Stability-Aware Simplification of Curve Networks

WILLIAM NEVEU, Université de Montréal, Canada
IVAN PUHACHOV, Université de Montréal, Canada
BERNHARD THOMASZEWSKI, ETH Zürich, Switzerland
MIKHAIL BESSMELTSEV, Université de Montréal, Canada

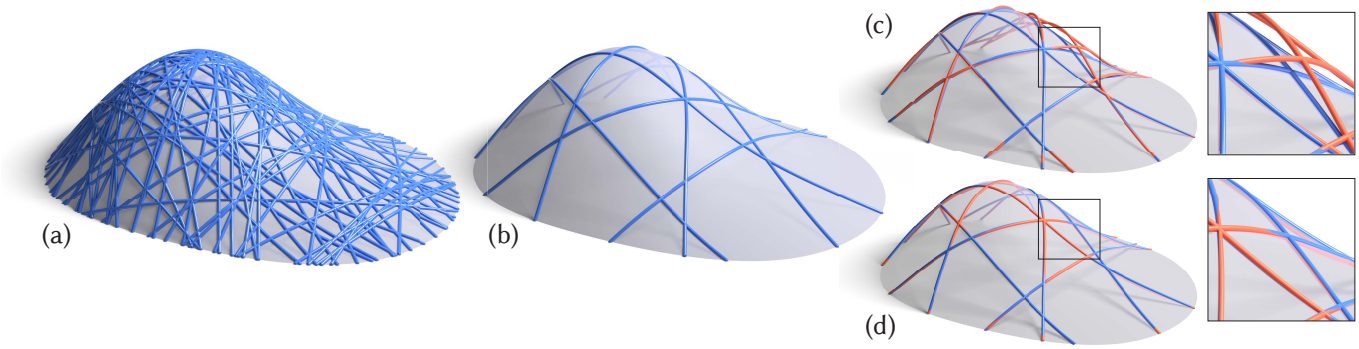


Fig. 1. We automatically simplify a user-provided curve network on a surface (a) to fit a given material budget with nearly optimal stability (b). Since the directions of applied loads are hard to predict, we optimize for *worst-case* stability and minimize the maximum deformation induced by a worst-case external force of unit norm in total (c), implying smaller deformations for all other unit-norm forces, in particular, for the standard gravitational load (d).

Designing curve networks for fabrication requires simultaneous consideration of structural stability, cost effectiveness, and visual appeal—complex, interrelated objectives that make manual design a difficult and tedious task. We present a novel method for fabrication-aware simplification of curve networks, algorithmically selecting a stable subset of given 3D curves. While traditionally stability is measured as magnitude of deformation induced by a set of pre-defined loads, predicting applied forces for common day objects can be challenging. Instead, we directly optimize for minimal deformation under the worst-case load.

Our technical contribution is a novel formulation of 3D curve network simplification for worst-case stability, leading to a mixed-integer semi-definite programming problem (MI-SDP). We show that while solving MI-SDP directly is infeasible, a physical insight suggests an efficient greedy approximation algorithm. We demonstrate the potential of our approach on a variety of curve network designs and validate its effectiveness compared to simpler alternatives using numerical experiments.

CCS Concepts: • **Computing methodologies** → **Parametric curve and surface models**; **Physical simulation**.

Additional Key Words and Phrases: curve networks, stability, fabrication-aware design

SIGGRAPH '22 Conference Proceedings, August 7–11, 2022, Vancouver, BC, Canada

ACM ISBN 978-1-4503-9337-9/22/08…$15.00
https://doi.org/10.1145/3528233.3530711

ACM Reference Format:
William Neveu, Ivan Puhachov, Bernhard Thomaszewski, and Mikhail Bessmeltsev. 2022. Stability-Aware Simplification of Curve Networks. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings (SIGGRAPH '22 Conference Proceedings), August 7–11, 2022, Vancouver, BC, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3528233.3530711

1 INTRODUCTION

From birds' nests to modern stadiums, from weaved baskets to environmental sculpture, from spiderwebs to rooftops, curve networks are found commonly in nature, used in art and crafts, architecture and engineering (Fig. 2). Made out of wicker, wire, bent wooden or metallic beams, these curves are an aesthetically pleasing and functional means to convey complex 3D surfaces. However, designing such curve networks for fabrication is challenging. Artists have to manage requirements of very different, often contradicting, nature: create appealing designs, use little material, and make sure the structure is stable. This often forces artists to follow a cumbersome process where they iteratively adjust their design and test its stability via simulation or prototyping.

