intersect the sphere at a circle whose tangent at every point $\mathbf{c} \in S^2 \cap H_j$ is "as parallel as possible" to the main principal directions of the sample points corresponding to the unit normals around $\mathbf{c}$.

In this way, the cutting plane serves as a generalized main principal plane, or a plane containing the main principal directions of every sample point in the patch. For a patch that is non-developable, we wish to initialize this main principal plane by using the main principal directions of the sample points weighted by a measure of confidence. A low weight indicates the difficulty in distinguishing between the two principal curvatures. Specifically, we introduce weight $w_k \in [0, 1]$ corresponding to each sample point $\mathbf{p}_k$ as

$$w_k = 1 - \frac{\min\{|\kappa_i(\mathbf{p}_k)|\}}{\max\{|\kappa_i(\mathbf{p}_k)|\}}, \quad i = 1, 2 \tag{2.5}$$

Now, for each patch U_j we need to solve the following optimization problem.

Optimization problem 1 Plane initialization

$$\begin{aligned} &\text{minimize} && \sum_{\mathbf{q}_k \in Q_j} w_k(\mathbf{v}_j \cdot \mathbf{q}_k)^2 \\ &\text{subject to} && \mathbf{v}_j^2 = 1 \end{aligned}$$

Optimization problem 1 is a special case of minimizing a quadratic form under a quadratic regularization constraint. Bringing the objective function into the form $\mathbf{v}_j^\top \mathbf{Q}\mathbf{v}_j$, the minimizer $\mathbf{v}_j^*$ is the normalized eigenvector corresponding to the smallest eigenvalue of $\mathbf{Q}$. Then, plane H_j is given by $\mathbf{v}_j^* \cdot \mathbf{x} + d_j = 0$, with

$$d_j = -\mathbf{v}_j^* \cdot \frac{1}{|N_j|} \sum_{\mathbf{n_k} \in N_j} \mathbf{n}_k,$$

where $|N_j|$ is the cardinality of N_j.

2.3.3 Problem formulation

We are now ready to formulate the central optimization problem of this chapter by defining the relevant individual energy functionals.

Developability energy. We formulate the desired property of each patch to have a planar Gauss image by introducing an appropriate energy term $\mathcal{E}_\mathrm{d}$. This energy term measures per patch the total sum of distances of the normals $\mathbf{n}_k \in S^2$ to the target patch plane, that is the quantity

$$\sum_j \sum_{\mathbf{n}_k \in N_j} (\mathbf{n}_k \cdot \mathbf{v}_j + d_j)^2, \tag{2.6}$$

where j is the indexing of the patches and $\mathbf{v}_j$, d_j are unit normal and distance from the origin of target plane H_j for patch U_j. To avoid trivial solutions, we introduce the following unit length constraint on the plane normals $\mathbf{v}_j$ in the form of an additional energy term,

$$\sum_j (\mathbf{v}_j^2 - 1)^2.$$

Additionally, the surface normals $\mathbf{n}_k$ are computed as

$$\mathbf{n}_k^{(m)} = \frac{\mathbf{S}_u^{(m)} \times \mathbf{S}_v^{(m)}}{\|\mathbf{S}_u^{(m-1)} \times \mathbf{S}_v^{(m-1)}\|},$$

where $a^{(m)}$ denotes the value of variable a at iteration step m in our iterative optimization process. We use the constant norm $\|\mathbf{S}_u^{(m-1)} \times \mathbf{S}_v^{(m-1)}\|$ from the previous iteration when normalizing the current vector $\mathbf{S}_u^{(m)} \times \mathbf{S}_v^{(m)}$ for the computation of the surface normal $\mathbf{n}_k$. This is standard practice to ensure that the objective function is polynomial.

All the above lead to an energy term of the form

$$\mathcal{E}_\mathrm{d} = \sum_j \sum_{\mathbf{n}_k \in N_j} (\mathbf{n}_k \cdot \mathbf{v}_j + d_j)^2 + \lambda_1 \sum_j (\mathbf{v}_j^2 - 1)^2, \tag{2.7}$$

where λ_1 is an appropriate weight for the unit length constraint.

The importance of having patches that are overlapping, or equivalently neighboring patches containing common sample points, becomes evident at this point. Each patch is optimized to have a Gauss image which is a subset of a spherical curve. This can have a competitive effect between patches that are adjacent due to diverging target planes, and cause slow convergence. By having the patches share sample points, we introduce a diffusion factor to the optimization that ensures smoothness of the resulting Gauss image curve.

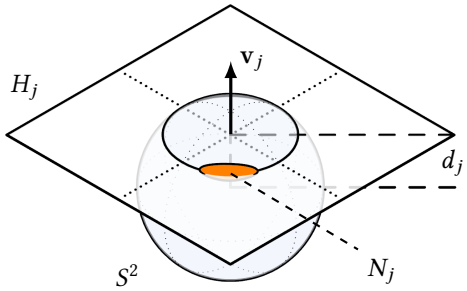


Figure 2.5: The Gauss image N_j of a single non-developable patch U_j is a 2-dimensional subset of S^2. The cutting plane H_j serves as the target plane for the normals $\mathbf{n}_k \in N_j$.

Soft constraints. We also introduce a set of additional energy terms to the main problem that constrain the output surface and aim to avoid degeneracies, produce more aesthetically pleasing results and give control to the user over the proximity of the resulting surface to a reference surface.

The energy term $\mathcal{E}_\mathrm{c}$ denotes a measure of the closeness of the resulting surface $\mathbf{S}$ to a reference surface $\mathbf{S}_\mathrm{ref}$, which can be either an arbitrary surface or the initial configuration of the design surface. The implementation we follow for the closeness energy term is based on the *tangential distance minimization* (TDM) [Pottmann et al., 2004; W. Wang et al., 2006]. The energy term is defined as the sum of squared distances of sample points to the tangent planes at their closest points on the reference surface. We use the already sampled points $\mathbf{p}_k \in \mathbf{S}$ and a set of sample points X from the reference surface $\mathbf{S}_\mathrm{ref}$. If the reference surface is the initial surface then $X = \{\mathbf{p}_k\}$; otherwise, X is an independent sampling. Then $\mathcal{E}_\mathrm{c}$ is defined as

$$\mathcal{E}_\mathrm{c} = \sum_k [(\mathbf{p}_k - \mathbf{x}_k) \cdot \mathbf{N}(\mathbf{x}_k)]^2, \tag{2.8}$$

where $\mathbf{x}_k$ is the closest point to $\mathbf{p}_k$ from the set of points X in the Euclidean metric, and $\mathbf{N}(\mathbf{x}_k)$ is the unit normal of $\mathbf{S}_\mathrm{ref}$ at point $\mathbf{x}_k$. At each iteration the closest point is updated. We utilize FLANN for the closest point query and refer to [Muja & Lowe, 2014] for the computational complexity.

A final fairness energy term $\mathcal{E}_\mathrm{f} = w_{\mathrm{f}_1}\mathcal{E}_{\mathrm{f}_1} + w_{\mathrm{f}_2}\mathcal{E}_{\mathrm{f}_2}$ is introduced to the objective function that avoids degeneracies in the resulting surface and is widely used in mesh optimization problems for the smoothing effect it provides. Specifically, we denote by $\mathcal{E}_{\mathrm{f}_1}$ the sum of squared norms of the first order differences of the control points in both grid directions, and by $\mathcal{E}_{\mathrm{f}_2}$ the second order equivalent, namely

$$\begin{aligned}
\mathcal{E}_{\mathrm{f}_1} &= \sum_{i,j} \left(\|\mathbf{P}_{i+1,j} - \mathbf{P}_{i,j}\|^2 + \|\mathbf{P}_{i,j+1} - \mathbf{P}_{i,j}\|^2 \right), \\
\mathcal{E}_{\mathrm{f}_2} &= \sum_{i,j} \left(\|\mathbf{P}_{i+1,j} - 2\mathbf{P}_{i,j} + \mathbf{P}_{i-1,j}\|^2 + \|\mathbf{P}_{i,j+1} - 2\mathbf{P}_{i,j} + \mathbf{P}_{i,j-1}\|^2 \right).
\end{aligned}$$

We assign $w_{\mathrm{f}_1} = 0$, $w_{\mathrm{f}_2} = 1$ in all the following applications unless stated otherwise.

Total energy. All energy terms $\mathcal{E}_\mathrm{d}$, $\mathcal{E}_\mathrm{c}$, $\mathcal{E}_\mathrm{f}$ are assigned weights w_d, w_c, w_f and collected in the total energy for developability optimization,

$$\mathcal{E} = w_\mathrm{d}\mathcal{E}_\mathrm{d} + w_\mathrm{c}\mathcal{E}_\mathrm{c} + w_\mathrm{f}\mathcal{E}_\mathrm{f}. \tag{2.9}$$

For details on the choice of weights, we refer to Section 2.5.

Increasing developability. Now our problem is reduced to the minimization of $\mathcal{E}$.

Optimization problem 2 Increasing developability

minimize $\quad \mathcal{E} = w_\mathrm{d}\mathcal{E}_\mathrm{d} + w_\mathrm{c}\mathcal{E}_\mathrm{c} + w_\mathrm{f}\mathcal{E}_\mathrm{f}$

The variables of $\mathcal{E}$ are the control points $\{\mathbf{P}_{i,j}\}$ of $\mathbf{S}$ and the patch planes H_j, defined by $\mathbf{v}_j$ and d_j. The optimization problem 2 is an unconstrained nonlinear least-squares problem. Any algorithm for nonlinear least-squares problem can be applied in our case. We follow the standard *Gauss-Newton method* in our implementation and experiments [Nocedal & Wright, 2006, Section 10.3].

2.4 Panelization

Motivated by applications in architecture, we consider the problem of approximating a given arbitrary surface by a C^0 continuous surface which consists of developable patches. As we optimize for developability with help of a planar Gauss image, the resulting surface patches include as important special cases rotational cylinders and rotational cones. We will particularly focus on the constraints which ensure that we obtain these special types of panels. Especially when working with glass, these rotational panels are preferred because there are special machines for their production. Figure 2.6 shows a recent example of an architectural freeform façade which has been constructed with mainly cylindrical glass panels to reduce manufacturing cost.

Figure 2.6: Side detail of *Nur Alem*, the main pavilion of the Astana EXPO 2017 Exhibition in Astana, Kazakhstan. Mostly cylindrical panels were used to rationalize the curved transparent freeform façade (different from the sphere).

2.4.1 Optimization setup

In this section, we will go through the differences between the central method that was presented in the previous sections and the variation for this new problem while introducing any new concepts that will be of use.

Surface. The main object of study in this section will be a surface S consisting of a grid of subsurfaces $\mathrm{S}^{(r)}$, with C^0 continuity at the inner boundaries. Specifically, S is a composite surface

$$\mathrm{S} = \bigcup_r \mathrm{S}^{(r)},$$

where r indexes the set of subsurfaces. Each $\mathrm{S}^{(r)}$ is a bicubic Bézier surface of the form

$$\mathrm{S}^{(r)}(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_{i,3}(u)B_{j,3}(v)\mathbf{P}_{i,j}^{(r)} \tag{2.10}$$

and will be referred to as a *panel* in the following. This configuration represents the paneling of a freeform surface. The C^0 continuity of neighboring panels is achieved by having common control points at the corresponding edges and models the connectivity and continuity found between distinct panels of a panelized surface. This allows for panelizations that are not in a grid configuration and can easily generalize to more complex surfaces of arbitrary topology just by appropriately "gluing" panels at their edges.

Sampling and grouping. We sample surface S at a collection of sample points $\{\mathbf{p}_k\}$ and group them to groups U_r, each corresponding to a single panel. The same follows for the corresponding surface normals N_r and the associated panel planes H_r.

We can therefore define the developability energy term per panel as

$$\mathcal{E}_{\mathrm{d}}^{(r)} = \sum_{\mathbf{n}_k \in N_r} (\mathbf{n}_k \cdot \mathbf{v}_r + d_r)^2 + \lambda_1 (\mathbf{v}_r^2 - 1)^2, \tag{2.11}$$

where λ_1 is an appropriate weight, and the developability energy term of surface S as

$$\mathcal{E}_{\mathrm{d}} = \sum_r \mathcal{E}_{\mathrm{d}}^{(r)}. \tag{2.12}$$

This modified grouping of the sample points $\{\mathbf{p}_k\}$ allows for the individual optimization of each panel, which will be studied in more detail in a following section.

Rotational panels. By introducing the additional constraint that the panel should be a rotational surface, we are optimizing for the panels to be either rotational cones or rotational cylinders. Rotational surfaces have the property that the surface normal lines are coplanar with the axis of rotation. Let L_1, L_2 be two lines in $\mathbb{R}^3$ with Plücker coordinates $(\mathbf{a}, \bar{\mathbf{a}})$, $(\mathbf{b}, \bar{\mathbf{b}}) \in \mathbb{R}^6$ respectively. The two lines are coplanar if their Plücker coordinates satisfy the condition

$$\mathbf{a} \cdot \bar{\mathbf{b}} + \bar{\mathbf{a}} \cdot \mathbf{b} = 0. \tag{2.13}$$

Recall that the Plücker coordinates $(\mathbf{a}, \bar{\mathbf{a}}) \in \mathbb{R}^6$ of a line $L \subset \mathbb{R}^3$ are given by the direction vector $\mathbf{a} \in \mathbb{R}^3$ and the moment vector $\bar{\mathbf{a}} = \mathbf{p} \times \mathbf{a} \in \mathbb{R}^3$, where $\mathbf{p} \in \mathbb{R}^3$ is a point on L. Obviously, these coordinates are not independent, but satisfy the Plücker condition $\mathbf{a} \cdot \bar{\mathbf{a}} = 0$. For more information on line geometry and relevant applications, we refer to the literature [Pottmann & Wallner, 2001, Section 2.1].

Consider now the Plücker coordinates $(\mathbf{n}_k, \bar{\mathbf{n}}_k) \in \mathbb{R}^6$ of the normal lines at the sample points of a panel U_r and of the unknown axis of rotation $(\mathbf{a}_r, \bar{\mathbf{a}}_r) \in \mathbb{R}^6$. The desired property that the panel is a rotational surface can be expressed as $\mathbf{a}_r \cdot \bar{\mathbf{n}}_k + \mathbf{n}_k \cdot \bar{\mathbf{a}}_r = 0 \; \forall \mathbf{n}_k \in N_r$. Thus, the problem of optimizing for rotational surface panels can be formulated as minimizing the energy

$$\sum_r \sum_{\mathbf{n}_k \in N_r} (\mathbf{a}_r \cdot \bar{\mathbf{n}}_k + \mathbf{n}_k \cdot \bar{\mathbf{a}}_r)^2, \tag{2.14}$$

under the constraint that $(\mathbf{a}_r, \bar{\mathbf{a}}_r)$ describe a line, i.e., satisfy the Plücker condition $\mathbf{a}_r \cdot \bar{\mathbf{a}}_r = 0$, and the unit length constraint $\mathbf{a}_r^2 = 1$ on the axis direction $\mathbf{a}_r$.

At this point, we focus on the fact that for a rotational panel $S^{(r)}$ with planar Gauss image, the normal $\mathbf{v}_r$ of the plane H_r containing the Gauss image and the

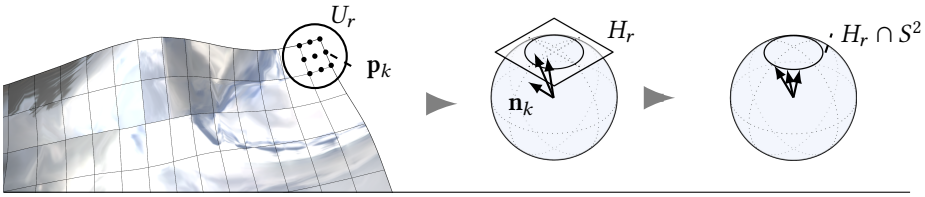


Figure 2.7: We focus on a single panel $\mathbf{S}^{(r)}$ of a panelized surface $\mathbf{S}$. We are optimizing for the endpoints of normals $\mathbf{n}_k$ corresponding to the sample points $\mathbf{p}_k \in U_r$ of panel $\mathbf{S}^{(r)}$ to lie on the same plane H_r.

direction of the rotation axis $\mathbf{a}_r$ coincide. Using this fact, we denote the Plücker coordinates of the rotation axis by $(\mathbf{v}_r, \bar{\mathbf{v}}_r)$.

By making this adaptation, we have covered the unit length constraint on the rotation axis direction by the corresponding constraint on the target plane normal in (2.11). The Plücker condition is added as an additional energy term with an appropriate weight λ_2. Considering all the above, the resulting rotationality energy term $\mathcal{E}_\mathrm{r}$ is of the form

$$\mathcal{E}_\mathrm{r} = \sum_r \sum_{\mathbf{n}_k \in N_r} (\mathbf{v}_r \cdot \bar{\mathbf{n}}_k + \mathbf{n}_k \cdot \bar{\mathbf{v}}_r)^2 + \lambda_2 \sum_r (\mathbf{v}_r \cdot \bar{\mathbf{v}}_r)^2. \tag{2.15}$$

While the Plücker coordinates of the normal lines are initialized in the optimization problem with their current values in the configuration of surface $\mathbf{S}$, the axis of rotation $(\mathbf{v}_r, \bar{\mathbf{v}}_r)$ of every panel U_r remains unknown at this point or, assuming the panels are in generic configuration, does not exist at all. An appropriate initialization for the Plücker coordinates of the axis of rotation of each panel is given by methods used in kinematic surface reconstruction applications, where the problem of fitting a velocity field to a set of surface normals is studied [Liu, Pottmann, & Wang, 2006; Pottmann & Randrup, 1998]. It follows the same thought process as the main idea behind the energy term (2.14). In fact, it is exactly the same energy that we aim to minimize but applied to each of the panels separately while considering the affine normal lines fixed. The resulting axis is the best fitting one in the least-squares sense. Formulating the above as an optimization problem leads us to the minimization of

$$\sum_{\mathbf{n}_k \in N_r} (\mathbf{v}_r \cdot \bar{\mathbf{n}}_k + \bar{\mathbf{v}}_r \cdot \mathbf{n}_k)^2. \tag{2.16}$$

We already have an appropriate initialization for the target plane normal $\mathbf{v}_r$, described in optimization problem 1. Thus, the objective function (2.16) is a quadratic function of the moment vector $\bar{\mathbf{v}}_r$. The latter is orthogonal to $\mathbf{v}_r$ and therefore can be expressed as

$$\bar{\mathbf{v}}_r = \mu_1 \mathbf{b}_1 + \mu_2 \mathbf{b}_2,$$

where $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^3$ form a basis of the plane perpendicular to $\mathbf{v}_r$. Substitution into (2.16) yields a quadratic function in μ_1, μ_2 and the optimal values of μ_1, μ_2 are the solutions of a linear system.

2.4.2 Problem formulation

Thus, the surface paneling problem is the following variation of the optimization problem 2, and is solved with the same approach.

Optimization problem 3 Surface paneling

$$\text{minimize} \quad \mathcal{E} = w_{\mathrm{d}}\mathcal{E}_{\mathrm{d}} + w_{\mathrm{r}}\mathcal{E}_{\mathrm{r}} + w_{\mathrm{c}}\mathcal{E}_{\mathrm{c}} + w_{\mathrm{f}}\mathcal{E}_{\mathrm{f}}$$

Individual panel treatment. Until now we have shown how to optimize the paneling of surface S in a global fashion. Since we defined the energy term $\mathcal{E}_{\mathrm{d}}^{(r)}$ per panel, this approach can be customized to consider each panel separately, achieving in the process increased control over the resulting panelization. We use the following obvious fact:

Lemma 2.4.1. *Let panel $\mathrm{S}^{(r)}$ be a rotational surface and H_r be a plane such that the Gauss image of the panel is entirely contained in plane H_r. Then the panel type is determined by the distance d_r of plane H_r from the origin O. Specifically,*

1. *If $d_r = 1$ then $\mathrm{S}^{(r)}$ is planar.*
2. *If $d_r = 0$ then $\mathrm{S}^{(r)}$ is a cylinder of revolution.*
3. *If $d_r \in (0, 1)$ then $\mathrm{S}^{(r)}$ is a cone of revolution whose rulings form the angle $\arcsin d_r$ with the rotation axis.*

This offers a good way to aim at cylindrical or conical panels with prescribed opening angle by setting the according values of d_r in the energy term $\mathcal{E}_{\mathrm{d}}^{(r)}$ in (2.11).

It is often the case in industrial applications that individual adjustments need to be made to the panelization for reasons that include aesthetics and the overall cost of the project. The advantages of the individual treatment of the panels become apparent in such cases, and the aforementioned main pavilion of the Astana EXPO 2017, shown in Figure 2.6, serves as an example. In that project, apart from the cylindrical panels which were the main ingredient of the panelization, double curved panels were also utilized in areas that the use of cylindrical panels would negatively affect the aesthetics of the result. Thus, by integrating a singular panel management strategy to the optimization we have the ability of dealing with isolated problematic areas without sacrificing the quality of the overall panelization.

2.5 Experiments and results

Example 2.5.1. In this example, we consider a mesh $\mathcal{M}$ which originated from scanning a thin deformed leather patch. The deformation was introduced to the material in the form of local stretches along its surface which result in areas of nonzero Gaussian curvature.

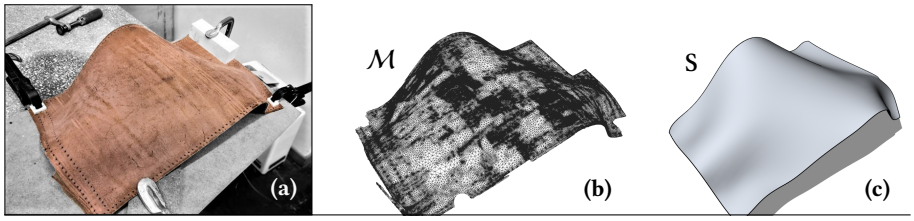


Figure 2.8: (a) The configuration of the deformed leather patch. **(b)** Mesh acquired from scanning the leather material. **(c)** The material's geometry is represented as a B-spline surface.

To apply our algorithm for increasing developability, we first fit the data with a bicubic B-spline surface S of the form (2.4) with 7×13 control points. This is done using the TDM optimization framework for surface fitting described in section 2.3.3. We refer to the initial configuration of surface S, given by the fitting optimization, as $\mathbf{S}_0$. Following the procedure described in section 2.3.1, we sample the resulting surface S uniformly along the parameter space at 30×60 evaluation points $\mathbf{p}_{i,j}$, $i \in [1, 30]$, $j \in [1, 60]$. We then group $\mathbf{p}_{i,j}$ in patches $U_{l,m}$, each one containing 5×5 points with an overlap in both directions of 2 points between neighboring patches, i.e. $U_{l,m} = \{\mathbf{p}_{i,j} \mid i \in [3l-2,\ 3l+2],\ j \in [3m-2,\ 3m+2]\}$. This completes the initialization of the optimization algorithm of problem 2.

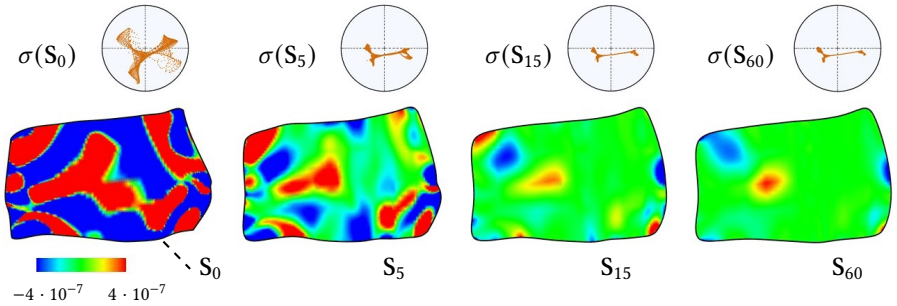


Figure 2.9: The Gauss map (top) and the Gaussian curvature (bottom) of surface S for different numbers of iterations, namely at 0, 5, 15 and 60 ($\mathbf{S}_t$ denotes the optimized surface at t iterations). The length of the surface has been scaled to be approximately 1.

We introduce to the optimization process a closeness energy term of the form (2.8) with relatively small weight to ensure proximity of S to its original position $\mathbf{S}_0$. As described before, this is implemented using the TDM framework. We consider the original surface $\mathbf{S}_0$ as the reference surface and use the already sampled points $\mathbf{p}_{i,j}$ of surface S as the evaluation points of the TDM algorithm. In our experiments, we observed that using this competing low-weight term in our main optimization procedure constrains the solution space by avoiding trivial solutions and producing results that are more desirable from the designer's point of view.

Figure 2.10 reveals the inner workings of the developability algorithm, which clearly produces a "thinner" Gauss image for the resulting surface and also illustrates a comparison between the original surface $\mathbf{S}_0$ and the resulting surface S. Figure 2.9 shows the Gauss map and the Gaussian curvature of the surface for several intermediate iterations of the optimization. The detailed statistics for this example are given in Table 2.1.

We already discussed in section 2.2.2 that the straightening of one family of principal curvature lines of S compared to the principal curvature lines of the initial surface $\mathbf{S}_0$ is an alternative indication of the increase in developability. Figure 2.11 demonstrates the straightening effect in this example. Also illustrated is that the preimage of a small collection of points in one of the "thinner" parts of the Gauss image corresponds to one of the approximate rulings of the surface.

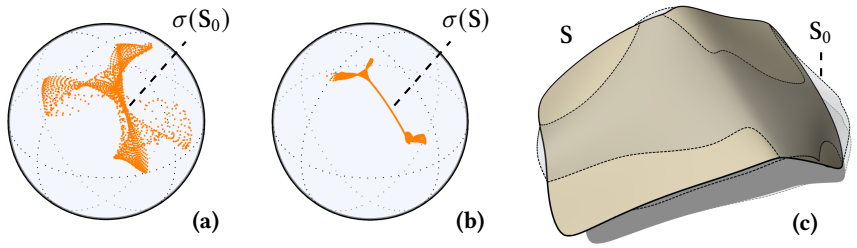


Figure 2.10: **(a)** The Gauss image of the initial configuration of B-spline surface S_0 representing the leather material. **(b)** The Gauss image of the optimized surface S. **(c)** The optimized B-spline surface S in solid color compared to the transparent initial surface S_0.

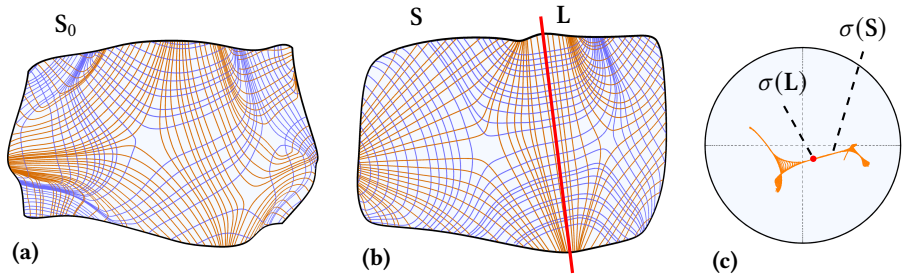


Figure 2.11: Visualization of the principal curvature lines. **(a)** The principal curvature lines of the initial surface S_0. **(b)** The principal curvature lines of the optimized surface S. Highlighted in red and extended slightly for clarity, one such principal curvature line **L**, which also approximately corresponds to the preimage of a small collection of points around the "thin" part of $\sigma(S)$. **(c)** The Gauss image $\sigma(S)$ of the optimized surface S. The Gauss image of **L** is highlighted in red.

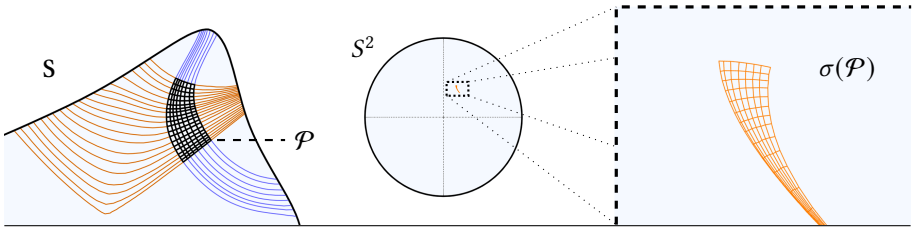


Figure 2.12: We consider a nearly developable patch of a surface S and the two families of principal curvature lines of S (blue and orange lines) over that patch. These families define a principal net denoted with $\mathcal{P}$. The Gauss image $\sigma(\mathcal{P})$ of the net is displayed on the right.

Number of...			Weights			Final energies			Number of	Time [s]		
Ctrl.pts	Patches	Variables	w_d	w_c	w_f	$\mathcal{E}_d$	$\mathcal{E}_c$	$\mathcal{E}_f$	iterations	T_{total}	T_{solver}	T_{iter}
91	200	1073	100	0.01	0.1	2.54	1.8	0.74	60	121.76	0.13	2.03

Table 2.1: We present the detailed information for the optimization of the leather surface S. The number of control points of S and the number of overlapping patches that cover the surface generate the number of variables (3 per control point and 4 per patch-associated plane). The surface was evaluated at 1800 points and each patch contained 25 points. The weights were chosen to favor the developability property. The initial and intermediate total energies of the problem were $\mathcal{E}_0 = 9328.17$, $\mathcal{E}_5 = 2103.75$, $\mathcal{E}_{15} = 356.702$ while the order of the final total energy $\mathcal{E}_{60} = 5.08$ was achieved at iteration 26, where $\mathcal{E}_{26} = 5.38$. Also provided, the total time, time used by the Newton solver, and the time per iteration (in seconds), measured on an Intel® Core™ i7-6700HQ processor.

Example 2.5.2. In this example, we will focus on optimizing two relatively simple non-developable surfaces for planarity of their respective Gauss images. We start with two bicubic Bézier surfaces S_0^a and S_0^b, where S_0^a is of mainly negative Gaussian curvature and S_0^b of positive Gaussian curvature.

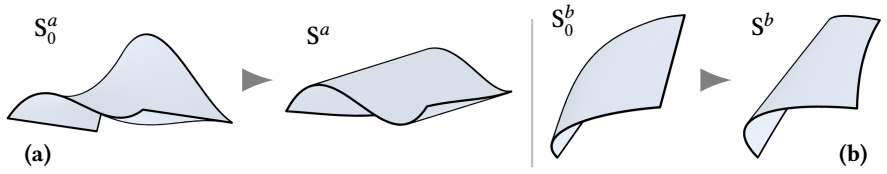


Figure 2.13: The initial surfaces S_0^a, S_0^b and the optimized surfaces S^a, S^b are shown from an appropriate angle to better showcase the emergence of rulings in the direction of least absolute principal curvature on each of the surfaces.

We follow optimization problem 3, defined over a single panel, and utilize only the closeness and developability terms. Given that the surfaces have approximately planar Gauss images after the optimization, we also execute the following procedure at a point set U on the surface to extrapolate the approximate rulings that are derived from their planar Gauss images, defined by the target plane H. We do this to present a visual comparison between these induced rulings and the computed rulings on the optimized surface.

Procedure Induced rulings

for all $\mathbf{p} \in U$ **do**
  $\mathbf{n} \leftarrow \sigma(\mathbf{p})$
  $\mathbf{q} \leftarrow$ *closest point of* $\mathbf{n}$ *to target circle* $H \cap S^2$
  $\mathbf{r}_\mathbf{q}^t \leftarrow$ *vector tangent to target circle at* $\mathbf{q}$
  $\mathbf{r}_\mathbf{q}^o \leftarrow$ *vector tangent to* S^2 *at* $\mathbf{q}$ *and orthogonal to* $\mathbf{r}_q^t$ ▷ *induced ruling direction*
  translate vectors $\mathbf{r}_q^t$, $\mathbf{r}_q^o$ *to* $\mathbf{p}$
end for

The vector $\mathbf{r}_q^o$ approximates the direction of the line generator of the surface at point $\mathbf{q}$. Moreover, for non-inflection rulings and non-planar regions on the optimized surfaces, vectors $\mathbf{r}_\mathbf{q}^t$, $\mathbf{r}_\mathbf{q}^o$ correspond to the principal directions of the surface at point $\mathbf{q}$.

Figure 2.13 shows the surfaces before and after the optimization, while Figure 2.14 shows the resulting vectors from the *Induced rulings* procedure.

Fig.	Number of...				Weights		Final energies		Number of	Time [s]		
No.	Ctrl.pts	Panels	Variables	Eval.pts	w_d	w_c	$\mathcal{E}_\mathrm{d}$	$\mathcal{E}_\mathrm{c}$	iterations	T_total	T_solver	T_iter
2.13a	16	1	52	169	100	1	0.65	81.28	10	1.9	0.1	0.19
2.13b	16	1	52	169	100	1	1.45	99.21	10	2.05	0.02	0.2

Table 2.2: The statistics for the Gauss image planarity optimization of panel surfaces S_0^a and S_0^b. The weights were chosen to favor the developability property. Also provided, the total time, time used by the Newton solver, and the time per iteration (in seconds), measured on an Intel® Core™ i7-6700HQ processor.

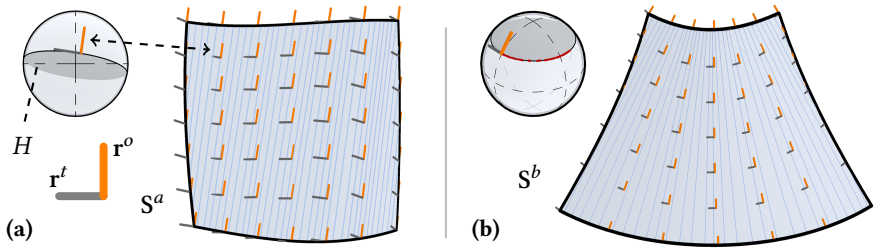


Figure 2.14: A top-down perspective of the optimized surfaces $\mathbf{S}^a$, $\mathbf{S}^b$ is shown with the rulings superimposed on the surfaces (darker blue lines) as well as the resulting vectors from the predefined *Induced rulings* procedure. We draw attention to the comparison between the orthogonal vectors $\mathbf{r}^o$ (orange) and the direction of the rulings (vanishing principal curvature direction). Furthermore, vectors $\mathbf{r}^t$ correspond to the directions of nonzero principal direction.

Example 2.5.3. We provide here an introductory example of paneling a simple double curved surface with a variable number of rotational cylindrical panels.

We consider a surface $\mathbf{S}_{\text{ref}}$ which is a subset of the positive-Gaussian-curvature part of a torus. The active surface $\mathbf{S}$ of the optimization consists of a $N \times 1$ grid of bicubic panels. The initial configuration of $\mathbf{S}$ is given by fitting surface $\mathbf{S}$ to $\mathbf{S}_{\text{ref}}$.

We optimize for the panels of $\mathbf{S}$ to be rotational cylinders in the following manner. First of all, we use Lemma 2.4.1 and assign to each panel an energy term of the form (2.11) with $d_r = 0$ since we are interested in only cylindrical panels. We then solve optimization problem 3 with equal weights assigned to $\mathcal{E}_{\text{d}}$ and $\mathcal{E}_{\text{r}}$, and relatively smaller weights assigned to $\mathcal{E}_{\text{c}}$ and $\mathcal{E}_{\text{f}}$.

Figure 2.15 shows the resulting panelization for different values of N. We wish to direct the reader's focus to the curved boundary lines that follow the reference design; a characteristic not present and inherently not possible without trimming in previous approaches that utilized strips linear in one direction.

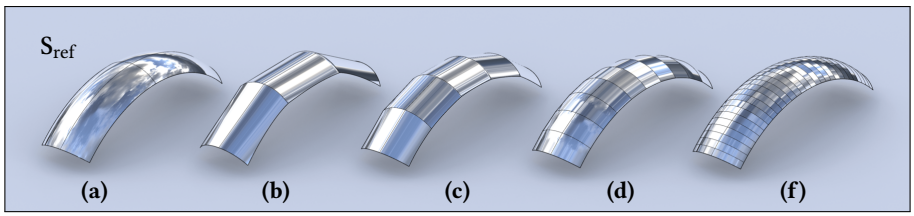


Figure 2.15: Paneling part of a torus with a different number of cylindrical panels. Both the cutting planes U_r per panel $\mathbf{S}^{(r)}$ and the inner boundary curves follow the direction of the smaller radius circles that define the torus.

Fig.	Number of...			Weights				Final energies			Number of	Time [s]		
No.	Ctrl.pts	Panels	Variables	w_d	w_r	w_c	w_f	$\mathcal{E}^{\dagger}_{d+r}$	$\mathcal{E}_c$	$\mathcal{E}_f$	iterations	T_{total}	T_{solver}	T_{iter}
2.15b	40	3	132	10^2	1	1	0.1	0.043	9.97	5.05	5	1.12	0.02	0.22
2.15c	64	5	212	10^2	1	1	0.1	0.004	2.09	5.01	5	2.05	0.03	0.4
2.15d	124	10	412	10^3	10	1	0.1	0.003	0.26	7.11	5	2.99	0.05	0.6
2.15e	364	30	1212	10^3	10	1	0.1	0.0002	0.05	18.68	5	8.11	0.17	1.62

$^{\dagger}\mathcal{E}_{d+r} = \mathcal{E}_d + \mathcal{E}_r$

Table 2.3: The statistics for the paneling of the torus subsurface $\mathbf{S}_{ref}$ for different numbers of panels. Each panel was sampled uniformly at 4×4 points for the developability term and at 10×10 points for the closeness term. The weights were chosen to favor the developability property. Also provided, the total time, time used by the Newton solver, and the time per iteration (in seconds), measured on an Intel® Core™ i7-6700HQ processor.

Example 2.5.4. We extend the previous example of optimizing a simple row of panels to be of cylindrical type to the task of optimizing a grid of panels to be of any developable type we have previously addressed for panels.

Motivated by the possible architectural applications of the algorithm presented in this chapter, we use as a reference surface an architectural design recently realized as the roof of the Department of Islamic Art at Musée du Louvre in Paris, France, shown in Figure 2.16. The underlying surface is a highly non-developable surface with a strong variation in the sign of Gaussian curvature. In this example, we set forth to compute an alternative realization of the same surface by using rotational conical and rotational cylindrical panels.