In traditional drawing, artists often ideate via a freeform rough sketch, which they then clean up, taking engineering requirements into account [Eissen and Steur 2007]. Similarly, one natural workflow to create a 3D curve network is to first draft a complex curve network on the target surface, guided only by aesthetics, then simplify it, taking into account engineering and budget constraints.

Inspired by this metaphor, we present a new method for fabrication-aware simplification of a given curve network. More specifically, our method selects a subset of given curves on a 3D surface to maximize the curve network's *worst-case* stability. The initial set of curves may be created by sketching interfaces [Arora and Singh 2020],

(a) (b) (c)

Fig. 2. Examples of curve networks in architecture, visual art, and design. (a) The roof of the National Maritime Museum in Amsterdam, (b) A sculpture by Patrick Dougherty at Smithsonian American Art Museum, (c) Beijing National Stadium. Images by Bristollcarus, Eb0178a (cropped), Peter23 respectively, licensed under CC BY-SA.

traditional modeling software [Autodesk 2022], or various curve tracing algorithms. We demonstrate our method produces complex yet stable simplified curve networks (Fig. 1).

Traditionally, stability of a structure is expressed as the amount of deformation induced by a particular set of loads. Yet, loads for common-day objects, small-scale architecture, or sculpture are hard to predict. Therefore, we focus on an alternative definition of stability, measuring the deformation of the structure under the *worst-case* load, i.e., an external force vector of fixed magnitude but *a priori* unknown direction. Thus, our problem is selecting a subset of the given curves, within a fixed material budget, maximizing the structure's worst-case stability.

Our technical innovation is a novel formulation of this problem, where the notion of worst-case stability, related to eigenanalysis of the network's stiffness matrix, leads us to a mixed-integer semidefinite programming (MI-SDP) instance. MI-SDP problems are known to be NP-hard and we show that optimization via modern solvers is infeasible. Instead, we leverage intuition from the physical world that suggests an elegant and simple greedy approximation approach, which we demonstrate to be efficient and accurate.

We validate our method numerically by analyzing its approximation properties, comparing it to a random subset selection scheme, and demonstrating its applications on a gallery of curve networks.

2 RELATED WORK

Our work is inspired by the progress in two areas: computational design of curve networks and truss topology optimization. We focus only on the most relevant works.

Fabrication-Aware Surface Design. The graphics community has seen a significant progress in fabrication-aware design of surfaces, including self-supporting surfaces [de Goes et al. 2013; Liu et al. 2013; Panozzo et al. 2013; Vouga et al. 2012], auxetic [Konaković-Luković et al. 2018], zippables [Schüller et al. 2018], and others. These works often focus on either surface approximation only [Schüller et al. 2018] or surface-specific aspects such as paneling and mold reuse [Eigensatz et al. 2010; Fu et al. 2010; Pellis et al. 2021] or face regularity [Vaxman et al. 2017]. In contrast, we focus on simplification of curve networks, whose faces may not be regular or even closed (Fig. 1).

Stability Optimization. Detecting and improving structural weaknesses in objects designed for 3D printing is a problem that has received considerable attention from the graphics community. One line of research aims at identifying regions of high stresses using simulation with user-defined or heuristically determined loads [Lu et al. 2014; Stava et al. 2012]. Instead of depending on pre-defined applied forces, Langlois et al. [2016] use stochastic optimization to estimate worst case loads. Cui et al. [2020] expand on this approach with a linear-time algorithm for probability gradient computation. Rather than relying on purely stochastic forces, Schumacher et al. [2018] account for uncertainties in load locations and directions by parameterizing the space of expected deviations. While they optimize for worst-case loads within a low-dimensional subspace, our eigenvalue optimization method considers all possible load directions simultaneously. The idea of using eigenanalysis to discover structural weaknesses has been explored before, including works by de Gournay et al. [2008], Zhou et al. [2013], or Zehnder et al. [2016]. These approaches use eigenanalysis to detect weak regions, but resort to other means for improving strength. In contrast, our method directly maximizes its worst-case stability measure, i.e., the minimum eigenvalue of the system's stiffness matrix.