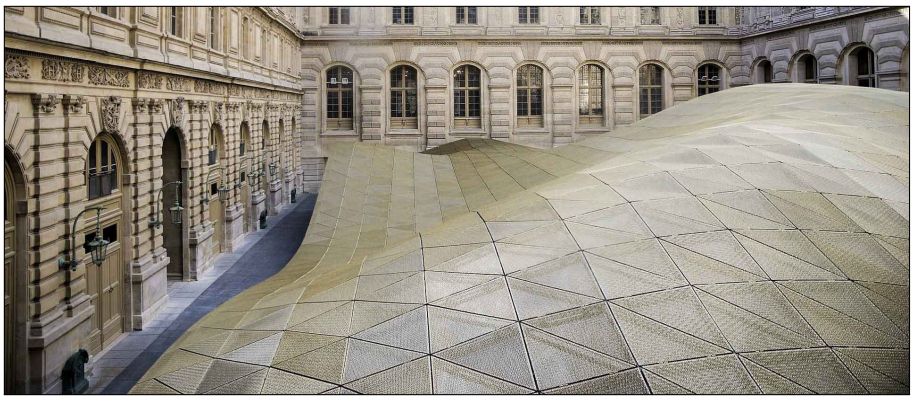


Figure 2.16: Detail from the *Cour Visconti* roof of the Department of Islamic Art at Musée du Louvre in Paris, France.

The user input in this case is the freeform reference surface $\mathbf{S}_{ref}$, the desired number of panels in each direction of the grid that will constitute the panelization of the surface and the preferred type of panels, which includes surfaces of constant slope or the more specialized and more widely-used rotational surfaces of constant slope, i.e. rotational conical and rotational cylindrical. The user, by adjusting the weights of the different energy terms involved in the corresponding optimization problem 3, has influence over the various desirable aspects of the resulting panelization. In this particular example, we wish to use any of the types introduced before, namely rotational conical, rotational cylindrical and planar panels.

We present in Figure 2.17 the resulting panelization of the reference surface for different panel grid resolutions. We set weight w_c, corresponding to the closeness of S to S_{ref}, relatively high to reinforce the resulting surface to not deviate significantly from the reference surface and closely follow the chosen design. The smoothness of the boundary curves is controlled by the fairness energy term weight w_f, which is assigned a small value to ensure more visually pleasing results.

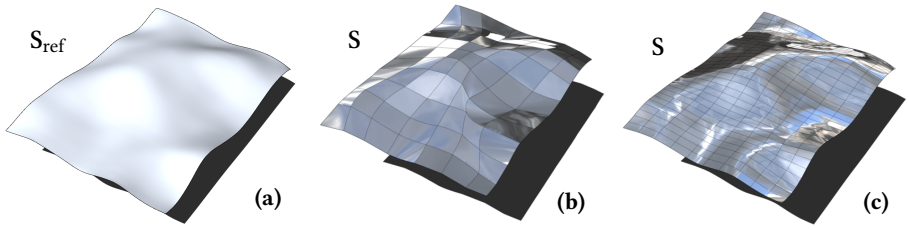


Figure 2.17: (a) The freeform reference surface to be panelized. **(b)** A coarse panelization consisting of 70 panels. **(c)** A denser panelization consisting of 300 panels. Runtime for both the coarse and the finer paneling was several minutes.

The coarse panelization of Figure 2.17b serves as a nice example of the dynamic panel layout adaptation which aims to approximate the given reference surface while satisfying the developability, rotationality and closeness constraints. On the contrary, by increasing the number of the panels utilized, we achieve the dense panelization of Figure 2.17c. As expected the increased number of panels produces an improved result, compared to the coarse equivalent. It not only better approximates the reference surface but also satisfies to a higher degree the additional secondary constraints, yielding a panelization of the reference surface that allows for a more structured arrangement of the panels.

Nevertheless, both results are welcome since each one of them serves as a valid panelization with specialized developables of the same architectural surface. Each one of the two panelizations of this example shown in Figures 2.17b, 2.17c manages to be architecturally aesthetically pleasing in its own style, while being realizable only by rotational cylindrical and rotational conical panels; highlighting the freedom of design expression that this method provides.

2.6 Discussion

In this chapter, we have introduced a methodology for increasing the developability of surfaces through an optimization algorithm which aims at a thin Gauss image. Our implementation uses B-spline surfaces, but an analogous approach could be formulated for other surface representations as well. Moreover, we have presented a novel paneling algorithm which—in contrast to prior work [Eigensatz et al., 2010]—optimizes both for the panels and the curve network of panel boundaries, under the constraint that panels are developable with a planar Gauss image and/or rotational.

Appropriate choice of weights leads to high-precision satisfaction of the hard nonlinear constraints. The fairness and closeness terms act as regularizers to the optimization problem, which is formulated through simple polynomial energies. The combination of the soft constraints and fixed points, avoids degenerate results. The complexity of the approach is derived by the degree of the surface to be optimized,

the reference surface (number of points of mesh representation) and number of evaluation points. In most applications, our experiments show that the computation time is limited to several seconds to get satisfactory results.

The presented local shaping approach achieves to minimize the predefined energies at every step, and guides iteratively the surface to an expected result. Any unwanted results were limited to surfaces that could not satisfy adequately both the closeness term and the developability term, meaning the result had to deviate considerably from the reference to satisfy the developability constraint.

Among the limitations of this approach, we first point to the lack of a material-dependent measure for the deviation from developability. The thickness of the Gauss image alone is not sufficient for judging whether a panel, fabricated from hardly stretchable material, can be easily bent into the computed shape. Moreover, our current implementation for paneling is limited to a grid type arrangement of panels and could benefit from additional improvements to the optimizer.

The next chapter aims to address the first point. We focus on glass as a material and explore the deformation space of cold bent glass panels. Since glass is much more easily bent than stretched, we could characterize the shapes that are attainable through cold bending of an initially planar glass panel as nearly developable. While we do not explicitly study the geometric properties of such a deformation space, we still aim to capture it for practical usage, i.e. interactive design of cold bent glass panelizations.

Computational Design of Cold Bent Glass Façades

3

3.1 Introduction

Curved glass façades allow the realization of aesthetically stunning looks for architectural masterpieces as shown in Figure 3.1. The curved glass is usually made with hot bending, a process where the glass is heated up and then formed into shape using a mold or tailored bending machines for spherical or cylindrical shapes. While unleashing designs from being restricted to flat panels, this process is laborious and expensive and thus an economic obstacle for the realization of exciting concepts such as the NHHQ skyscraper project by Zaha Hadid Architects (Figure 3.10). As a cost-effective alternative, in recent years architects have started exploring cold bending [Beer, 2015]. Planar glass sheets are deformed by mechanically attaching them to a curved frame. Cold bending introduces a controlled amount of strain and associated stress in flat glass at ambient temperatures to create double-curved shapes [Datsiou, 2017]. Compared to hot bent glass, it has the advantage of higher optical and geometric quality, a wide range of possibilities with regards to printing and layering, usage

Figure 3.1: Examples of Curved Glass Façades. Left: *Louis Vuitton Foundation* by Frank Gehry in Paris, France (photo by Anzola, 2020). Right: *Emporia* by Gert Wingårdh in Malmö, Sweden (photo by Eklind, 2015).

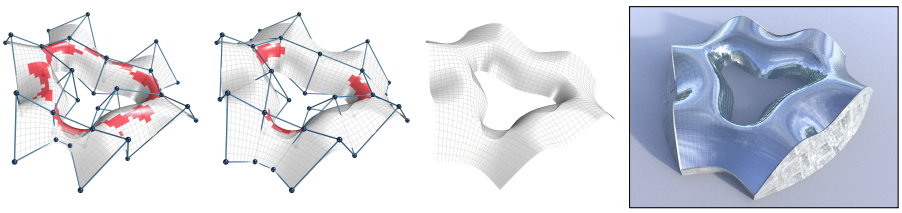


Figure 3.2: Material-aware form finding of a cold bent glass façade. From left to right: initial and revised panel layouts from an interactive design session with immediate feedback on glass shape and maximum stress (red color indicates panel failure). The surface design is then optimized for stress reduction and smoothness. The final façade realization using cold bent glass features double-curved areas and smooth reflections.

of partially tempered or toughened safety glass, and the possibility of accurately estimating the stresses due to deformation [Fildhuth & Knippers, 2011]. Furthermore, it reduces energy consumption and deployment time because no mold, heating of the glass, nor elaborate transportation is required.

However, designing cold bent glass façades faces a challenging form finding process. How to identify a visually pleasing surface that meets aesthetic requirements such as smoothness between panels while ensuring that the solution is physically feasible and manufacturable? Significant force loads can occur at the connection between glass and frame, and it is essential that the deformation of the glass stays within safe limits to prevent it from breaking.

In this chapter, we present an interactive, data-driven approach for designing cold bent glass façades. Starting from an initial quadrangulation of a surface, our system provides a supporting frame and interactive predictions of the shape and the maximum stress of the glass panels. Following a designer-in-the-loop optimization approach, our system enables users to quickly explore and automatically optimize designs based on desired trade-offs between smoothness, maximal stress, and closeness to a given input surface. Our workflow allows working on the 3D surface and the frame only, liberating the designer from the need of considering or manipulating the shape of flat panels—the optimal shape of the flat rest configuration of the glass panels is computed automatically.

At a technical level, we aim to determine the minimum energy states of glass panels conforming to the desired boundary without knowing their rest configuration. Based on extensive simulations of more than a million panel configurations with boundary curves relevant for our application domain, we observed the existence of several (in most cases up to two) stable states for many boundary curves. Identifying both *minimum energy states without knowing the rest configuration* and potentially *multiple stable states* is a non-trivial problem and cannot be easily computed with standard simulation packages. Furthermore, as a prerequisite for enabling *interactive design* for glass façades, we need to solve this problem for hundreds of panels within seconds.

To achieve these goals, we develop a learning-based method utilizing a deep neural network architecture and Gaussian mixture model that accurately predicts the shape and maximum stress of a glass panel given its boundary. Training data for the network is acquired from a physics-based shape optimization routine. Predictions of the trained network not observed originally are re-simulated and used for the database enrichment. Our model is differentiable, fast enough to interactively optimize and explore the shape of glass façades consisting of hundreds of tiles, and

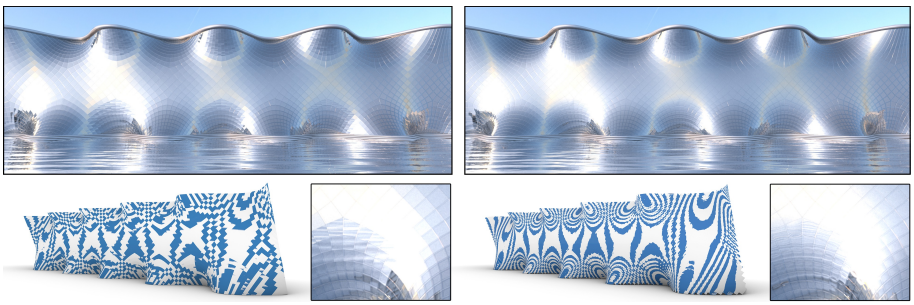


Figure 3.3: Double-curved surface panelized using a planar quad mesh following the principal curvature network (left). This is the smoothest possible panelization of this surface achievable with flat panels [Pellis et al., 2019]. The solution using cold bent glass panels designed with our method (right) shows much smoother results. Bottom pictures show the corresponding zebra stripping for both solutions; clearly smoother stripes are indicators of higher visual smoothness.

tailored to be easily integrated into the design workflow of architects. As a proof-of-concept, we have integrated our system in Rhino. We have carefully validated the accuracy and performance of our model by comparing it to real-world examples, and demonstrate its applicability by designing and optimizing multiple intricate cold bent glass façades.

3.1.1 Related work

Interactive Design and Shape Optimization are areas that have a considerable history in Computer Graphics research [Bermano et al., 2017; Bickel et al., 2018], including tools for designing a wide variety of physical artifacts, such as furniture [Umetani et al., 2012], cloth [Wolff & Sorkine-Hornung, 2019], robotics [Megaro et al., 2015], and structures for architecture [Eigensatz et al., 2010].

Motivated by the digitalization of manufacturing, there is an increased need of computational tools that can predict and support optimizing the physical performance of an artifact during the design process. Several approaches were developed to guarantee or improve the structural strength of structures [Stava et al., 2012; Ulu et al., 2017]. Focusing on shell-like structures, Musialski et al. [2015] optimized their thickness such that it minimizes a provided objective function. More recently, Zhao et al. [2017] proposed a stress-constrained thickness optimization for shells, and Gil-Ureta et al. [2019] computed a rib-like structure for reinforcing shells, that is, adding material to the shell to increase its resilience to external loads. Considering both aesthetic and structural goals, Schumacher et al. [2016] designed shells with an optimal distribution of artistic cutouts in a manner that produces a stable final result. While we share the general goal of structural soundness, in our problem setting we cannot change the thickness or material distribution. Additionally, even just determining the feasibility of a desired bent glass shape requires not only solving a forward simulation problem, but also an inverse problem, as the rest shape of the glass panel is a priori unknown. Finding an optimal rest shape is often extremely important. Schumacher et al. [2018] investigate sandstone as building material that is weak in tension, thus requiring computing an undeformed configuration for which the overall stress is minimized. Similarly, glass panels have a low tensile strength and are subject to very high compression loads during the assembly process, which motivates the finding of minimal energy panels.

Notably, several methods have recently been proposed to design double-curved objects from flat configurations [Guseinov et al., 2017; Konaković-Luković et al., 2018; Malomo et al., 2018; Panetta et al., 2019]. However, all these methods rely on significantly more elastic materials and are not targeted to be used within an interactive design pipeline. In our application, having an accurate estimation of the stress is critical to predict panel failure and interactively guide designers towards feasible solutions. The need to bridge the gap between accuracy and efficiency motivates the use of a data-driven approach.

Computational design of façades. We already partially covered the related work on the panelization problem in 2. We continue the discussion here, aiming towards the specific problem of panelizing a façade with cold bent glass panels.

We remind the reader that covering general freeform surfaces with planar quadrilateral panels is a fundamental problem in architectural geometry and has received a lot of attention; see e.g. [Glymph et al., 2004; Liu, Pottmann, Wallner, et al., 2006; Liu et al., 2011; Mesnil et al., 2017; Pottmann et al., 2015]. The difficulties lie in the close relation between the curvature behavior of the reference surface and the possible panel layouts. Problems occur especially in areas of negative curvature and if design choices on façade boundaries are not aligned with the curvature constraints imposed by planar quad meshes (Figure 3.3). Using triangular panels, the problems are shifted towards high geometric complexity of the nodes in the support structure [Pottmann et al., 2015]. Eigensatz et al. [2010] formulated relevant aspects for architectural surface paneling into a minimization problem that also accounted for re-using molds, and thereby reducing production cost. Restricting the design to single-curved panels, Pottmann et al. [2008] presented an optimization framework for covering freeform surfaces by developable panels arranged along surface strips. However, glass does not easily bend into general developable shapes, which limits the applicability of this technique for paneling with glass.

A recent alternative for manufacturing double-curved panels is cold bending of glass. A detailed classification and description of the performance of cold bent glass can be found in [Datsiou, 2017]. Eversmann, Ihde, et al. [2016] explored simulations based on a particle-spring method, a commercially available FE-analysis tool, and physical prototyping of cold bent glass and compared the resulting geometries to the measurements of the physical prototypes. For designing multi-panel façade layouts, Eversmann, Schling, et al. [2016] calculated the maximum Gaussian curvature for a few special types of double-curved panels. This defined a minimal bending radius for exploring multi-panel façade layouts. While conceptually simple, we found this approach too limiting for general curved panels and thus base our approach on a data-driven method.

Machine learning for data-driven design. Finite Element methods (FEM) are widely used in science and engineering for computing accurate and realistic results. Unfortunately, they are often slow and therefore prohibitive for real-time applications, especially in the presence of complex material behavior or detailed models.

Dimensionality reduction is a powerful technique for improving simulation speed. Reduced space methods, for example based on modal analysis [Barbič & James, 2005; Pentland & Williams, 1989], are often used to construct linear subspaces, assuming that the deformed shape is a linear combination of precomputed modes. Simulations can then be performed in the spanned subspace, which, however, limits its accuracy,

especially in the presence of non-linear behavior. Non-linear techniques, such as numerical coarsening [D. Chen et al., 2015], allow to reduce models with inhomogeneous materials, but usually require to precompute and adjust material parameters or shape functions [J. Chen et al., 2018] of the coarsened elements. Recently, Fulton et al. [2019] proposed employing autoencoder neural networks for learning nonlinear reduced spaces representing deformation dynamics. Using a full but linear simulation, NNWarp [Luo et al., 2020] attempts to learn a mapping from a linear elasticity simulation to its nonlinear counterpart. Common to these methods is that they usually precompute a reduced space or mapping for a *specific* rest shape but are able to perform simulations for a wide range of Neumann and Dirichlet boundary constraints. In our case, however, we are facing a significantly different scenario. First, we need to predict and optimize the behavior of a whole range of rest shapes, which are defined by manufacturing feasibility criteria (in our case close to, but not necessarily perfect rectangular flat panels). Second, our boundary conditions are fully specified by a low-dimensional boundary curve, which corresponds to the attachment frame of the glass panel. Instead, we therefore propose to directly learn the deformation and maximal stress by the boundary curve.

Recently, data-driven methods have shown great potential for interactive design space exploration and optimization, for example for garment design which can be used for animation [T. Y. Wang et al., 2019], or optimized tactile rendering based on a data-driven skin mechanics model [Verschoor et al., 2020]. An overview of graphics-related applications of deep learning can be found in [Mitra et al., 2019]. In the context of computational fabrication, data-driven approaches were used for example for interactively interpolating the shape and performance of parameterized CAD Models [Schulz et al., 2017], or learning the flow for interactive aerodynamic design [Umetani & Bickel, 2018]. While these methods are based on an explicit interpolation scheme of close neighbors in the database [Schulz et al., 2017] or Gaussian processes regression [Umetani & Bickel, 2018], in our work we demonstrate and evaluate the potential of predicting the behavior and solving the inverse problem of designing a cold bent glass façade using neural networks. This entails the additional challenge of dealing with multistable equilibrium configurations which, to our knowledge, has not not been addressed before in a data-driven computational design problem.

3.1.2 Contributions

The main contributions covered in this chapter are as follows:

- We train a mixture density network—a regression model capable of handling multistable configurations—to accurately predict the shape and maximal stress of a cold bent glass panel. The model was trained on a dataset of more than a million pre-computed simulations. This allows us to interactively navigate the design space of cold bent glass panels without the need of additional simulations.

- We integrate the data-driven model to an interactive design tool that allows for real-time feedback on panelizations of freeform façades with cold bent glass panels. The differentiability of the model allows for form-finding of practically feasible and aesthetically pleasing cold bent glass façades through appropriately formulated optimization.

- We provide multiple results, applications of the design tool, and validation of the method through a proof-of-concept physical prototype.

3.1.3 Overview

We propose a method for the interactive design of freeform surfaces composed of cold bent glass panels that can be seamlessly integrated in a typical architectural design pipeline. Figure 3.4 shows an overview of the design process. The user makes edits on a base quad mesh that is automatically completed by our system to a mesh with curved Bézier boundaries. Our data-driven model then interactively provides the deformed shape of the cold bent glass panels in form of Bézier patches conforming to the patch boundaries and the resulting maximal stress. This form finding process helps the designer to make the necessary decisions to avoid panel failure. At any point during the design session, the user can choose to run our simulation-based optimization method to automatically compute a suitable panelization while retaining some desirable features such as surface smoothness and closeness to the reference design.

In Section 3.2, we show how the base mesh controlled by the user is extended through special cubic Bézier curves to the set of patch boundaries. Each patch is delimited by planar boundary curves of minimum strain energy. These special Bézier patch boundaries are convenient for modeling glass panels as they facilitate the construction of supporting frames while providing a smooth approximation to the desired design.

Bézier boundaries do not convey any information on the deformed or undeformed configuration of the panel. Our method uses simulation to compute both configurations of the panel such that certain conditions are met which are derived from manufacturing constraints. First, current panel assembly does not guarantee C^1 continuity at the boundary between neighbor panels as it is very hard to enforce normals along the frame in practice. Second, glass panels have low tensile strength and are prone to breaking during the installation process in the presence of large tangential forces. Following these criteria, we let the panel be defined by the boundary curve of the frame, and compute both the deformed and undeformed shapes of the panel such that the resulting total strain energy is minimal. This way, we ensure our panelization has at least C^0 continuity and the assembly of the panels requires minimal work, thus reducing the chances of breakage. In Section 3.3, we describe in detail the physical model and the computation of minimal energy panels.

Panel shape optimization provides us with a mapping between our design space of Bézier boundary curves and theoretically realizable cold bent panels, in both undeformed and deformed configurations. Our material model also accurately estimates the maximum stress endured by the glass. The user is free to interactively edit the base mesh while receiving immediate feedback on the maximum stress, but this neither ensures the panels will not break, nor fosters the approximation of a target reference surface. To achieve this goal, we solve a design optimization problem: Bézier boundary curves are iteratively changed to minimize closeness to an input target surface (and other surface quality criteria) while keeping the maximum stress of each panel within a non-breaking range. In Section 3.5, we describe in detail our formulation of the design optimization.

However, accurately computing minimal energy panels is computationally very challenging which makes physical simulation infeasible for being directly used within the design optimization loop. Furthermore, the mechanical behavior of glass panels

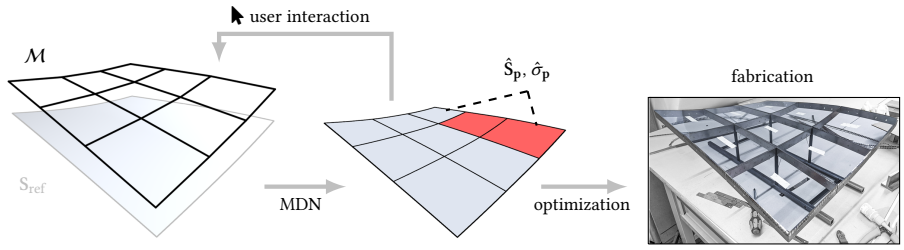


Figure 3.4: Overview of our design tool workflow. The user interacts with a quad mesh $\mathcal{M}$ and gets immediate feedback on the (predicted) deformed shape $\hat{\mathbf{S}}_\mathbf{p}$ and maximal stress $\hat{\sigma}_\mathbf{p}$ of glass panels defined by their boundary $\mathbf{p}$. When needed, an optimization procedure interactively refines the surface to minimize safety and fairness criteria. Optionally, a reference surface $\mathbf{S}_{\text{ref}}$ may be used as a target surface.

under compression often leads to multiple stable minimal energy configurations depending on the initial solution. This complicates the optimization even more: not only does the problem turn into a combinatorial one, but there is no algorithmic procedure to count and generate all existing static equilibria given some boundary curve. We address this challenge by building a data-driven model of the physical simulation. First, we densely sample the space of Bézier planar boundary curves and compute the corresponding minimal energy glass panels, together with an estimation of the maximum stress. Then we train a mixture density network (MDN) to predict the resulting deformed shape and maximum stress given the boundary of the panel. The MDN explicitly models multistability, and also allows us to discover alternative stable equilibria that can be used to enrich the training set. In Section 3.4, we elaborate on the characteristics of our regression network and our sampling and training method. The trained regression network can be finally used to solve the inverse design optimization problem. Once the user is satisfied with the design, our shape optimization procedure generates the rest planar panels, which are ready to be cut and assembled into a beautiful glass façade.

3.2 Geometry representation

We build a panelization of an architectural surface upon a quadrangular base mesh $\mathcal{M} = (V, E, F)$, where vertices V determine the panel corner points and each quad face in F is filled by one curved panel. In practice, the user interacts with the design tool by making edits to $\mathcal{M}$ through any parametric mesh design method, in our case Catmull-Clark subdivision from a coarser mesh (see Figure 3.2). This helps to achieve fair base meshes and gives a reasonable control for edits. However, any other mesh design scheme could be potentially used.

Each edge in E is then automatically replaced by a planar cubic Bézier curve defining the boundaries of the panel and the inner control points are predicted using our regression model. In this section we describe the details for getting from $\mathcal{M}$ to the union of curved panels. Moreover, we show how to express the panels with a minimal number of parameters which are later used for the data-driven model.

3.2.1 Panel parameterization

We model each glass panel as a bicubic Bézier patch $\mathbf{S} : [0,1]^2 \to \mathbb{R}^3$, defined by 16 control points $\mathbf{P}_{i,j}$, where $i,j \in \{0,1,2,3\}$. The corner points $\mathbf{P}_{0,0}$, $\mathbf{P}_{0,3}$, $\mathbf{P}_{3,3}$, $\mathbf{P}_{3,0}$ are

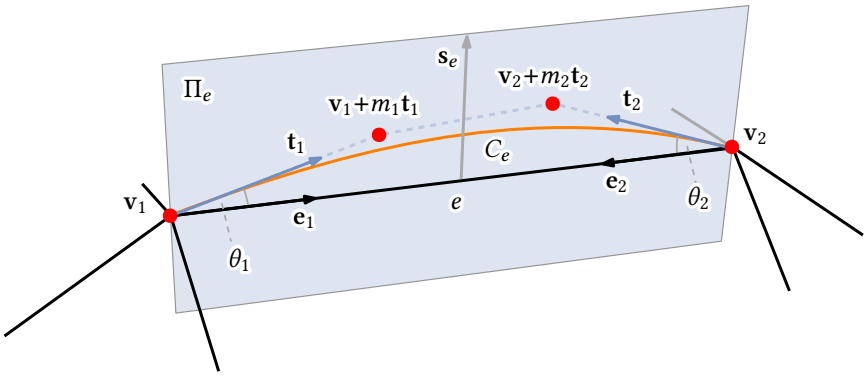


Figure 3.5: Parameterization of a panel boundary curve from a pair of tangent directions $\mathbf{t}_1$, $\mathbf{t}_2$ corresponding to dual halfedges. The final boundary curve C_e (orange) is computed by minimizing a linearized bending energy.

vertices in $\mathcal{M}$.

Panel boundary. Each edge e of $\mathcal{M}$ is associated with a patch boundary curve $\mathbf{C}_e$. To describe its construction, we focus on a single edge e with vertices $\mathbf{v}_1, \mathbf{v}_2$, and we denote the unit vectors of the half-edges originating at $\mathbf{v}_i$ by $\mathbf{e}_i$ (see Figure 3.5). We opted for planar boundary curves of the panels and thus we first define the plane Π_e which contains $\mathbf{C}_e$. We do this by prescribing a unit vector $\mathbf{s}_e \in \mathbb{R}^3$ which lies in Π_e and is orthogonal to e. The two inner control points of the cubic curve $\mathbf{C}_e$ lie on the tangents at its end points. Tangents are defined via the angles θ_i they form with the edge. Hence, the unit tangent vectors are

$$\mathbf{t}_i = \mathbf{e}_i \cos\theta_i + \mathbf{s}_e \sin\theta_i,$$

In view of our aim to get panels which arise from flat ones through bending, we further limit the cubic boundary curves to those with minimal (linearized) bending energy as described in [Yong & Cheng, 2004]. For them, the two inner control points are given by $\mathbf{v}_i + m_i\mathbf{t}_i$, $i = 1, 2$, with

$$m_1 = \frac{(\mathbf{v}_2 - \mathbf{v}_1) \cdot [2\mathbf{t}_1 - (\mathbf{t}_1 \cdot \mathbf{t}_2)\mathbf{t}_2]}{4 - (\mathbf{t}_1 \cdot \mathbf{t}_2)^2},$$

and m_2 is obtained analogously by switching indices 1 and 2.

The boundary of $\mathbf{S}$ is thus fully parameterized by the 4 corner vertices $\mathbf{P}_{i,j}$, $i, j \in \{0, 3\}$, the 4 edge vectors $\mathbf{s}_e$, and the 8 tangent angles θ (2 per edge). This parameterization of panels is used in the regression model and the design tool implementation described in Sections 3.4 and 3.5 respectively.

Panel interior. The interior control points $\mathbf{P}_{i,j}$, $i, j \in \{1, 2\}$ express the shape of a panel enclosed by a given boundary. We found that within the admissible ranges of the boundary parameters any optimal glass panel (see detailed description in Section 3.3) can be very closely approximated by fitting the internal nodes of the Bézier patch. Moreover, we need to regularize the fitting, since for a given Bézier patch,

it is possible to slide the inner control points along its surface while the resulting geometry stays nearly unchanged.

We denote the vertices of the target panel shape $\mathbf{x}_i$ and the corresponding vertex normals $\mathbf{n}_i$. For every $\mathbf{x}_i$ we find the closest points $\mathbf{y}_i$ on the Bézier surface and fix their coordinates in the parameter domain. The fitting is then formulated as follows:

$$\min_{\mathbf{P}_{i,j}} \quad \left\|\left(\mathbf{y}_i(\mathbf{P}_{i,j}) - \mathbf{x}_i\right) \cdot \mathbf{n}_i\right\|^2 A_i + w_{\mathrm{B}} \sum_k E_k^2(\mathbf{P}_{i,j}), \quad i,j \in \{1,2\},$$

where A_i are Voronoi cell areas per panel vertex, E_k are lengths of all control mesh edges incident to the internal nodes, and w_{B} is the regularizer weight which we set to 10^{-5}. In order to achieve independence of rigid transformations, we express the inner control points in an orthonormal coordinate frame adapted to the boundary. The frame has its origin at the barycenter of the four corner points. Using the two unit diagonal vectors

$$\mathbf{g}_0 = \frac{\mathbf{P}_{3,3} - \mathbf{P}_{0,0}}{\|\mathbf{P}_{3,3} - \mathbf{P}_{0,0}\|}, \quad \mathbf{g}_1 = \frac{\mathbf{P}_{3,0} - \mathbf{P}_{0,3}}{\|\mathbf{P}_{3,0} - \mathbf{P}_{0,3}\|},$$

the x-axis and y-axis are parallel to the diagonal bisectors, $\mathbf{g}_1 \pm \mathbf{g}_0$, and the z-axis is parallel to $\mathbf{b} = \mathbf{g}_0 \times \mathbf{g}_1$, which we call the face normal.

3.2.2 Compact representation

The panel boundary is used as input to a neural network to predict the shape and stress of the minimal energy glass panel(s) conformal to that boundary. Thus, it is beneficial to reduce the input to the essential parameters, eliminating rigid transformations of the boundary geometry.

We consider $\mathbf{d} \in \mathbb{R}^6$ to be the vector of the six pairwise squared distances of vertices $\mathbf{P}_{i,j}$, $i,j \in \{0,3\}$. Given $\mathbf{d}$, we can recover two valid mirror-symmetric embeddings of the 4 corner points. Assuming that the order of the vertices is always such that

$$\det(\mathbf{P}_{0,3} - \mathbf{P}_{0,0}, \mathbf{P}_{3,0} - \mathbf{P}_{0,0}, \mathbf{P}_{3,3} - \mathbf{P}_{0,0}) \geq 0$$

holds, the embedding is unique up to rigid transformations. We would assume such a vertex ordering from now on. The plane Π_e for each edge is then characterized by its oriented angle γ_e with the face normal $\mathbf{b}$. Finally, we define $\mathbf{p} \in \mathbb{R}^{18}$ (for *panel boundary*) as the concatenation of the distance vector $\mathbf{d}$, the 4 edge plane inclinations γ_e, and the 8 tangent angles θ (2 per edge). The vector $\mathbf{p}$ is used as input to the neural network defined in Section 3.4.

3.3 Panel shape optimization

Our method leverages mechanical simulation to create a large dataset of minimal energy panels that conform to cubic Bézier boundaries. Given some boundary curves, we are interested in finding deformed glass configurations that are as developable as possible. Non-developable panels result in high tangential forces which complicate the installation of the panel and increase the chances of breakage. By finding the pair of deformed and undeformed shapes of the panel that minimize the strain energy subject to a fixed frame, we ensure the work required for its installation is minimal and help to reduce the tangential force exerted at the boundary. This dataset

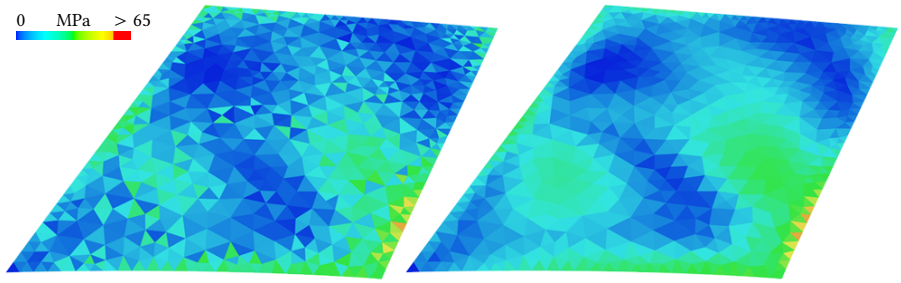


Figure 3.6: Comparison between the stress distribution produced with the typical shape operator used in, e.g., Pfaff et al., 2014 (left), and ours suggested in Grinspun et al., 2006 (right). The latter is much smoother and results in a more reliable estimation of the maximal stress.

is used to train and test a model that predicts the deformed state and maximum stress of such panels, which is suitable for rapid failure detection and inverse design. In this section we describe in detail the simulation method used for the computation of the deformed and undeformed states of a minimal energy glass panel.

3.3.1 Continuous formulation

We aim to define a mechanical model that is sufficiently precise to accurately predict glass stresses under small strains, but still suitable for the fast simulation of a very large number of deformation samples. Consequently, we make some reasonable simplifying assumptions in a similar way to Gingold et al. [2004]. We geometrically represent a glass panel as a planar mid-surface extruded in two opposite normal directions by a magnitude $h/2$, where the total thickness h is much smaller than the minimal radius of curvature of the reference boundary frame. We assume the lines normal to the mid-surface always remain straight and do not undergo any stretching or compression. Under linearity assumption, the following expression for the volumetrically-defined Green's strain tensor $\mathbf{E}$ with offset z in the normal direction can be derived:

$$\mathbf{E}(\mathbf{x}, \bar{\mathbf{x}}, z) = \bar{\mathbf{E}}(\mathbf{x}, \bar{\mathbf{x}}) + z\hat{\mathbf{E}}(\mathbf{x}, \bar{\mathbf{x}}). \tag{3.1}$$

Here $\mathbf{x}$ and $\bar{\mathbf{x}}$ are respectively the deformed and undeformed configurations of the mid-surface, and $\hat{\mathbf{E}}$ is the quadratic bending strain, equivalent to the shape operator of the deformed mid-surface. The membrane strain $\bar{\mathbf{E}} = 0.5(\mathbf{F}^T\mathbf{F} - \mathbf{I})$ is the in-plane Green's strain tensor defined in terms of the deformation gradient $\mathbf{F}$. We refer to the original paper for a detailed explanation of the continuous formulation. We will focus on our discrete formulation which has been previously considered by Weischedel [2012].

3.3.2 Discrete formulation

We discretize glass panels using a triangulated surface mesh with N nodes and M edges. We separately consider the membrane and bending strains from Equation (3.1) and define two corresponding mid-surface energy densities integrated over the panel thickness.

Membrane energy density. To discretize membrane strain, we assume piecewise constant strains over FEM elements. In this context, in-plane Green's strain is computed as follows:

$$\bar{\mathbf{E}} = \frac{1}{16A^2} \sum_{i}^{3} s_i (\mathbf{t}_j \otimes \mathbf{t}_k + \mathbf{t}_k \otimes \mathbf{t}_j), \tag{3.2}$$

where A is the triangle area, $s_i = \bar{l}_i^2 - l_i^2$ (i'th edge strain), $\mathbf{t}_j$ and $\mathbf{t}_k$ are the two other edge vectors rotated by $-\pi/2$. For computing the corresponding membrane energy density integrated over the panel thickness, we adopt the Saint Venant-Kirchhoff model:

$$\bar{W} = h \left(\frac{\lambda}{2} \left(\operatorname{Tr} \bar{\mathbf{E}} \right)^2 + \mu \operatorname{Tr} \left(\bar{\mathbf{E}}^2 \right) \right), \tag{3.3}$$

where λ and μ are respectively first and second Lamé parameters.

Bending energy density. Bending strain is directly defined as the geometric shape operator of the continuous surface. We compute a discrete approximation of the shape operator using the triangle-based discretization suggested in [Grinspun et al., 2006], which faithfully estimates bending strain regardless of the irregularity of the underlying triangle mesh. In addition to mesh nodes, this metric considers additional degrees of freedom per edge defining deviation of mid-edge normals from the adjacent triangle-averaged direction:

$$\hat{\mathbf{E}} = \sum_{i}^{3} \frac{\theta_i/2 + \phi_i}{A l_i} (\mathbf{t}_i \otimes \mathbf{t}_i). \tag{3.4}$$

Here, θ_i is a dihedral angle associated with the edge i and ϕ_i is the deviation of the mid-edge normal towards the neighbor triangles normals. Overall, the discrete deformed state of the glass panel is defined with a vector $\mathbf{x} \in \mathcal{R}^{3N+M}$. We denote the corresponding undeformed configuration with $\bar{\mathbf{x}}$. Bending energy density integrated over the panel thickness, is then defined by the Koiter's shell model [Koiter, 1966]:

$$\hat{W} = \frac{\mu h^3}{12} \left(\frac{\lambda}{\lambda + 2\mu} \left(\operatorname{Tr} \hat{\mathbf{E}} \right)^2 + \operatorname{Tr} \left(\hat{\mathbf{E}}^2 \right) \right) .7 \tag{3.5}$$

Contrary to simpler thin shell bending models commonly used in computer graphics, e.g. [Pfaff et al., 2014], the discrete shape operator suggested in [Grinspun et al., 2006] more faithfully captures principal strain curves and outputs smoother stress distributions (Figure 3.6). In the next section, we will describe how we find the minimal energy configuration corresponding to some given Bézier boundaries.

3.3.3 Minimal energy panels

Given a parametric design of a façade composed of a quadrangular mesh with Bézier curves at the edges, we aim to find a suitable panelization using cold bent glass. While deforming a glass panel to conform to cubic boundaries is feasible, the fragility of this material imposes nontrivial constraints on the maximum amount of stress tolerated by the panels. Thus, we designed a method to compute the glass panel design with the lowest possible strain energy while fitting our installation constraints. Note that, in practice, existing assembly methods do not preserve normals across neighboring

panels, and thus we restrict our problem to guarantee only C^0 continuity. By computing the fabricable panel with the lowest possible total strain energy, we minimize the net work required to install the panel and notably reduce the local tangential stress suffered by the glass.

Overall, our pipeline takes as input the 16 control points of the Bézier boundaries, and automatically computes both the deformed $\mathbf{x}$ and the undeformed $\bar{\mathbf{x}}$ configurations of a planar glass panel that conforms to the boundary and has minimal energy. This is done in two steps.

Initialization. At first, we generate a regular mesh uniformly discretizing the parameter domain of the surface (a unit square), and lift the vertices to an initial Bézier patch defined by the boundaries. In our pipeline, such a patch can be obtained in two ways:

- Generated by our prediction model, when shape optimization is used to enrich the database or to compute the undeformed shape of the final design panels.
- Initialized as a surface patch with zero twist vectors at the corners (quad control mesh has parallelograms as corner faces) when shape optimization is used to build the initial database.