Eigenvalue Optimization. Our worst-case stability criterion gives rise to a constrained optimization problem with bounds on the minimum eigenvalue of the stiffness matrix. Eigenvalue optimization problems occur naturally in many applications of engineering design, e.g., when tuning the frequency response of a structure [Torigaki et al. 1994] or optimizing a continuous elastic structure to minimize its worst-case compliance [Cherkaev and Cherkaev 2004]. In the graphics community, eigenvalue optimization problems have been investigated, e.g., in geometry processing as a means of enforcing bounds on deformation [Kovalsky et al. 2014]. Our approach likewise gives rise to a semi-definite programming problem, but whereas those works optimize over a set of continuous parameters, our decision variables are binary. Eigenvalue optimization problems have also been investigated in the context of computational design using both gradient-free [Bharaj et al. 2015] and gradient-based methods [Musialski et al. 2016; Panetta et al. 2017]. Rather than optimizing for target eigenvalues, Chen et al. [2017] aim at preserving the smallest eigenvalue during mesh coarsening for elastodynamics applications. A generalization of this idea to spectrum-preserving coarsening of geometric linear operators was proposed by Liu et al. [2019]. While all of these methods work on continuous variables, we address the discrete problem of selecting an optimal subset from a large set of pre-defined candidate curves.

Computational Design of Curve Networks. A number of works consider the design of curve networks on surfaces. One line of research focuses on biaxial or triaxial weaving to create regular ribbon structures [Akleman et al. 2009; Campen and Kobbelt 2014; Ren et al. 2021; Takezawa et al. 2016; Tao et al. 2016, 2017; Vekhter et al. 2019]. Other physical curve networks that have been explored include wire meshes [Garg et al. 2014], structures made from planar pre-bent rods [Miguel et al. 2016] and circular arcs [Bo et al. 2011], 3D-printed curve networks [Pérez et al. 2017; Perez et al. 2015], and regular gridshells [Lienhard and Knippers 2015; Panetta et al. 2019; Pillwein and Musialski 2021; Schling et al. 2018]. While the above methods focus on highly regular networks and/or fixed topology,

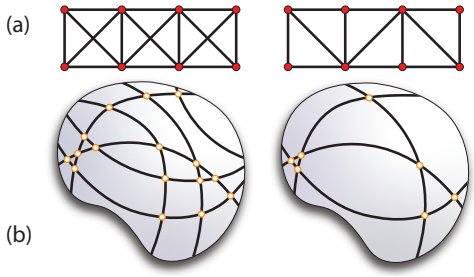


Fig. 3. Typical truss topology optimization approaches (a) optimize a 2D/3D network of straight bars (black) connecting at predefined nodes (red), either optimizing the compliance for a set of loads or satisfying vibration constraints. In contrast, we simplify a curve network on a given surface where the set of nodes, i.e., curve intersections, is not known beforehand (b).

we target curve networks that are not necessarily regular with *a priori* unknown combinatorics.

Following previous work on curve network design, our method relies on discrete elastic rods [Bergou et al. 2010, 2008] and its extension to network connections [Panetta et al. 2019; Perez et al. 2015] as an underlying simulation model.

Truss Topology Optimization For Structural Design. Our optimization method (Sec. 3) is inspired by progress in truss topology optimization (TTO) for structural systems, where a network of straight *bars* connected at a predefined set of *nodes* is optimized to sustain a fixed load or satisfy a vibration constraint (Fig. 3). We outline only the most relevant works, see [Stolpe 2016] for an in-depth overview.

Many TTO approaches, starting from the classical work of Dorn et al. [1964], find thicknesses of the straight bars, assuming a predefined set of nodes where the loads are applied. For instance, [Ben-Tal and Nemirovski 1997] optimize straight bar widths with respect to a set of load cases. Achtziger and Kočvara [2007] introduce TTO formulations using the minimum eigenvalues to analyze the free vibrations of the structures. More recently, Kočvara [2015] consider a variant of TTO, where widths can take only integer values, including 0, leading to an integer linear SDP problem. These works rely on knowing beforehand the set of nodes where the bars connect, necessary to define loads or vibration modes. In our case, however, nodes, i.e. curve intersections, are not fixed and may appear or disappear during optimization, leading to a strictly harder problem.

A small number of works address this problem by including the set of nodes as binary variables in the overall optimization, focusing on constraining feasible topologies [Cerveira et al. 2010] or preventing buckling [Mela 2014].

In a recent work, Arora et al. [2019] find a regular, structurally sound truss configuration given a load, using mesh parameterization and quad meshing methods, where parameterization directions are aligned with principal stresses. Jiang et al. [2019] optimize structure weight and compliance, given a load case, via a combination of geometrical and topological optimization.