The lifted mesh is conformally flattened with minimal distortion. We uniformly resample the boundary of this mesh targeting a total number of edges M_b and triangulate the interior using Delaunay triangulation with bounded maximal triangle area. Finally, vertices of this mesh are mapped back to the parameter domain and lifted to the initial Bézier patch. As a result, we obtain an initial configuration for a deformed glass panel conforming to Bézier boundaries and its corresponding undeformed configuration.

Minimization. The initial solution is not in static equilibrium and has arbitrarily high stresses. We compute the minimal energy configuration by minimizing the discrete strain energies defined in Equations 3.3 and 3.5 over deformed $\mathbf{x}$ and undeformed $\bar{\mathbf{x}}$ configurations. We refer to the vector of all deformed nodes at the boundary and the internal nodes as $\mathbf{b}$ and $\mathbf{i}$ respectively. To reduce the complexity of the problem and keep a high quality triangulation of the undeformed configuration, we assume internal nodes at the rest configuration $\bar{\mathbf{i}}$ are computed through Laplacian

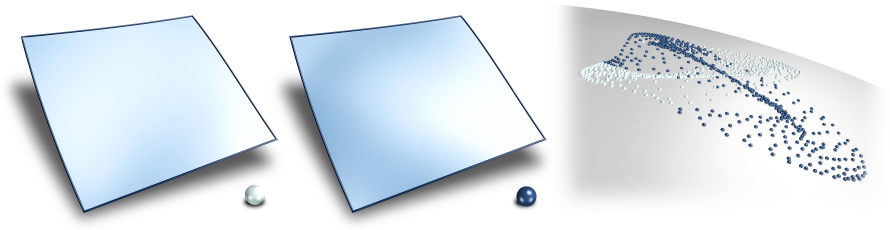


Figure 3.7: Comparison between two alternative stable equilibria for a given Bézier boundary. The two resulting panels produce radically different Gauss maps (right) leading to very distinguishable reflection effects.

smoothing of the boundary vertices $\bar{\mathbf{i}} = \mathbf{L}\bar{\mathbf{b}}$. Then, the aforementioned minimization problem results:

$$\min_{\mathbf{i},\phi,\bar{\mathbf{b}}} W(\mathbf{x}, \phi, \bar{\mathbf{b}}) + R(\bar{\mathbf{b}}), \tag{3.6}$$

where W is the sum of all strain energy terms and R is a regularization term removing the null-space due to the translation and rotation of the undeformed configuration. Note that we only consider undeformed boundary nodes $\bar{\mathbf{b}}$ as variables of the optimization; after each solver iteration, we project internal nodes coordinates $\bar{\mathbf{i}}$ through Laplacian smoothing. In addition, boundary nodes of the deformed configuration remain fixed and conformal to Bézier boundaries.

As it can be seen in Figure 3.7, minimizing Equation 3.6 does not always produce a unique solution. For a given boundary, glass panels can potentially adopt multiple stable equilibria corresponding to locally optimal shapes that depend on the initialization of the problem. While for some boundary curves, there is a clearly preferred shape which is more energetically stable than the rest, in other cases, several stable equilibria are valid solutions that might be practically used in a feasible panelization. Furthermore, maximum stress levels differ a lot between stable configurations. Multistability impose two challenges for building a data-driven model of glass panel mechanics. First, we do not know in advance the number of local minima that exist for a given boundary nor how energetically stable these configurations are in practice; and second, we do not know how to initialize the minimization problem in order to obtain such solutions. Both challenges motivated the use of a MDN as a regressor for the shape and corresponding stress of the glass panels. In Section 3.4, we describe our regression model and the methodology we followed to enrich the database by discovering new stable equilibria through an iterative process.

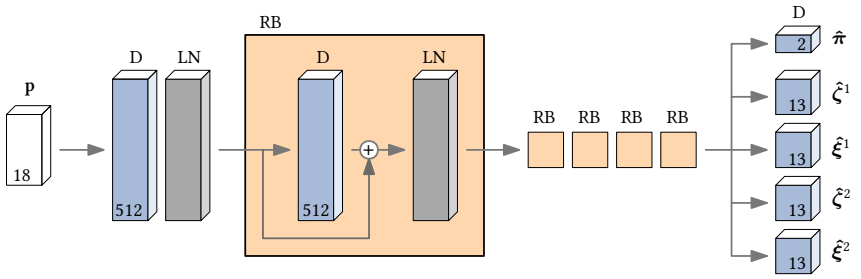
3.3.4 Failure criterion

To estimate whether the panel is going to break, we compute the maximal engineering stress across all the elements of the discretization. We estimate the stress of an element by computing the first Piola-Kirchhoff stress tensor $\mathbf{P} = \mathbf{FS}$. Here, $\mathbf{F}$ is the deformation gradient of the element and $\mathbf{S}$ is the corresponding second Piola-Kirchhoff stress tensor. In a similar fashion to [Pfaff et al., 2014], we compute the total stress of a panel using our estimation of the combined bending and membrane strain introduced in Equation 3.1:

$$\mathbf{S}\big(\mathbf{E}(\bar{\mathbf{E}}, \hat{\mathbf{E}}, z)\big) = \lambda \operatorname{Tr}(\mathbf{E})\mathbf{I} + 2\mu\mathbf{E}. \tag{3.7}$$

The maximal engineering stress is then evaluated as the maximum absolute singular value of $\mathbf{P}$ across all the elements. That is, for each element, we compute $\mathbf{S}\big(\mathbf{E}(\bar{\mathbf{E}}, \hat{\mathbf{E}}, \pm h/2)\big)$, where the bending contribution to the stress becomes maximum, and pick the highest absolute singular value. The global maximal stress value is generally at most C^0-continuous with respect to the panel boundary curves which makes its direct usage in a continuous optimization undesired. Instead, we compute an L_p-norm of maximal principal stress per element. In practice, we found that $p = 12$ suffices. We denote the resulting value σ and refer to it as *maximal stress* for brevity.

Taking our assumptions, it is important to note that neither the overall shape nor the maximal stress value changes for a given panel under uniform scaling. This implies that only the ratio of the thin shell dimensions and the panel thickness matters. For simplicity, we choose 1 mm as our canonical thickness for the simulations and scale the obtained results accordingly for every other target thickness.



D: dense layer, LN: layer normalization, RB: residual block

Figure 3.8: Architecture of our data-driven model. The input is a panel boundary $\mathbf{p}$; the model predicts means $\hat{\boldsymbol{\zeta}}^k$, variances $\hat{\boldsymbol{\xi}}^k$, and component weights $\hat{\boldsymbol{\pi}}$ for a two-component Gaussian mixture over the shape and stress of the minimal-energy surface. Numbers in dense layers indicate the number of output units.

3.4 Data-driven model

We require a model that can efficiently predict the shape and stress of the minimum-energy panels for a given boundary. The simulation described in Section 3.3 calculates these quantities, but is too slow to incorporate in an interactive design tool. Our data-driven model aims to predict the deformed shape and corresponding maximum stress of the panels more efficiently. Moreover, it will allow us to calculate its derivatives with respect to the input boundary, which is required for gradient-based design optimization.

We therefore learn a statistical model that maps panel boundaries to the shapes and stresses of minimal-energy conforming cold bent glass surfaces. Section 3.4.1 describes the model and training process. Training requires a large dataset of boundaries and the resulting panel shapes and stresses; in Section 3.4.2 we describe the space of boundaries we sample from, and how the shape optimization and stress computation of Section 3.3 is applied to them. To improve the results further, we augment the dataset to better cover regions of the input space where predictions do not match the training data due to multistability of the glass panels (see Section 3.4.3), and retrain the model on this enriched dataset.

3.4.1 Multi-modal regression model

Our prediction model takes as input a vector $\mathbf{p} \in \mathbb{R}^{18}$ representing a panel boundary. As noted in Section 3.3.3, several different surfaces may conform to a given boundary, corresponding to different local minima of the strain energy. Predicting a single output therefore yields poor results, typically the average over possible shapes. Instead, we use a *mixture density network* (MDN)—a neural network model with an explicitly multi-modal output distribution [Bishop, 2006]. For a given boundary, each mode of this distribution should correspond to a different conforming surface.

Whereas training a neural network to minimize mean squared error is equivalent to maximizing the data likelihood under a Gaussian output distribution, an MDN instead maximizes the likelihood under a Gaussian mixture model (GMM) parameterized by the network. It must therefore output means and variances of a fixed number K of mixture components, as well as a vector $\hat{\boldsymbol{\pi}}$ of component probabilities[1].

[1] In the remainder of this thesis, all variables with hats denote predictions from our data-driven model,

In our model, each component is a $(12+1)$-dimensional Gaussian with diagonal covariance, corresponding to the four interior control points of the shape, $\mathbf{P}_{i,j} \in \mathbb{R}^3$, $i, j \in \{1, 2\}$, and stress σ, of one possible conforming surface. We denote the mean of the concatenated shape and stress of the k^{th} mixture component by $\hat{\boldsymbol{\zeta}}^k$, and the variance by $\hat{\boldsymbol{\xi}}^k$; both are output by the neural network and so depend on the input boundary $\mathbf{p}$, and network weights $\mathbf{w}$.

Model architecture. We use a densely-connected model with six layers of 512 exponential linear units (ELU) [Clevert et al., 2015], with residual connections [He et al., 2016] and layer-normalization [Ba et al., 2016] at each hidden layer (see Figure 3.8). In simulation, we observed that a given boundary could potentially admit more than two stable states. However, these cases were extremely rare and we therefore set $K = 2$. This suffices to capture the vast majority of stable states observed in our dataset, and results in a low validation error. The output layer therefore has 54 units, with no activation for the means $\hat{\boldsymbol{\zeta}}^k$, exponential activation for the variances $\hat{\boldsymbol{\xi}}^k$, and a softmax taken over the mixing probabilities $\hat{\boldsymbol{\pi}}$.

Model training. The model is trained to minimize the negative log-likelihood of a training set $\mathcal{T}$ under the GMM:

$$\mathcal{L}(\mathcal{T}; \mathbf{w}) = -\sum_{(\mathbf{p},\boldsymbol{\zeta})\in\mathcal{T}} \log\left\{\sum_{k=1}^{K} \hat{\pi}^k(\mathbf{p}; \mathbf{w})\, \mathcal{N}\Big(\boldsymbol{\zeta} \,\Big|\, \hat{\boldsymbol{\zeta}}^k(\mathbf{p}; \mathbf{w}),\, \hat{\boldsymbol{\xi}}^k(\mathbf{p}; \mathbf{w})\Big)\right\} \tag{3.8}$$

where $\boldsymbol{\zeta}$ is a true output for panel $\mathbf{p}$, i.e. the concatenation of shape and stress from one simulation run, and $\mathcal{N}$ represents a diagonal Gaussian density. We also add an L2 regularization term with strength 10^{-4} on the weights $\mathbf{w}$, to discourage over-fitting.

We use the stochastic gradient method Adam [Kingma & Ba, 2014] to minimize the above loss function with respect to the network weights $\mathbf{w}$. We use a batch size of 2048, learning rate of 10^{-4}, and early stopping on a validation set with patience of 400 epochs. We select the best model in terms of the validation loss obtained during the training process. A single epoch takes approximately 30 seconds on a single NVIDIA Titan X graphics card, and in total training takes around 20 hours.

Model output. For brevity in the remainder of the thesis, for a given panel boundary $\mathbf{p}$ and for a possible state $k \in \{1, 2\}$, we write:

- $\hat{\mathbf{S}}^k_{\mathbf{p}} : [0, 1]^2 \to \mathbb{R}^3$ for the Bézier surface patch which is defined by the boundary $\mathbf{p}$ and the predicted interior control nodes from the mean of the k^{th} component (i.e. the leading 12 elements of $\hat{\boldsymbol{\zeta}}^k$).
- $\hat{\sigma}^k_{\mathbf{p}}$ for the stress value (i.e. the last element of $\hat{\boldsymbol{\zeta}}^k$).
- $\hat{\pi}^k_{\mathbf{p}}$ for the k^{th} component probability $\hat{\pi}^k(\mathbf{p}; \mathbf{w})$.

Furthermore, we write $\hat{\mathbf{S}}_{\mathbf{p}}$ and $\hat{\sigma}_{\mathbf{p}}$ (i.e. without the k superscript) to refer to the *best* prediction for boundary $\mathbf{p}$, which is determined by two factors:

1. if any of the component probabilities $\hat{\pi}^k_{\mathbf{p}}$ is greater than 95% we discard the alternative and define the corresponding shape/stress prediction as best, or

as opposed to values from the physical simulation.

2. otherwise, the best is determined depending on the application, either as the lower stress, the smoother shape, or the closer shape to a reference surface.

We discard components with $\hat{\pi}_{\mathrm{p}}^{k} < 0.05$ since the modes with near-zero probability imply a low level of confidence in the corresponding prediction.

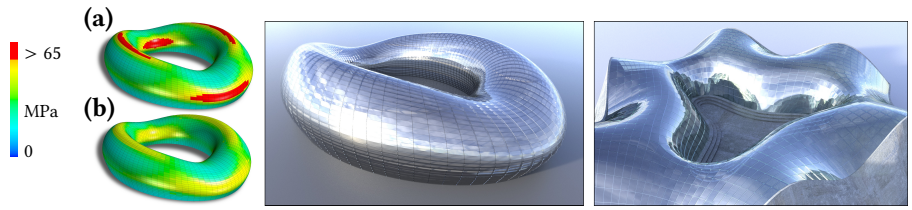


Figure 3.9: Left: Initial design includes panels exceeding stress limits **(a)** and is optimized for stress-reduction **(b)**. Right: two different façades designed with our tool.

3.4.2 Dataset construction

In order to train our prediction model, we require a dataset of boundaries that is representative of our target application. These are then paired with the shapes and stresses of conforming surfaces with minimal energy. Recall from Section 3.2 that a panel boundary may be parameterized invariantly to rigid transformations, by corner pairwise squared-distances $\boldsymbol{d}$, edge-plane inclinations $\boldsymbol{\gamma}$, and halfedge tangent directions $\boldsymbol{\theta}$. We generate boundaries by sampling these parameters from the ranges and distributions described in Appendix A. Note that the physical model for the deformed shape and stress is invariant under scaling of all geometric magnitudes; we choose our sampling ranges so it would be possible to scale the results to panel length-to-thickness ratios commonly used in cold bent glass façades (e.g., 150–600). By applying the shape optimization described in Section 3.3 to these boundaries, we obtain fine discrete meshes representing the deformed cold bent panels. We obtain a Bézier representation of such panels by keeping the sampled boundaries and fitting interior control points to match the simulated surface using the method described in Section 3.2.1. Plus, in our representation, any non-flat panel geometry can be represented in four alternative ways depending on vertex indexing and can be mirrored remaining valid. We therefore transform each simulated panel into eight samples by permuting the vertex indexes and adding their mirror-symmetric representations.

In total, we simulated approximately 1.5 million panels, which corresponds to 12 million samples after vertex permutations and adding mirror-symmetric panels. We reserved 10% of these samples as a validation set for tuning the optimization hyperparameters and network architecture. To acquire such large amounts of data requiring massive computations, we employed cloud computing.

3.4.3 Dataset enrichment

When a given boundary has multiple conforming panels, the physical simulation returns only one of these determined by the twist-free Bézier patch initialization. Conversely, our data-driven model always predicts $K = 2$ states, though one may have a very small mixture weight $\hat{\pi}^{k}$ indicating it is unlikely to be a valid optimal panel. We observed that after training, the model often predicts shapes for boundaries in its training set, that differ from those returned by the simulation—however,

re-simulating these boundaries with different initialization recovers a solution close to that predicted by the data-driven model. This observation suggests a method to extend the dataset with new samples to improve prediction error.

Specifically, we use the prediction from the model as an initialization for the simulator, which is then likely to converge to a stable surface that was not reached from the default initialization. The resulting surface can be added to the training set, so after retraining, the model will give an even more accurate prediction in the same region of parameter space. We apply the data-driven model to every panel in the training set, and collect the predicted shapes $\hat{S}_{\mathrm{p}}^{k}$, $k \in \{1, 2\}$ where $\pi^{k} > 5\%$. For each of these, we calculate the maximum deviation of the internal control points along any dimension, from the true shape in the training set. We then retain the 200K panels (∼15% of the original training set) for which this deviation is largest. For each such boundary, we re-run the simulation, using the predicted shape $\hat{S}_{\mathrm{p}}^{k}$ as the initialization. Finally, we select all panels which have at least 2 mm difference along any dimension of any internal control node compared to the panel obtained originally, and add these to the training set. The resulting, enriched training set is used to retrain the model.

3.5 Interactive design

In this section, we show how we arrive at a practical interactive design tool for freeform surface panelization with cold bent glass panels. We aim at a tool compatible with the standard design workflow of an architectural designer. At every moment during the editing process, the user gets immediate feedback on the physical properties of the panelization (i.e., shape and stress predictions for the panels). Upon request, an automated process running at interactive times uses optimization to "guide" the design. Figures 3.2 and 3.9 show two different double-curved glass surfaces that have been interactively designed from scratch using our tool. While it is generally desired to create designs free from breaking panels, in a real project, one might like to assume the cost of hot-bending a small proportion of the panels. Therefore, there is a practical trade-off between the smoothness and aesthetics of a design and its manufacturability. We consider this option by explicitly weighting various design criteria in the formulation of our inverse design problem.

3.5.1 Optimization setup

Depending on the specific application domain, desired properties might vary. This translates into the minimization of a composite target functional $\mathcal{E}$ through the following optimization problem.

Optimization problem 4 Cold bent glass panelization

$$\text{minimize} \quad \mathcal{E} = w_{\sigma}\mathcal{E}_{\sigma} + w_{\mathrm{s}}\mathcal{E}_{\mathrm{s}} + w_{\mathrm{f}}\mathcal{E}_{\mathrm{f}} + w_{\mathrm{p}}\mathcal{E}_{\mathrm{p}} + w_{\mathrm{c}}\mathcal{E}_{\mathrm{c}}$$

Overall, the total energy $\mathcal{E}$ depends on the vertex positions $\mathbf{v} \in V$, the edge plane vectors $\mathbf{s}_e$ defining the plane Π_e associated with e, the tangent angles θ_i and some auxiliary variables associated with inequality constraints. Each weighted contribu-

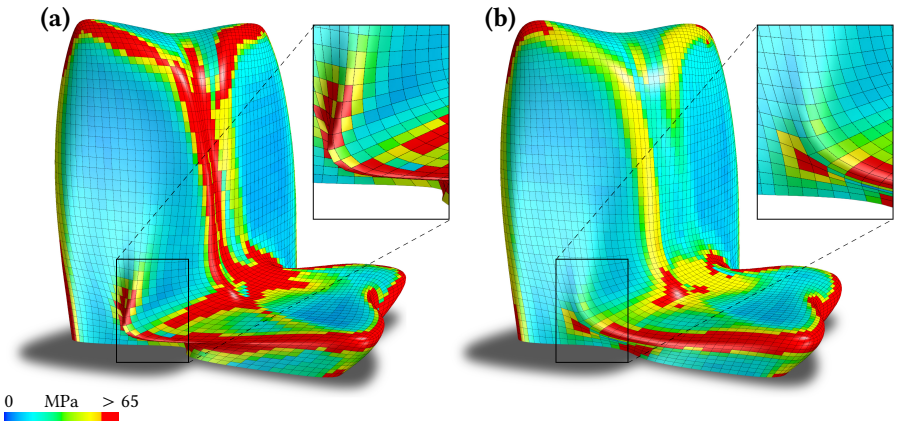


Figure 3.10: Optimization of the NHHQ skyscraper design (Zaha Hadid Architects). We first optimize for smoothness of the overall design, and then optimize selected high stress areas for stress reduction. **(a)** Stress on panels computed on the original shape and panel layout. Red panels exceed the threshold of 65 MPa. **(b)** Stress on panels after optimization. The inset shows an area with clearly visible shape change. We decrease the number of panels exceeding 65 MPa from 1517 to 874.

tion to $\mathcal{E}$ represents a desired property of the final design, which we discuss in detail in the following sections.