Trusses, however, support the load only via *compression* of *straight* rods. In our setup, however, both *compression and bending* of *curvilinear* rods are important load-bearing mechanisms, requiring a different formulation.

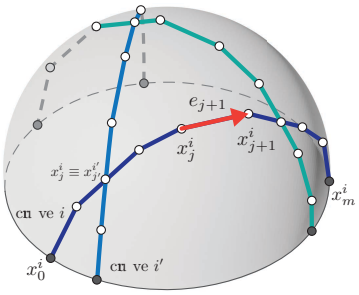


Fig. 4. Curves are represented as polylines on the input mesh. Intersecting curves share their common vertices. Each curve is modeled as a discrete elastic rod.

3 CURVE NETWORK SIMPLIFICATION

At the foundation of our method lies the mathematical model to simulate the deformation of a network of bent rods. We first focus on the physical model for the simulation of a curve network with known geometry and connectivity (Sec. 3.1), then describe our algorithm to simplify a curve network (Sec. 3.2).

3.1 Computational model of a curve network

We model each curve, a polyline initially lying on the target mesh, as a discrete elastic rod [Bergou et al. 2010, 2008]. Curves are connected to each other at the intersection points and their endpoints are fixed at the boundary of the given surface (Fig. 4).

For the simplicity of exposition, we assume each rod is straight when manufactured and then bent during assembly. Initial curvature can be incorporated into the model with no changes to our algorithm. We do not enforce relative curve orientations at intersections since this would lead to large moments at the joints. Furthermore, we neither constrain cross sectional orientations of curves at intersections, nor at their ends. The lowest energy configuration thus will always be twist-free, regardless of how bent curves are. We therefore omit the twisting energy from Bergou et al. [2008].

Therefore, the elastic energy of each curve i with vertex coordinates $x^i_j \in \mathbb{R}^3, j = 0, \ldots, m$ is a sum of stretching and bending terms

$$E^i_{\text{stretch}} = \sum_{j=1}^{m} E^{i,j}_{\text{stretch}} = \sum_{j=1}^{m} k_s \left(\frac{|e_j|}{|\bar{e}_j|} - 1 \right)^2 |\bar{e}_j| \tag{1}$$

$$E^i_{\text{bend}} = \sum_{j=1}^{m-1} E^{i,j}_{\text{bend}} = \sum_{j=1}^{m-1} \frac{k_b}{\ell_j} \left\| \kappa_j \right\|_2^2, \tag{2}$$

with the binormal curvature

$$\kappa_j = \frac{2 e_{j-1} \times e_j}{||\bar{e}_{j-1}|| \cdot ||\bar{e}_j|| + e_{j-1} \cdot e_j},$$

where k_b and k_s are the rods' bending and stretching stiffness coefficients respectively, $\bar{e}_j$ is an edge in the original polyline that becomes $e_j = x^i_j - x^i_{j-1}$ after the deformation, and $\ell_j = 0.5(||\bar{e}_j|| + ||\bar{e}_{j-1}||)$. Here we omit some subscripts i for brevity. Note that the stretching energy is discretized per edge, while bending energy depends on an angle between edges and hence is discretized per vertex.

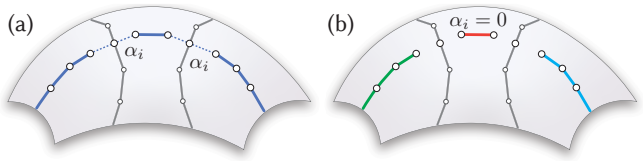


Fig. 5. For each curve, the parameter α_i controls the stiffness of the two edges next to the intersections (dashed lines) with other curves (a). We interpret the extreme case of $\alpha_i = 0$, which effectively splits the curve into disconnected pieces, as removing the curve from the network.

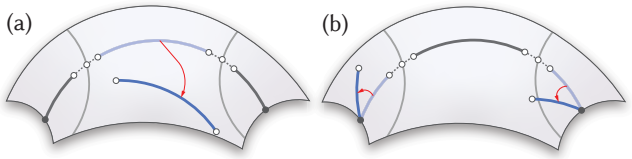


Fig. 6. Once a curve is disconnected from the rest of the network, some segments can be easily translated and rotated (a) and some can be rotated around the fixed endpoint (b).