Panel stress $\mathcal{E}_\sigma$. The most important property is the manufacturability of the final design. Failure in a specific type of glass is modelled by estimating the maximal stress present in the glass panel and comparing it to the maximum allowed stress value.

The MDN from Section 3.4 acts as a stress estimator. We constrain the predicted stress value $\hat{\sigma}_\mathbf{p}$ for a given boundary $\mathbf{p}$ to be less than a stress bound $\sigma_{\max}$. We assign $\sigma_{\max}$ to a value lower than the stress value at which failure occurs, taking into account a safety factor and the estimator error. The inequality constraints $\hat{\sigma}_\mathbf{p} \leq \sigma_{\max}$ are converted to equality constraints by introducing an auxiliary variable $u_\mathbf{p} \in \mathbb{R}$ per panel boundary $\mathbf{p}$, and formulating the manufacturability energy as

$$\mathcal{E}_\sigma = \sum_\mathbf{p} (\hat{\sigma}_\mathbf{p} - \sigma_{\max} + u_\mathbf{p}^2)^2. \tag{3.9}$$

Figure 3.13 shows the effect of limiting the maximum stress of the design for a section of the façade of the Lilium tower.

Smoothness $\mathcal{E}_\mathbf{s}$. Here we collect some terms in the final objective function which aim in various ways at panelizations which are as smooth as possible. As shown in Figure 3.11, this is essential for achieving the stunning look of curved glass façades as it greatly affects the reflection pattern. The smoothness term is the sum of two individual functionals, i.e. $\mathcal{E}_\mathrm{s} = \mathcal{E}_1 + \mathcal{E}_2$.

Kink angle smoothing $\mathcal{E}_1$. It is in general not possible to get smooth connections along common boundary curves of panels, but we can try to minimize the kink angle. For each pair of faces f_i, f_j sharing a common edge e, we consider their respective predicted panels $\hat{\mathrm{S}}_i$, $\hat{\mathrm{S}}_j$ and minimize the angle between their surface normals $\mathbf{n}_i$, $\mathbf{n}_j$

evaluated at the parameter $t = 0.5$ of the shared curve

$$\mathcal{E}_1 = \frac{1}{10} \sum_{e \in E_I} (1 - \mathbf{n}_i \cdot \mathbf{n}_j)^2, \tag{3.10}$$

where E_I is the set of interior edges of $\mathcal{M}$, and $1/10$ is an appropriate importance weight within the smoothness term. Note that $\hat{\mathbf{S}}_i$, $\hat{\mathbf{S}}_j$ are shape predictions for the respective boundary curves of the two faces f_i, f_j, and thus optimization involves computing the Jacobian of the MDN output w.r.t. the input boundaries. Figure 3.12 shows the effect of including the kink smoothing term in the design of the NHHQ façade.

Curve network smoothing $\mathcal{E}_2$. Each edge in the dominant mesh polylines of $\mathcal{M}$ determines a cubic patch boundary curve, and the sequence of these curves should be also as smooth as possible. At each connection of two edges, the corresponding tangents should agree, and thus the inwards directed unit tangent vectors satisfy $\mathbf{t}_i = -\mathbf{t}_j$, or equivalently $\mathbf{t}_i \cdot \mathbf{t}_j + 1 = 0$. This tangent continuity constraint explains the first part in the smoothness term

$$\mathcal{E}_2 = \sum (\mathbf{t}_i \cdot \mathbf{t}_j + 1)^2 + \sum \left[\mathbf{s}_e \cdot (\mathbf{n}_i + \mathbf{n}_{i+1})\right]^2 . \tag{3.11}$$

The second part concerns the planes Π_e. We consider an edge e with endpoints $\mathbf{v}_i$, $\mathbf{v}_{i+1}$. The discrete osculating plane at $\mathbf{v}_i$ is spanned by $(\mathbf{v}_{i-1}, \mathbf{v}_i, \mathbf{v}_{i+1})$ and has a unit normal $\mathbf{n}_i$. Likewise, $(\mathbf{v}_i, \mathbf{v}_{i+1}, \mathbf{v}_{i+2})$ define a discrete osculating plane with normal $\mathbf{n}_{i+1}$ at $\mathbf{v}_{i+1}$. We want Π_e to be the bisecting plane between these two, i.e. $\mathbf{s}_e \cdot (\mathbf{n}_i + \mathbf{n}_{i+1}) = 0$. Of course, the sums are taken over all occurrences of the described situations.

Finally, in practice, a few other parts are added to the smoothness term $\mathcal{E}_s$ which concern special cases. At combinatorially singular vertices of $\mathcal{M}$ we constrain the tangent vectors to lie in a tangent plane. Plus, there are various symmetry considerations which are used at the boundary, but those could easily be replaced by other terms with a similar effect.

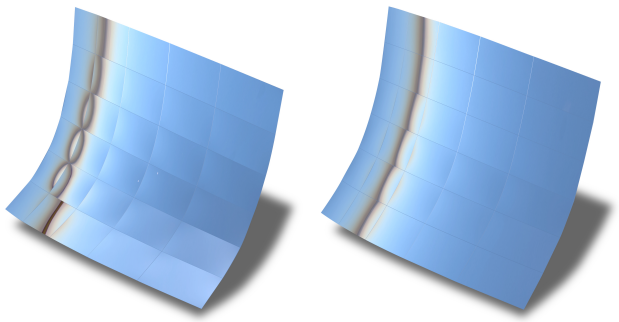


Figure 3.11: Effect of optimization on visual smoothness. On the left, a selection of cold bent panels computed on a given layout. On the right, the same panels after optimization of the layout for kink angle and bending stress reduction.

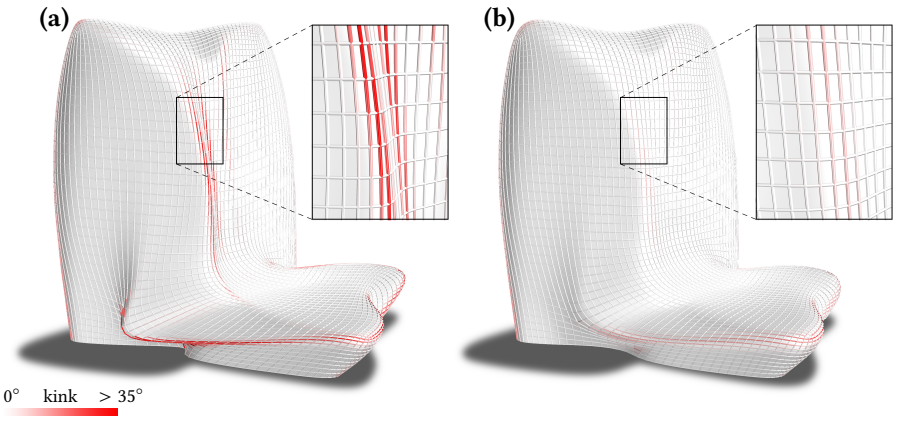


Figure 3.12: Comparison of the kink angle between panels in the NHHQ model, before **(a)** and after **(b)** running our design optimization algorithm. As a result, the mean kink angle is lowered from 3.7° to 2.7°, while the maximum was reduced from 60.9° to 36.0°.

Mesh fairness $\mathcal{E}_\mathbf{f}$. So far we have dealt with smoothness of the panelization to a given mesh $\mathcal{M}$. Since we also allow the mesh $\mathcal{M}$ to change during design, we need to care about its fairness. This is done in the standard way using second-order differences of consecutive vertices along dominant mesh polylines,

$$\mathcal{E}_\mathrm{f} = \sum (\mathbf{v}_{i-1} - 2\mathbf{v}_i + \mathbf{v}_{i+1})^2. \tag{3.12}$$

Proximity to reference mesh $\mathcal{E}_\mathbf{p}$. When designing a panelization for a given reference geometry, it is not sufficient to have the mesh $\mathcal{M}$. One will usually have a finer mesh $\mathcal{M}_\mathrm{ref}$ describing the reference geometry (Figure 3.4). In order to let $\mathcal{M}$ change, but stay close to the reference surface, we need a term which allows for the gliding of $\mathcal{M}$ along $\mathcal{M}_\mathrm{ref}$. This is done in the familiar way: to let a vertex $\mathbf{v_i}$ stay close to $\mathcal{M}_\mathrm{ref}$, we consider its closest point $\mathbf{v}_i^*$ on $\mathcal{M}_\mathrm{ref}$ and the unit surface normal $\mathbf{n}_i^*$ at $\mathbf{v}_i^*$. In the next iteration, $\mathbf{v_i}$ shall stay close to the tangent plane at $\mathbf{v}_i^*$, which is expressed via

$$\mathcal{E}_\mathrm{p} = \sum_{\mathbf{v}_i \in V} \left[(\mathbf{v_i} - \mathbf{v}_i^*) \cdot \mathbf{n}_i^* \right]^2. \tag{3.13}$$

Design space constraints $\mathcal{E}_\mathbf{c}$. Since we want the neural network to produce reliable estimates, we need to ensure panel boundary curves remain within the range used for training (Section 3.4.2). This is achieved as the sum of two constraint functionals $\mathcal{E}_\mathrm{c} = \mathcal{E}_3 + \mathcal{E}_4$. First, we constrain the tangent angles to $|\theta_i| = \angle(\mathbf{t}_i, \mathbf{e}_i) \leq 4.9°$ for all angles θ_i of halfedges $\mathbf{e}_i$ with tangent vectors $\mathbf{t}_i$. We again convert inequality constraints to equality constraints by introducing auxiliary variables u_i,

$$\mathcal{E}_3 = \sum (\theta_i^2 - (4.9°)^2 + u_i^2)^2. \tag{3.14}$$

Second, we are working under the assumption that the vectors $\mathbf{s}_e$ are unitary and orthogonal to their respective edges e, which results

$$\mathcal{E}_4 = \sum_e [(\mathbf{s}_e \cdot \mathbf{e})^2 + (\mathbf{s}_e^2 - 1)^2]. \tag{3.15}$$

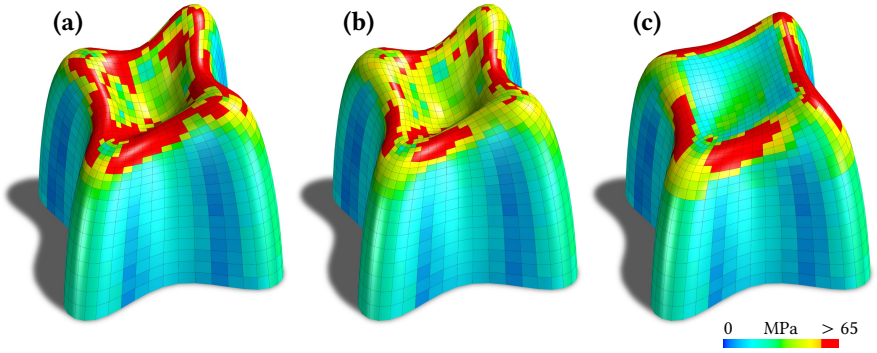


Figure 3.13: Optimization of the Lilium Tower (model by Zaha Hadid Architects) for different target properties. **(a)** Stress values for the initial panel layout. **(b)** Optimizing the design only for stress reduction and proximity to the original design leads to more panels within the stress threshold, but also to a non-smooth curve network. **(c)** Allowing the design to deviate from the input and including fairness, produces a smoother result with reduced stress. Number of panels exceeding 65 MPa is respectively 293, 131, and 225

3.5.2 Initialization

The edge plane vector $\mathbf{s}_e$ of an edge e is initialized so that Π_e is the bisecting plane of two discrete osculating planes, as in the explanation of (Equation 3.11). The angles θ_i are initialized so that they are at most 5° and the tangents lie as close as possible to the estimated tangent planes of the reference geometry. After initializing all other variables and computing an estimated stress value per face panel, the auxiliary variables are initialized such that they add up to the inequality constraint bound, or zero otherwise (i.e. the inequality constraint is not satisfied). The shape $\mathbf{S_p}$ of each panel is initialized with the MDN prediction using the initial boundary parameters $\mathbf{p}$. In case there are two possible shapes, we use the one that provides the best solution considering application dependent criteria (e.g., stress reduction). When looking for the smoothest fit, we pick the one minimizing $\sum(1-\mathbf{n}_i \cdot \mathbf{n}_e)^2$, i.e., a measure of angle deviation between each edge normal (sum of two adjacent face normals orthonormalized to $\mathbf{e}$) and the surface normal $\mathbf{n}_i$ evaluated at the parameter $t = 0.5$ of the edge curve.

3.5.3 Optimization solution

The minimization of $\mathcal{E}$ results in a nonlinear least-squares problem that we solve using a standard Gauss-Newton method, similarly to Chapter 2. Derivatives are computed analytically and, since each distinct term of $\mathcal{E}$ has local support, the linear system to be solved at each iteration is sparse. We employ Levenberg-Marquardt regularization and sparse Cholesky factorization using the TAUCS library Toledo, 2003.

Optimization weights. The weights associated with the target functional $\mathcal{E}$ act as handles for the designer to guide the output of the optimization to the desired result. We do not opt for a fixed weight configuration since the ideal balance is not uniquely defined, but is instead governed by project-dependent factors such as budget and design ambition.

In all our experiments we found it sufficient to assign the weights either to zero or to values $10^{\{-2,-1,0\}}$. Figure 3.13 shows one example of the different effects possible when changing the property importance. In practice, and as a rule of thumb for a standard optimization where we prioritize stress reduction and smooth panels (in that order), we use weight values $w_\sigma = 1$, $w_\mathrm{s} = w_\mathrm{c} = 10^{-2}$, and $w_\mathrm{p} = w_f = 10^{-1}$. We also reduce the fairness importance at the i-th optimization iteration by scaling its weight by 0.9^i.

3.6 Experiments and results

3.6.1 Experimental validation

We experimentally validated our simulation results and design workflow. For practical reasons the experiments were done at a small scale using borosilicate thin glass of about 180×130 mm^2 and 0.35 mm thickness.

For the validation of the simulation results (see Figure 3.15) high precision frames were machined from cast aluminum. The glass panel is pressed down on a 2 mm wide smooth support surface by a dense array of stainless steel finger springs cushioned by 0.5 mm Polytetrafluoroethylene (PTFE). The support frame matches a thin boundary strip of the simulated glass panel. We 3D-scanned the shape and the obtained surface was registered to the output of our shape optimization routine. We observed a worst-case deviation of 0.12 mm, see Figure 3.15. Note that we registered an offset surface from the optimal mid-surface to account for the glass thickness.

The frames for the design model illustrated in Figure 3.14 were built from laser cut and welded 1.2 mm thick stainless steel sheet metal. The glass, cushioned by tape, is pressed down on to the frame by L-shaped stainless steel fixtures spot welded to the frame. The presented design model is negatively curved and consists of nine

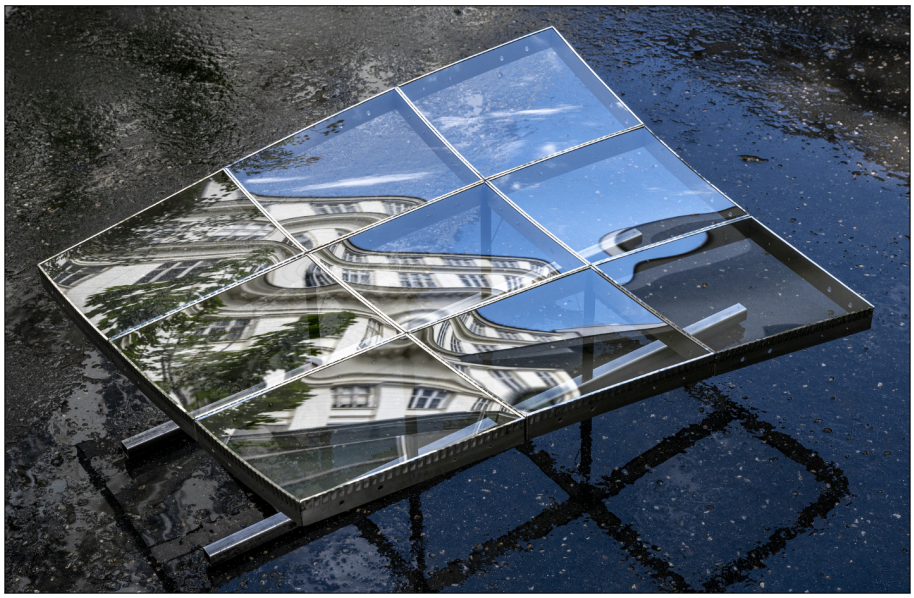


Figure 3.14: Realization of a double-curved surface using 3×3 cold bent panels.

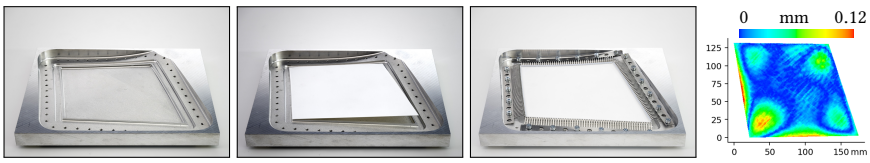


Figure 3.15: A double-curved panel of thickness 0.35 mm with off-plane corner deviation of 6.9 mm. Right: deviation from the simulation by at most 0.12 mm.

individual panels, each about 200×170 mm^2 in size. The expected stress levels range from 20 to 62 MPa.

3.6.2 Validation of data-driven model

Our data-driven model (Section 3.4) must reproduce the output of the physical simulation model efficiently and accurately. To evaluate its accuracy, we generate a test set of 10K panel boundaries and use the data-driven model to predict the conforming surfaces. We consider only admissible surfaces, i.e. those with predicted probability at least 5%. The surface predictions are used to initialize our physical shape optimization routine to obtain the true shape and stress values for comparison. We evaluate shape prediction on panels with maximal stress below 65 MPa, which results in mean absolute error (MAE) $\sim$ 0.5 mm. Note that this is significantly less than the assumed 1 mm thickness of the glass. We evaluate stress on panels whose true maximal stress is in the range 50–65 MPa (our region of interest); our predictions have MAE of $\sim$ 2.9 MPa.