Each pair of intersecting curves i, i' is coupled via a shared vertex, i.e. $x^i_j \equiv x^{i'}_{j'}$. Physically, this is equivalent to a rotational joint at the intersection allowing curve rotations but not translations. The first and the last vertices of each curve are fixed via a soft penalty constraint with $w = 80$:

$$E^i_{\text{endpoints}} = w(||x^i_0 - \bar{x}^i_0||^2 + ||x^i_m - \bar{x}^i_m||^2). \tag{3}$$

The total energy of the curve network with n curves is then a sum of their the stretching, bending, and endpoint energies:

$$E = \sum_{i=1}^{n} E^i_{\text{stretch}} + E^i_{\text{bend}} + E^i_{\text{endpoints}}. \tag{4}$$

3.2 Simplification Framework

Using this basic model, we now formulate our simplification problem. This stage chooses a subset of the curve network within a given budget, maximizing the worst-case stability of the structure. Here we define material budget as an upper bound on the total length of the curves in the simplified network.

For each curve $i = 1, \ldots, n$, we need to decide whether to keep it or reject it, encoded as $\alpha_i = 1$ or $\alpha_i = 0$, respectively. In the following algorithm, we perform a relaxation $\alpha_i \in [0, 1]$, so we treat α_i as a factor controlling stiffnesses k_s, k_b of two edges around each intersection of that curve with the other curves (Fig. 5a). Thus, α_i can be seen as a factor of cross-section area of those edges for the stretching energy and of the squared area for the bending energy.

To define this formally, let $\mathcal{V}$ be the set of the vertices shared between curve i and the curves intersecting it. Then we can rewrite energies Eq. 1 and 2 as functions of α_i (Fig. 5):

$$E^i_{\text{stretch}}(\alpha_i) = \sum_{j \text{ and } j+1 \notin \mathcal{V}} E^{i,j}_{\text{stretch}} + \alpha_i \sum_{j \text{ or} j+1 \in \mathcal{V}} E^{i,j}_{\text{stretch}} \tag{5}$$

$$E^i_{\text{bend}}(\alpha_i) = \sum_{j \notin \mathcal{V}} E^{i,j}_{\text{bend}} + \alpha_i \sum_{j \in \mathcal{V}} E^{i,j}_{\text{bend}}. \tag{6}$$

Then, grouping the terms with and without α_i and denoting the corresponding partial sums as E^i and E^i_0 respectively, the total energy in Eq. 4 can then be expressed as:

$$E(\alpha_1, \ldots, \alpha_n) = \sum_{i=1}^{n} E^i_{\text{stretch}} + E^i_{\text{bend}} + E^i_{\text{endpoints}} = \sum_{i=1}^{n} E^i_0 + \alpha_i E^i. \tag{7}$$

Structural stability. Our goal is to find a set of $\alpha_i \in \{0, 1\}, i = 1, \ldots, n$, within the given budget, such that we maximize the stability of the structure. We define the structural stability of a set of curves via studying its *worst-case load*, a set of external forces, of unit norm in total, applied to the vertices of the structure that causes the largest displacements x. Such structural stability is known to be equivalent to computing the smallest eigenvalue of the Hessian $\mathbf{H} = \mathbf{H}(\alpha_1, ..., \alpha_n)$ of the elastic energy [Achtziger and Kočvara 2007], a symmetric semi-positive definite matrix:

$$\mathbf{H}x = \lambda x. \tag{8}$$

Naively defining the smallest eigenvalue $\lambda_1(\mathbf{H})$ as the stability of a structure, however, would fail in our setup, since if $\alpha_i = 0$, the Hessian $\mathbf{H}(\alpha_1, ..., \alpha_n)$ may become singular, yielding $\lambda_1(\mathbf{H}) = 0$. Intuitively, if a curve i has at least two other curves intersecting it, as soon as $\alpha_i = 0$, the curve segment between those intersections becomes disconnected from the rest of the structure and can be moved freely (Fig. 6a). Similarly, the segments next to the boundary become free to rotate around fixed endpoint (Fig. 6b). Furthermore, when two intersecting curves i, j have both $\alpha_i = \alpha_j = 0$, their shared vertex becomes disconnected from the overall structure and can be translated in any direction without resistance, again making the Hessian singular.

Instead, following Achtziger and Kočvara [2007], we define the structural stability of our structure for given $\alpha_1, \ldots, \alpha_n$ as the smallest **non-zero** eigenvalue value, which we denote as $\widetilde{\lambda_1}(\mathbf{H})$. We first give a formulation of our problem using $\widetilde{\lambda_1}$, then give an equivalent formulation using the standard minimum eigenvalue λ_1 via Hessian regularization (Sec. 3.3).

We first represent the Hessian H of the total elastic energy as a function of α_i. Elastic energy is a linear function with respect to α_i, so the Hessian is also a linear function of the corresponding Hessians:

$$\mathbf{H}(\alpha_1, ..., \alpha_n) = \sum_{i=1}^{n} \mathbf{H}^i_0 + \sum_{i=1}^{n} \alpha_i \mathbf{H}^i, \tag{9}$$

where $\mathbf{H}^i$ and $\mathbf{H}^i_0$ are Hessians of E^i and E^i_0 in Eq. 7 respectively.

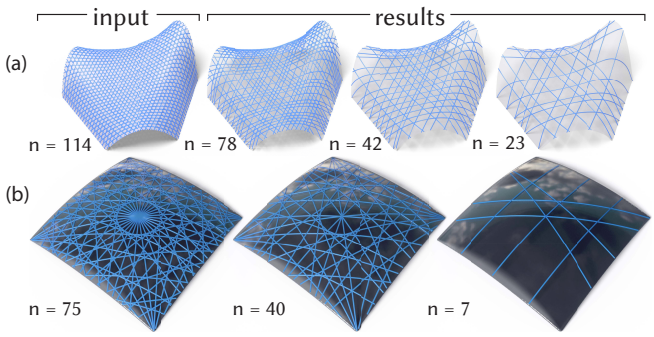


Fig. 7. The user-provided budget constraint can be used to control the final number of curves. Since we store all intermediate results, we allow user to instantaneously revert to any higher number of curves, if desired.

Finally, our curve network simplification can then be formulated as follows:

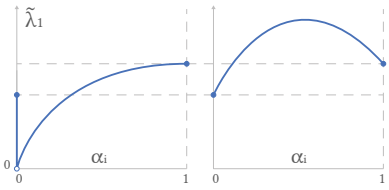
$$\begin{aligned} \max\ & \widetilde{\lambda_1}\left(\sum_{i=1} \mathbf{H}_0^i + \sum_{i=1}^{n} \alpha_i \mathbf{H}^i\right) \\ \text{s.t. } & \alpha_1, ..., \alpha_n \in \{0, 1\} \\ & \sum_{i=1}^{n} \alpha_i L_i \le V \end{aligned} \tag{10}$$

where L_i is the initial length of the i^{th} curve and V is the user-defined material budget.

The curves with $\alpha_i = 0$ are considered to be removed from the system, hence we compute the cost of the curve network as $\sum_{i=1}^{n} \alpha_i L_i$ for the budget constraint.

3.3 Mixed-Integer Semidefinite Programming (MISDP) formulation

Unfortunately, $\widetilde{\lambda_1}(\mathbf{H}(\alpha_1, \ldots, \alpha_n))$ is discontinuous as a function of α_i, and thus cannot be directly used in a gradient-based optimization (inset, left). Intuitively, as α_i is approaching 0, $\widetilde{\lambda_1}$ is also decreasing, until it becomes equal to a different eigenvalue when $\alpha_i = 0$ — hence the discontinuity. To alleviate this problem, we propose to use Hessian regularization. Namely, we can express the minimum non-zero eigenvalue $\widetilde{\lambda_1}$ of the original Hessian as the ordinary minimum eigenvalue λ_1 of the regularized Hessian (inset, right):



$$\widetilde{\lambda_1} = \lambda_1\left(\sum_{i=1}^{n} \mathbf{H}_0^i + \sum_{i=1}^{n} \left[\alpha_i \mathbf{H}^i + (1-\alpha_i)\mathbf{Q}_i\right] + \sum_{i,j} \beta_{i,j}\mathbf{Q}_{i,j}\right),$$

where $\beta_{i,j}$ are auxiliary variables. Here $\mathbf{Q}_i$ and $\mathbf{Q}_{i,j}$ are diagonal Tikhonov regularization matrices. We define $\mathbf{Q}_i$ as 1 on the diagonal for all the vertices of curve i except for shared intersections, and $\mathbf{Q}_{i,j}$ as 1 for all the shared intersection vertices between curves i and j, and 0 otherwise. Note that since the non-zero column-vectors of our regularizer form a basis of the Hessian null space for $\alpha_i = 0$, such regularization preserves the non-zero minimum eigenvalue, i.e., the values of $\widetilde{\lambda_1}$ for $\alpha_i \in \{0, 1\}$ are the same with and without regularization, as illustrated by the two dashed horizontal lines in the inset.