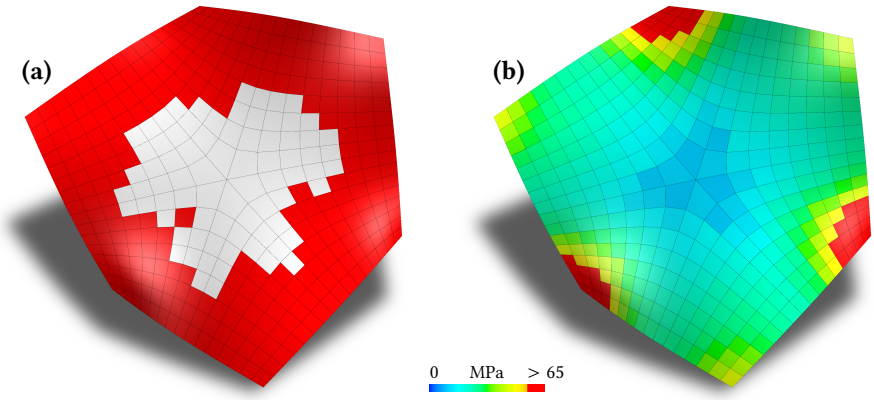


Figure 3.16: Bent glass capabilities. **(a)** A quadrilateral mesh where red faces exceed a deviation of planarity of 0.02 (measured as the distance between diagonals divided by average edge length), and are therefore not suitable for a flat glass panelization. **(b)** A cold bent panelization with corresponding face stresses. The stress values for the six central panels have been computed via simulation since they were outside the MDN input domain. According to a stress limit of 65 MPa, most panels optimized are feasible. The resulting cold bent panelization is shown in Figure 3.17.

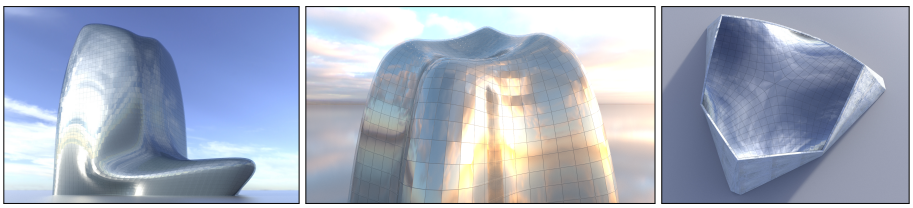


Figure 3.17: Dominant cold bent glass realizations of the NHHQ model (left), the Lilium Tower (center) after optimization for smoothness and stress reduction. The surface from Figure 3.16 as an architectural design (right). Panels exceeding the maximum stress (check Figures 3.10, 3.13, 3.16) are realized with hot bending.

3.6.3 Applications

From a manufacturing point of view, the simplest solution to clad architectural surfaces it the use of planar panels. However, this simplification sacrifices the visual smoothness of the surface. Moreover, planar panels impose a restriction on the panels layout, and in negatively curved areas there is often no other choice than to follow the principal curvature directions of the surface. On the other hand, a panelization with double-curved panels is often prohibitive, due to the high production cost of custom molds. Cold bent glass can be then a suitable solution. In Figure 3.3, we compare the visual appearance of the smoothest possible panelization achievable with planar panels with a cold bent one, while in Figure 3.16, we show a panelization layout that is mostly feasible with cold bent glass, but not with planar panels. In the following, we illustrate how users can employ our workflow for architectural panelization and design.

Façade panelization. In this case, the input is a quadrilateral mesh that encodes both the design shape and the panel layout. Once the edges of the input mesh are smoothed via cubic Bézier curves, we can predict panels' shapes and stresses. Those panels exceeding the failure criterion shall be realized with custom molds. At this point, tuning the weights described in Section 3.5.3, the user can optimize the shape for the reduction of stress and kinks between panels, choosing an appropriate compromise between fidelity to the original shape, number of custom molds needed, and visual smoothness (see Figures 3.10, 3.12, and 3.13). To show cold bent glass capabilities in façade panelization, we tested this workflow on the challenging NHHQ and Lilium Tower models by Zaha Hadid Architects, which were never realized. Results are shown in Figure 3.17.

Façade design. Besides the panelization of a given shape, our workflow is very well suited as an interactive design tool. In this case, the user can interactively modify the quad mesh representing the panel layout and gets immediate feedback on which panels can be produced with cold bent glass, while exploring different designs. Estimation times are compatible with an interactive design session. Once the user is satisfied with a first approximate result, the panelization can be further optimized, as described in Section 3.6.3, to improve smoothness and reduce panel stresses. In this step, we can further reduce the number of panels which are not feasible for cold bending. Figures 3.2, 3.9, and 3.16 show some sample architectures designed with this procedure. In particular, Figure 3.18 demonstrates a screenshot from the design tool in use.

All interactive design sessions were performed on an Intel® Core™ i7-6700HQ CPU at 2.60 GHz and NVIDIA GeForce GTX 960M. The MDN is implemented in TensorFlow 2.1 and is run on the GPU. For 1K panels, prediction time is 0.1 seconds while optimization averages to 3 seconds per iteration. We usually deal with less panels since we target selected high-stress areas of the overall design. A total of 10–20 iterations are enough for the desired results. In comparison, our shape optimization as described in Section 3.3 implemented in C++ and using the IPopt optimization library [Wächter & Biegler, 2006] with code-generated derivatives takes around 35 seconds on average for a single panel with $\sim 10^3$ elements.

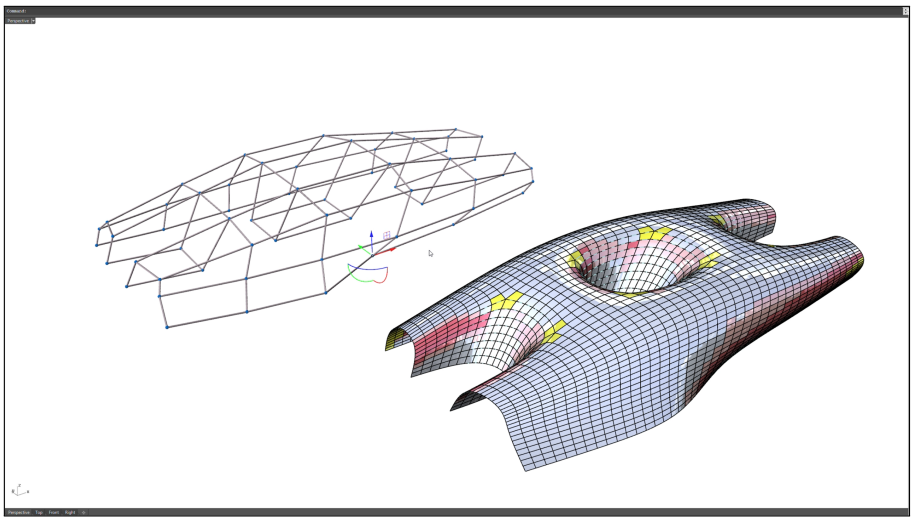


Figure 3.18: Screenshot from an interactive session with our design tool, which was integrated to Rhino. In this case, we have a panelization (right) derived from Catmull-Clark subdivision of a control mesh (left). The tool provides in real-time predictions for the panel shape in the form of cubic Bézier surfaces, and for the maximal stress of the panel; here color-coded as safe (blue), critical (pink), and breaking (red). Yellow panels are ignored as their shape is outside the prediction domain of the model.

3.7 Discussion

We have introduced an interactive, data-driven approach for material-aware form finding of cold bent glass façades. It can be seamlessly integrated into a typical architectural design pipeline, allows non-expert users to interactively edit a parametric surface while providing real-time feedback on the deformed shape and maximum stress of cold bent glass panels, and automatically optimize for fairness criteria while maximal stresses are kept within glass limits. Our method is based on a deep neural network architecture and multi-modal regression model, overcoming the limitation of traditional simulation and optimization approaches for glass where the computational complexity is prohibiting interactivity. By coupling geometric design and fabrication-aware design, we believe our system will provide a novel and practical workflow, allowing to efficiently find a compromise between economic, aesthetic, and engineering aspects.

Identifying such a compromise usually involves multiple competing design goals. While we have demonstrated the applicability of our system for several design criteria, it would be interesting to extend the design workflow by adding capabilities, for example, for strictly local edits, marking some panels as a priori hot bent, or specifying kink edges. Due to our differentiable network architecture, in theory it should be trivial to incorporate additional criteria to our optimization target functional or even employ a different numerical optimization algorithm if desired.

Similar to all data-driven techniques, we should only expect accurate predictions from our network if similar training data was available. Surprisingly, we noted that we were able to discover stable states that we initially did not find with the traditional optimization approach, and used these to enrich our database. However, we cannot guarantee that our database contains all relevant stable states and that all of them will be predicted. Identifying *all* stable states and optimally sampling the database using this information would be an interesting avenue for future work. For fabrication, in our experiments reproducing the desired particular state was trivial and emerged when intuitively attaching the glass to the frame.

In the presence of more than one potential state, our system currently selects in each iteration per panel the state that best fits our application dependent criteria. An alternative would be to compute a global, combinatorial optimal solution among all potential states. However, due to the combinatorial complexity, this would result in a much harder and probably computationally intractable optimization problem. We also considered solving the combinatorial problem by using a continuous relaxation, but ultimately did not find evidence in our experiments that would indicate the need for such an approach as we observed stable convergence to satisfactory results. However, identifying the global minimum would nevertheless be an interesting research challenge.

We believe our workflow could serve as an inspiration for many other material-aware design problems. For future work, it would be exciting to explore extensions to different materials, for instance metal, wood, or programmable matter that can respond to external stimuli, such as shape memory polymers or thermo-reactive materials.

Conclusion 4

In this thesis we have investigated two important instances of panelization problems. We briefly summarize the main contributions before discussing some limitations of the research, and providing the reader with an outlook to future work.

4.1 Summary

In Chapter 2, we aimed to cover an architectural freeform surface with specific types of developable or nearly developable panels that are relevant to architecture. This is an important problem that fits into the broader area of constraint-based modeling, and expands upon previous work by allowing the curve network of the panelization to be variable during optimization. We achieved this by introducing a novel method for increasing the developability of a given surface; in our case B-spline surfaces. The method is based on a property of developable surfaces that has not been used in this setting before, namely that they possess 1−dimensional Gauss images. We proved that they are locally well-approximated by developable surfaces with planar Gauss images and formulated this fact as an optimization problem. In particular, we introduced an energy functional that measures the deviation of the surface from possessing this property, and subsequently aimed to minimize it. This method was then applied to the panelization problem by considering a grid of cubic B-spline surfaces, that represent the panels, and optimizing each panel to be a developable of a certain type. These types include (rotational) cylinders, (rotational) cones, and planar panels; all of great interest to architecture. The focus of this work was not exact developability, but rather near developability. The motivation for our approach was the fact that most materials allow for a little bit of stretching and therefore developability needs not be satisfied to a high degree.

One such material is glass which was the main focus of the second panelization problem of this thesis, investigated in Chapter 3. Toughened glass can withstand higher stresses, and therefore allows initially planar glass panels to be elastically bent and fixed at ambient temperatures to a curved frame. This cold bending process produces panels that can exhibit double curvature, providing a cost- and energy-efficient alternative of higher optical quality to traditional hot bent glass panels. We present a computational framework that achieves interactive material-aware design of cold bent glass façades. In order to capture the design space of cold bent glass panels, we performed approximately 1.5 million simulations of random curved panel

boundaries, within reasonable deformation bounds. The simulation computed panels that satisfied the curved boundary condition and were of minimal deformation energy. We discussed that the solution to the simulation optimization was not always unique, but in some cases multimodal. The resulting simulations were used to populate a database of curved boundaries and associated panel shapes and maximal stresses present in the panel. This database was then used to train a mixture-density network, a regression model capable of capturing multimodality. We investigated the validity and accuracy of the data-driven model and showed that it is sufficient for our application setting. This model allowed us to interrogate the deformation space of cold bent glass panels at interactive rates, without the need for additional simulations. This speedup was utilized in an interactive design tool, implemented in Rhino, capable of providing immediate feedback on the shapes and maximal stresses of panelizations consisting of hundreds or thousands of cold bent glass panels. By integrating the data-driven model to a constrained-based optimization problem, we were also able to optimize the cold bent glass panelizations for both manufacturability and aesthetics.

For both applications explored in this work, a plethora of results and examples are provided. These results urge us to consider what else is possible as an extension of this work, but also determine any possible limitations. We briefly discuss the latter in the following section.

4.2 Limitations

Limited by computational constraints and the complexity of the maximal stress function, we opted for a simulation model that, while sufficient for our needs, is not the industry standard for glass simulation. As an example, Datsiou [2017] uses twenty-node, quadratic, brick elements with reduced integration properties to prevent shear locking and models the glass panel with a thickness of 2 elements. While appropriate for onetime simulations, this simulation model is not viable when multiple simulations are needed—as in our case where we require an as-dense-as-possible sampling of the deformation space. This leads to a regression model which is sufficient for approximate predictions but does not provide the same guarantees as a high-accuracy simulation.

Another discussion point is the multimodal capability of the method presented in Chapter 3. While multimodality is sufficiently captured and represented in our model, it is not clear what is the subspace of target curved panel boundaries that conforms to multiple stable configurations. This is of course a general problem present in nonlinear optimization. Furthermore, multimodality can be better integrated to the design tool. The current implementation proceeds to determine the best configuration locally, if more than one are possible, at each iteration of the optimization. We tested that this is sufficient for our needs, and keeps the optimization within interactive rates. Nevertheless, an optimization approach that can handle the combinatorial complexity of multimodal configurations in a global fashion, while constraining the computation time to reasonable limits, would be ideal.

We consider the limitations mentioned here as possible future research directions. We move this discussion to the following section where we provide additional interesting questions that emerge from this research as candidates for future work.

4.3 Outlook

We mentioned near developability multiple times across this thesis. In Chapter 2 we explored nearly developable surfaces as constrained minimizers of an appropriate energy functional. In Chapter 3 we considered the achievable forms through cold bending of planar glass panels as nearly developable, since glass is a material that mostly bends but also allows for some stretching. All these are minor steps that lead us towards the geometrically largely unexplored area of nearly developable surfaces, and the class of interesting questions it contains. The proper characterization and analysis of nearly developable surfaces is a theoretically intriguing research direction for future work.

Furthermore, it should be clear to the reader that the computational framework presented in Chapter 3 is not limited to glass, but can be adapted or extended to other interesting materials, such as wood, metal, and programmable matter. Accurately capturing the design space of other materials, or structural and decorative elements, in a way that allows for its fast exploration is a research direction that we find very interesting. The recent advances in machine learning algorithms and cheap computational resources, as well as the availability of big data, have made this possible.

This observation also naturally leads to the question of what else is possible with machine learning in the architectural and manufacturing industries. A lot of problems or approaches that were discarded until recently as infeasible or prohibitive are becoming very simple with modern methodology.

As a final note, we wish to stimulate the reader's imagination with the vision of completely intuitive design tools that provide only the necessary interaction handles to the user, allowing even non-experts to experiment. Their composition is multidisciplinary, and their inception and construction are exciting research challenges for the future. Coming full circle, this vision is driven by—and justifies—the quote at the beginning of this thesis. By hiding their core mechanisms from the user, fully integrating essential properties and constraints, and automating optimization and prediction procedures, tomorrow's tools ultimately bring creativity to the forefront and allow the designer to just play.

Sampling panel boundaries

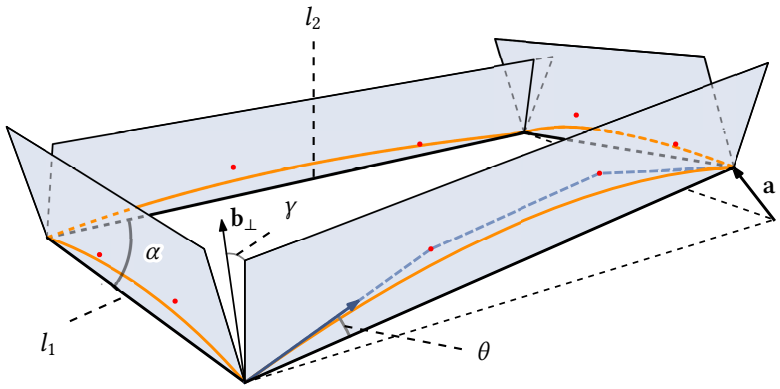


Figure A.1: Visualization of some representative random variables that define the panel boundary. Vector $\mathbf{b}_\perp$ is the rejection of the face normal $\mathbf{b}$, i.e. the normalized cross product of the diagonals of the (skew) quad, from the corresponding edge. In orange is the final sampled panel boundary.

We describe here how the panel boundaries forming the training set for our data-driven model are sampled (Section 3.4.2). We parameterize panel boundaries invariantly to rigid transformations, by corner pairwise squared distances $\mathbf{d}$, edge-plane inclinations $\boldsymbol{\gamma}$, and halfedge tangent directions $\boldsymbol{\theta}$ (Section 3.2).

In order to sample $\mathbf{d}$ such that it represents a valid quad, we start with two adjacent edge lengths l_1, l_2, an angle α between them, and a displacement $\mathbf{a}$ of the remaining vertex from the point that would form a parallelogram.

All the parameters are sampled as follows:

- Edge lengths l_1, $l_2 \sim \mathcal{U}[0.15, 0.60]$, i.e. 15–60 cm for a 1 mm thick panel,
- angle $\alpha \sim \mathcal{U}[60°, 120°]$,
- vector $\mathbf{a}$ is given by sampling a point on the unit sphere, then scaling it by a factor drawn from $\mathcal{U}[0, \min\{l_1, l_2\}/4]$,
- angle $\gamma_i \sim \mathcal{U}[-90°, 90°)$, and
- angle $\theta_i = (-1)^X \arccos Y$, where $X \sim \mathcal{U}\{0, 1\}$ and $Y \sim \mathcal{U}[\cos 5°, 1]$, so $\theta_i \in [-5°, 5°]$,

where $\mathcal{U}$ denotes the uniform distribution. Figure A.1 visualizes these parameters.

Note that our model for the deformed shape and stress is invariant under scaling of all geometric magnitudes. Our sampling ranges are chosen to allow scaling the results to thickness, edge length, and curvature ratios commonly used in cold bent glass façades.

Other publications

B

Not all research work produced during the period of this PhD was included in this thesis. Here we provide the abstracts of the excluded publications.

- **Void Filling of Digital Elevation Models with Deep Generative Models.**
  Konstantinos Gavriil, Georg Muntingh, and Oliver J.D. Barrowclough.
  IEEE Geoscience and Remote Sensing Letters, **16** (8), 1645-1649, 2019.

  Abstract. In recent years, advances in machine learning algorithms, cheap computational resources, and the availability of big data have spurred the deep learning revolution in various application domains. In particular, supervised learning techniques in image analysis have led to superhuman performance in various tasks, such as classification, localization, and segmentation, while unsupervised learning techniques based on increasingly advanced generative models have been applied to generate high-resolution synthetic images indistinguishable from real images.

  In this paper we consider a state-of-the-art machine learning model for image inpainting, namely a Wasserstein Generative Adversarial Network based on a fully convolutional architecture with a contextual attention mechanism. We show that this model can successfully be transferred to the setting of digital elevation models (DEMs) for the purpose of generating semantically plausible data for filling voids. Training, testing and experimentation is done on GeoTIFF data from various regions in Norway, made openly available by the Norwegian Mapping Authority.

- **Interpolation of syzygies for implicit matrix representations.**
  Ioannis Emiris, Konstantinos Gavriil, and Christos Konaxis.
  7th International Conference on Algebraic Informatics, 2017.

Abstract. We examine matrix representations of curves and surfaces based on syzygies and constructed by interpolation through points. They are implicit representations of objects given as point clouds. The corresponding theory, including moving lines, curves and surfaces, has been developed for parametric models. Our contribution is to show how to compute the required syzygies by interpolation, when the geometric object is given by a point cloud whose sampling satisfies mild assumptions. We focus on planar and space curves, where the theory of syzygies allows us to design an exact algorithm yielding the optimal implicit expression. The method extends readily to surfaces without base points defined over triangular patches. Our Maple implementation has served to produce the examples in this paper and is available upon demand by the authors.