We need to regularize the shared vertices at the intersections of curves i, j only when both curves are removed, adding $\beta_{i,j}Q_{i,j}$, where $\beta_{i,j} = 1$ when $\alpha_i = \alpha_j = 0$. A naive implementation of this requirement on $\beta_{i,j}$ would lead to a quadratic constraint and therefore a nonlinear SDP formulation, notoriously hard to solve efficiently. Instead, we observe that for $\alpha \in \{0, 1\}$ this constraint is equivalent to $\beta_{i,j} \le \min(1 - \alpha_i, 1 - \alpha_j)$, or two linear inequalities.

Finally, finding $\lambda_1(A)$ is equivalent to a maximization problem $\max \lambda$ with a semidefinite constraint $A \succeq \lambda I$ [Vandenberghe and Boyd 1999]. Using the regularization of the Hessian, we come to our final formulation of curve network simplification as an instance of mixed-integer semidefinite programming (MI-SDP):

$$\begin{aligned} \max_{\alpha}\ & \lambda \\ \text{s.t. } & \sum_{i=1} \mathbf{H}_0^i + \sum_{i=1}^{n} \left[\alpha_i \mathbf{H}^i + (1-\alpha_i)\mathbf{Q}_i\right] + \sum_{i,j} \beta_{i,j}\mathbf{Q}_{i,j} \succeq \lambda I \\ & \alpha_i, \beta_{i,j} \in \{0, 1\} \\ & \beta_{i,j} \le 1 - \alpha_i;\ \beta_{i,j} \le 1 - \alpha_j \\ & \sum_{i=1}^{n} \alpha_i L_i \le V \end{aligned} \tag{11}$$

To eliminate the trivial solution $\alpha_1 = \ldots = \alpha_n = 0$, when there are no curves left and the minimum eigenvalue λ is driven by the regularization, we add a constraint $\sum_{i=1}^{n} \alpha_i \ge 1$.

4 SOLVER MECHANISM

A natural approach to solving formulation in Eq. 11 is via its relaxation into an instance of semidefinite programming (SDP) by replacing the integer constraints $\alpha_i \in \{0, 1\}$ with a continuous constraint $0 \le \alpha_i \le 1$. This relaxation gives an upper bound to the mixed-integer problem. This can be easily seen from the previous inset, right: the function is concave and has maximum inside the interval, not on the endpoints that would correspond to the integer solution $\alpha_i = 0$ or $\alpha_i = 1$. The relaxed solution α_{SDP} can be either used directly as an approximation via rounding each element to the nearest integer, or as a conservative upper bound for mixed-integer optimization algorithms such as branch and bound [Gally et al. 2018].

We tried a standard MI-SDP solver YALMIP [Löfberg 2004] and implemented a custom branch and bound method using MOSEK [ApS 2020] as the SDP solver. Our experiments showed that SDP relaxation significantly reduces the space of branch and bound as compared to an exhaustive search (cutting out 50% − 70% of the search space). Unfortunately, for small to medium number of curves the overhead of solving SDP makes branch and bound significantly slower than the exhaustive search. For instance, for 9 initial curves,

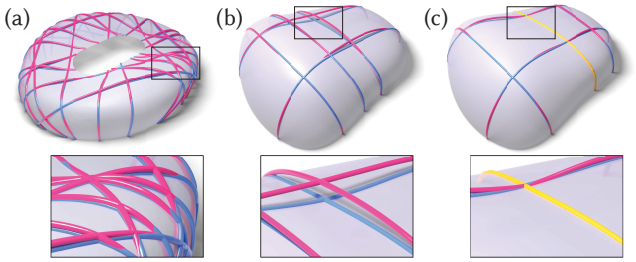


Fig. 8. Many of our initial networks are nearly at their equilibrium state (e.g. (a), blue: initial curves, pink: the curves at the equilibrium). The equilibrium configuration for a simplified curve network might deviate from the target surface (b). For this example, however, adding a single, manually selected, rigid pre-bent curve (c, yellow) is enough to alleviate the deviation

.

branch and bound takes several hours, while the exhaustive search is nearly instant. While theoretically for larger number exhaustive search should become more expensive, for those numbers of curves both approaches are infeasible.