Bibliography

Anzola, F. (2020). Gehry [Online; accessed under CC BY 2.0 on August 20, 2020]. URL: https://flic.kr/p/2iBvQvb

Aumann, G. (1991). Interpolation with developable Bézier patches. *Computer Aided Geometric Design*, *8*(5), 409–420. DOI: 10.1016/0167-8396(91)90014-3

Aumann, G. (2003). A simple algorithm for designing developable Bézier surfaces. *Computer Aided Geometric Design*, *20*(8-9), 601–619. DOI: 10.1016/j.cagd.2003.07.001

Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). *Layer normalization*. arXiv: 1607.06450.

Barbič, J., & James, D. L. (2005). Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Trans. Graph.*, *24*(3), 982–990. DOI: 10.1145/1073204.1073300

Beer, B. (2015). Structural silicone sealed cold-bent glass–High-rise projects experience leading to a new design concept. *GPD Glass Performance Days*, 235–240.

Bermano, A. H., Funkhouser, T., & Rusinkiewicz, S. (2017). State of the art in methods and representations for fabrication-aware design. *Computer Graphics Forum*, *36*(2), 509–535. DOI: 10.1111/cgf.13146

Bickel, B., Cignoni, P., Malomo, L., & Pietroni, N. (2018). State of the art on stylized fabrication. *Computer Graphics Forum*, *37*(6), 325–342. DOI: 10.1111/cgf.13327

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer-Verlag New York.

Cerda, E., Mahadevan, L., & Pasini, J. M. (2004). The elements of draping. *Proceedings of the National Academy of Sciences*, *101*(7), 1806–1810. DOI: 10.1073/pnas.0307160101

Chen, D., Levin, D. I. W., Sueda, S., & Matusik, W. (2015). Data-driven finite elements for geometry and material design. *ACM Trans. Graph.*, *34*(4), 74:1–74:10. DOI: 10.1145/2766889

Chen, J., Bao, H., Wang, T., Desbrun, M., & Huang, J. (2018). Numerical coarsening using discontinuous shape functions. *ACM Trans. Graph.*, *37*(4), 120:1–120:12. DOI: 10.1145/3197517.3201386

Chen, M., & Tang, K. (2010). A fully geometric approach for developable cloth deformation simulation. *The Visual Computer*, *26*(6), 853–863. DOI: 10.1007/s00371-010-0467-5

Chu, C.-H., & Chen, J.-T. (2004). Geometric design of uniform developable B-spline surfaces. *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (pp. 431–436). DOI: 10.1115/DETC2004-57257

Chu, C.-H., & Séquin, C. (2002). Developable Bézier patches: Properties and design. *Computer-Aided Design*, *34*(7), 511–527. DOI: 10.1016/S0010-4485(01)00122-1

Clevert, D.-A., Unterthiner, T., & Hochreiter, S. (2015). *Fast and accurate deep network learning by exponential linear units (ELUs).* arXiv: 1511.07289.

Datsiou, K. C. (2017). *Design and performance of cold bent glass* (Doctoral dissertation). University of Cambridge. DOI: 10.17863/CAM.15628

Decaudin, P., Julius, D., Wither, J., Boissieux, L., Sheffer, A., & Cani, M.-P. (2006). Virtual garments: A fully geometric approach for clothing design. *Computer Graphics Forum*, *25*(3), 625–634. DOI: 10.1111/j.1467-8659.2006.00982.x

Dubé, T. W. (1990). The structure of polynomial ideals and Gröbner bases. *SIAM Journal on Computing*, *19*(4), 750–773. DOI: 10.1137/0219053

Eigensatz, M., Kilian, M., Schiftner, A., Mitra, N., Pottmann, H., & Pauly, M. (2010). Paneling architectural freeform surfaces. *ACM Trans. Graph.*, *29*(4), 45:1–45:10. DOI: 10.1145/1778765.1778782

Eklind, M. (2015). Emporia shoppping mall [Online; accessed under CC BY-SA 2.0 on August 20, 2020]. URL: https://flic.kr/p/yhQBR9

Eversmann, P., Ihde, A., & Louter, C. (2016). Low cost double curvature-–Exploratory computational modelling, FE-analysis and prototyping of cold-bent glass. *Challenging Glass Conference Proceedings*, *5*, 81–92. DOI: 10.7480/cgc.5.2233

Eversmann, P., Schling, E., Ihde, A., & Louter, C. (2016). Low-cost double curvature: Geometrical and structural potentials of rectangular, cold-bent glass construction. *Proceedings of IASS Annual Symposia.*

Farouki, R. (2008). *Pythagorean-hodograph curves: Algebra and geometry inseparable.* Springer.
DOI: 10.1007/978-3-540-73398-0

Fildhuth, T., & Knippers, J. (2011). Geometrie und Tragverhalten von doppelt gekrümmten Ganzglasschalen aus kalt verformten Glaslaminaten. *Stahlbau*, *80*(S1), 31–44.
DOI: 10.1002/stab.201120005

Frey, W. (2004). Modeling buckled developable surfaces by triangulation. *Computer-Aided Design*, *36*(4), 299–313.
DOI: 10.1016/S0010-4485(03)00105-2

Fulton, L., Modi, V., Duvenaud, D., Levin, D. I. W., & Jacobson, A. (2019). Latent-space dynamics for reduced deformable simulation. *Computer Graphics Forum*, *38*(2), 379–391.
DOI: 10.1111/cgf.13645

Gil-Ureta, F., Pietroni, N., & Zorin, D. (2019). *Structurally optimized shells.* arXiv: 1904.12240.

Gingold, Y., Secord, A., Han, J., Grinspun, E., & Zorin, D. (2004). A discrete model for inelastic deformation of thin shells.

Glymph, J., Shelden, D., Ceccato, C., Mussel, J., & Schober, H. (2004). A parametric strategy for free-form glass structures using quadrilateral planar facets. *Automation in Construction*, *13*(2), 187–202.
DOI: 10.1016/j.autcon.2003.09.008

Grinspun, E., Gingold, Y., Reisman, J., & Zorin, D. (2006). Computing discrete shape operators on general meshes. *Computer Graphics Forum*, *25*(3), 547–556.
DOI: 10.1111/j.1467-8659.2006.00974.x

Guseinov, R., Miguel, E., & Bickel, B. (2017). CurveUps: Shaping objects from flat plates with tension-actuated curvature. *ACM Trans. Graph.*, *36*(4), 64:1–64:12.
DOI: 10.1145/3072959.3073709

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Julius, D., Kraevoy, V., & Sheffer, A. (2005). D-charts: Quasi-developable mesh segmentation. *Computer Graphics Forum*, *24*(3), 581–590.
DOI: 10.1111/j.1467-8659.2005.00883.x

Jung, A., Hahmann, S., Rohmer, D., Begault, A., Boissieux, L., & Cani, M.-P. (2015). Sketching folds: Developable surfaces from non-planar silhouettes. *ACM Trans. Graph.*, *34*(5), 155:1–155:12.
DOI: 10.1145/2749458

Kingma, D. P., & Ba, J. L. (2014). *Adam: A method for stochastic optimization.* arXiv: 1607.06450.

Koiter, W. T. (1966). On the nonlinear theory of thin elastic shells. *Proc. Koninkl. Ned. Akad. van Wetenschappen, Series B, 69*, 1–54.

Konaković-Luković, M., Panetta, J., Crane, K., & Pauly, M. (2018). Rapid deployment of curved surfaces via programmable auxetics. *ACM Trans. Graph.*, *37*(4), 106:1–106:13.
DOI: 10.1145/3197517.3201373

Lang, J., & Röschel, O. (1992). Developable $(1, n)$-Bézier surfaces. *Computer Aided Geometric Design*, *9*(4), 291–298.
DOI: 10.1016/0167-8396(92)90036-O

Liu, Y., Pottmann, H., Wallner, J., Yang, Y.-L., & Wang, W. (2006). Geometric modeling with conical meshes and developable surfaces. *ACM Trans. Graph.*, *25*(3), 681–689.
DOI: 10.1145/1141911.1141941

Liu, Y., Pottmann, H., & Wang, W. (2006). Constrained 3D shape reconstruction using a combination of surface fitting and registration. *Computer-Aided Design*, *38*(6), 572–583.
DOI: 10.1016/j.cad.2006.01.014

Liu, Y., Xu, W., Wang, J., Zhu, L., Guo, B., Chen, F., & Wang, G. (2011). General planar quadrilateral mesh design using conjugate direction field. *ACM Trans. Graph.*, *30*(6), 1–10.
DOI: 10.1145/2070781.2024174

Luo, R., Shao, T., Wang, H., Xu, W., Chen, X., Zhou, K., & Yang, Y. (2020). NNWarp: Neural network-based nonlinear deformation. *IEEE Transactions on Visualization and Computer Graphics*, *26*(4), 1745–1759.

Malomo, L., Pérez, J., Iarussi, E., Pietroni, N., Miguel, E., Cignoni, P., & Bickel, B. (2018). FlexMaps: Computational design of flat flexible shells for shaping 3D objects. *ACM Trans. Graph.*, *37*(6), 241:1–241:14.
DOI: 10.1145/3272127.3275076

Megaro, V., Thomaszewski, B., Nitti, M., Hilliges, O., Gross, M., & Coros, S. (2015). Interactive design of 3D-printable robotic creatures. *ACM Trans. Graph.*, *34*(6), 216:1–216:9.
DOI: 10.1145/2816795.2818137

Mesnil, R., Douthe, C., Baverel, O., & Leger, B. (2017). Marionette meshes: Modelling free-form architecture with planar facets. *International Journal of Space Structures*, *32*(3-4), 184–198.
DOI: 10.1177/0266351117738379

Mitani, J., & Suzuki, H. (2004). Making papercraft toys from meshes using strip approximate unfolding. *ACM Trans. Graph.*, *23*(3), 259–263.
DOI: 10.1145/1015706.1015711

Mitra, N. J., Kokkinos, I., Guerrero, P., Thuerey, N., Kim, V., & Guibas, L. (2019). CreativeAI: Deep learning for graphics. *ACM SIGGRAPH 2019 Courses*.
DOI: 10.1145/3305366.3328059

Muja, M., & Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*, 2227–2240.

Musialski, P., Auzinger, T., Birsak, M., Wimmer, M., & Kobbelt, L. (2015). Reduced-order shape optimization using offset surfaces. *ACM Trans. Graph.*, *34*(4), 102:1–102:9.
DOI: 10.1145/2766955

Narain, R., Pfaff, T., & O'Brien, J. F. (2013). Folding and crumpling adaptive sheets. *ACM Trans. Graph.*, *32*(4), 51:1–51:8.
DOI: 10.1145/2461912.2462010

Nocedal, J., & Wright, S. (2006). *Numerical optimization*. Springer.
DOI: 10.1007/978-0-387-40065-5

Panetta, J., Konaković-Luković, M., Isvoranu, F., Bouleau, E., & Pauly, M. (2019). X-Shells: A new class of deployable beam structures. *ACM Trans. Graph.*, *38*(4), 83:1–83:15.
DOI: 10.1145/3306346.3323040

Pellis, D., Kilian, M., Dellinger, F., Wallner, J., & Pottmann, H. (2019). Visual smoothness of polyhedral surfaces. *ACM Trans. Graph.*, *38*(4), 31:1–31:11.
DOI: 10.1145/3306346.3322975

Pentland, A., & Williams, J. (1989). Good vibrations: Modal dynamics for graphics and animation. *SIGGRAPH Comput. Graph.*, *23*(3), 207–214.
DOI: 10.1145/74334.74355

Pérez, F., & Suárez, J. A. (2007). Quasi-developable B-spline surfaces in ship hull design. *Computer-Aided Design*, *39*(10), 853–862.
DOI: 10.1016/j.cad.2007.04.004

Pfaff, T., Narain, R., de Joya, J. M., & O'Brien, J. F. (2014). Adaptive tearing and cracking of thin sheets. *ACM Trans. Graph.*, *33*(4), 110:1–110:9.
DOI: 10.1145/2601097.2601132

Piegl, L., & Tiller, W. (1997). *The NURBS Book* (2nd). Springer.
DOI: 10.1007/978-3-642-59223-2

Pottmann, H., Eigensatz, M., Vaxman, A., & Wallner, J. (2015). Architectural geometry. *Comput. Graph.*, *47*(100), 145–164.
DOI: 10.1016/j.cag.2014.11.002

Pottmann, H., Leopoldseder, S., & Hofer, M. (2004). Registration without ICP. *Computer Vision and Image Understanding*, *95*(1), 54–71.
DOI: 10.1016/j.cviu.2004.04.002

Pottmann, H., & Randrup, T. (1998). Rotational and helical surface approximation for reverse engineering. *Computing*, *60*(4), 307–322.
DOI: 10.1007/BF02684378

Pottmann, H., Schiftner, A., Bo, P., Schmiedhofer, H., Wang, W., Baldassini, N., & Wallner, J. (2008). Freeform surfaces from single curved panels. *ACM Trans. Graph.*, *27*(3), 76:1–76:10.
DOI: 10.1145/1360612.1360675

Pottmann, H., & Wallner, J. (2001). *Computational line geometry*. Springer.
DOI: 10.1007/978-3-642-04018-4

Rabinovich, M., Hoffmann, T., & Sorkine-Hornung, O. (2018a). Discrete geodesic nets for modeling developable surfaces. *ACM Trans. Graph.*, *37*(2), 16:1–16:17.
DOI: 10.1145/3180494

Rabinovich, M., Hoffmann, T., & Sorkine-Hornung, O. (2018b). The shape space of discrete orthogonal geodesic nets. *ACM Trans. Graph.*, *37*(6), 228:1–228:17.

Rose, K., Sheffer, A., Wither, J., Cani, M.-P., & Thibert, B. (2007). Developable surfaces from arbitrary sketched boundaries. *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, 163–172.

Schiftner, A., Eigensatz, M., Kilian, M., & Chinzi, G. (2013). Large scale double curved glass facades made feasible – the Arena Corinthians west facade. *Glass Performance Days Finland (Conference Proceedings)* (pp. 494–498).

Schneider, M., & Mehrtens, P. (2013). Cladding freeform surfaces with curved metal panels–a complete digital production chain. *Advances in Architectural Geometry 2012* (pp. 237–242). Springer.
DOI: 10.1007/978-3-7091-1251-9_19

Schulz, A., Xu, J., Zhu, B., Zheng, C., Grinspun, E., & Matusik, W. (2017). Interactive design space exploration and optimization for CAD models. *ACM Trans. Graph.*, *36*(4), 157:1–157:14.
DOI: 10.1145/3072959.3073688

Schumacher, C., Thomaszewski, B., & Gross, M. (2016). Stenciling: Designing structurally-sound surfaces with decorative patterns. *Computer Graphics Forum*, *35*(5), 101–110.
DOI: 10.1111/cgf.12967

Schumacher, C., Zehnder, J., & Bächer, M. (2018). Set-in-stone: Worst-case optimization of structures weak in tension. *ACM Trans. Graph.*, *37*(6), 252:1–252:13. DOI: 10.1145/3272127.3275085

Shelden, D. (2002). *Digital surface representation and the constructibility of Gehry's architecture* (Doctoral dissertation). M.I.T.

Solomon, J., Vouga, E., Wardetzky, M., & Grinspun, E. (2012). Flexible developable surfaces [Proc. Symposium Geometry Processing]. *Computer Graphics Forum*, *31*(5), 1567–1576. DOI: 10.1111/j.1467-8659.2012.03162.x

Stava, O., Vanek, J., Benes, B., Carr, N., & Měch, R. (2012). Stress relief: Improving structural strength of 3D printable objects. *ACM Trans. Graph.*, *31*(4), 48:1–48:11. DOI: 10.1145/2185520.2185544

Stein, O., Grinspun, E., & Crane, K. (2018). Developability of triangle meshes. *ACM Trans. Graph.*, *37*(4), 77:1–77:14. DOI: 10.1145/3197517.3201303

Tang, C., Bo, P., Wallner, J., & Pottmann, H. (2016). Interactive design of developable surfaces. *ACM Trans. Graph.*, *35*(2), 12:1–12:12. DOI: 10.1145/2832906

Toledo, S. (2003). TAUCS, a library of sparse linear solvers. URL: http://www.tau.ac.il/~stoledo/taucs

Ulu, E., Mccann, J., & Kara, L. B. (2017). Lightweight structure design under force location uncertainty. *ACM Trans. Graph.*, *36*(4), 158:1–158:13. DOI: 10.1145/3072959.3073626

Umetani, N., & Bickel, B. (2018). Learning three-dimensional flow for interactive aerodynamic design. *ACM Trans. Graph.*, *37*(4), 89:1–89:10. DOI: 10.1145/3197517.3201325

Umetani, N., Igarashi, T., & Mitra, N. J. (2012). Guided exploration of physically valid shapes for furniture design. *ACM Trans. Graph.*, *31*(4), 86:1–86:11. DOI: 10.1145/2185520.2185582

Vavpetič, A., & Žagar, E. (2019). A general framework for the optimal approximation of circular arcs by parametric polynomial curves. *Journal of Computational and Applied Mathematics*, *345*, 146–158. DOI: 10.1016/j.cam.2018.06.020

Verschoor, M., Casas, D., & Otaduy, M. A. (2020). Tactile rendering based on skin stress optimization. *ACM Trans. Graph.*, *39*(4), 90:1–90:13. DOI: 10.1145/3386569.3392398

Wächter, A., & Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, *106*(1), 25–57.

Wang, C., & Tang, K. (2004). Achieving developability of a polygonal surface by minimum deformation: A study of global and local optimization approaches. *The Visual Computer*, *20*(8), 521–539.
DOI: 10.1007/s00371-004-0256-0

Wang, C. C. L., Wang, Y., & Yuen, M. M.-F. (2004). On increasing the developability of a trimmed NURBS surface. *Engineering with Computers*, *20*(1), 54–64.
DOI: 10.1007/s00366-004-0272-8

Wang, T. Y., Shao, T., Fu, K., & Mitra, N. J. (2019). Learning an intrinsic garment space for interactive authoring of garment animation. *ACM Trans. Graph.*, *38*(6), 220:1–220:12.
DOI: 10.1145/3355089.3356512

Wang, W., Pottmann, H., & Liu, Y. (2006). Fitting B-spline curves to point clouds by curvature-based squared distance minimization. *ACM Trans. Graph.*, *25*(2), 214–238.
DOI: 10.1145/1138450.1138453

Weischedel, C. (2012). A discrete geometric view on shear-deformable shell models.

Wolff, K., & Sorkine-Hornung, O. (2019). Wallpaper pattern alignment along garment seams. *ACM Trans. Graph.*, *38*(4), 62:1–62:12.
DOI: 10.1145/3306346.3322991

Yamauchi, H., Gumhold, S., Zayer, R., & Seidel, H.-P. (2005). Mesh segmentation driven by Gaussian curvature. *The Visual Computer*, *21*(8), 659–668.
DOI: 10.1007/s00371-005-0319-x

Yong, J.-H., & Cheng, F. ((2004). Geometric Hermite curves with minimum strain energy. *Computer Aided Geometric Design*, *21*(3), 281–301.
DOI: 10.1016/j.cagd.2003.08.003

Zhao, H., Xu, W., Zhou, K., Yang, Y., Jin, X., & Wu, H. (2017). Stress-constrained thickness optimization for shell object fabrication. *Computer Graphics Forum*, *36*(6), 368–380.
DOI: 10.1111/cgf.12986