Instead, we propose an efficient and straightforward approximation algorithm. We observe that intuitively, removing any curve from a set can only decrease the stability of the structure. This suggests a process where, starting with the original curve network, we iteratively remove the curve that is the least important for the overall stability, until the budget constraint is satisfied. At each iteration, we compute the equilibrium configuration via Newton method. We denote the initial set of curves S, and $\widetilde{\lambda_1}(A)$ as the minimum non-zero eigenvalue of the Hessian for the structure made of curves in $A \subseteq S$. Using Eq. 9, we set $\widetilde{\lambda_1}(A) = \widetilde{\lambda_1}(\mathbf{H}(\alpha_1, \ldots, \alpha_n))$, where $\alpha_i = 1$ for all $i \in A$ and 0 otherwise. Finally, denoting $Vol(A) = \sum_{i \in A} L_i$, we get Algorithm 1. Here the maximization is performed by simply trying to remove each curve from the current set.

```
Set A = S;
while Vol(A) > V do
    Find c' = arg max_{c∈A} λ̃1(A \ {c});
    Set A = A \ {c'};
end
```

Algorithm 1: Greedy Approximation Algorithm

Note that while formally $\widetilde{\lambda_1}(A \setminus \{c\})$ can be larger than $\widetilde{\lambda_1}(A)$, e.g. when a rare especially weak curve, for instance long with high curvature, exhibits localized buckling, this does not change or impede the algorithm; no special treatment is needed for such cases.

As we discuss in the following section, we find that this simple algorithm approximates the optimal solution with a high ratio, is efficient, parallelizable per iteration, and outperforms naive approaches.

5 RESULTS, VALIDATION, AND DISCUSSION

Using our method, we have simplified a number of curve networks, depicted in 1, 7, 10 and 11. The input surfaces include positive

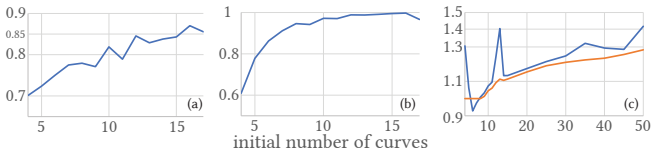


Fig. 9. (a) The approximation ratio of our algorithm, $\lambda_{\text{ours}}/\lambda_{\text{optimum}} > 80\%$ for a sufficient number of curves. (b) Percentile of our solutions among all combinations; typically within the 90-99% percentile. (c) The ratio $\lambda_{\text{ours}}/\lambda_{\text{random search}}$; average in blue, median in orange. Our solutions are more stable than random, sometimes by 40%.

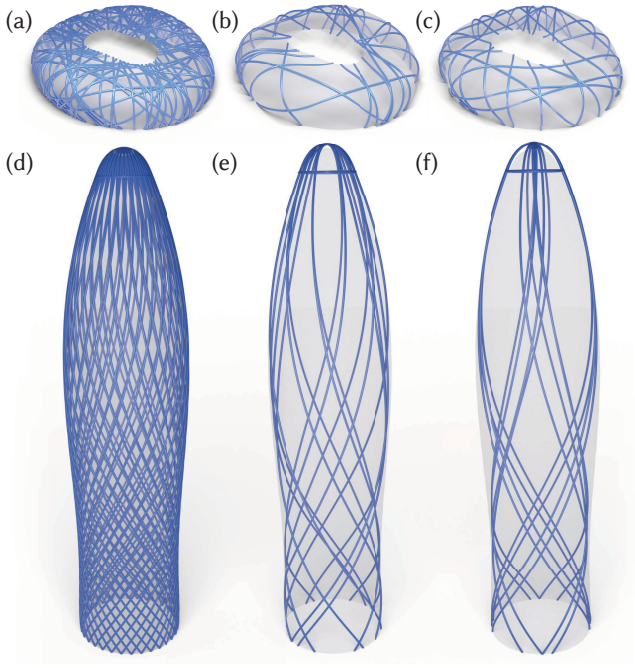


Fig. 10. Compared to the results of random search algorithm (b,e), our results (c,f) are significantly more stable (186% for (c), 116% for (f)). For a highly irregular input curve network (a), worst-case stability may not be an intuitive notion (cf. b, c). For a regular curve network (d), stability may be related to irregularities and thus is easier to see (cf. e, f).

(e.g. tower in Fig. 10d) and negative Gaussian curvature (e.g. tent, Fig. 11b), surfaces with one or two boundaries (stadium in Fig. 11d).

Initial Curve Networks. The curves on a surface can be modeled via conventional [Autodesk 2022] or modern interfaces [Arora and Singh 2020]. To demonstrate our method, we automatically generate random geodesic curves for the bunny (Fig. 11b), the hill (Fig. 1), the stadium (Fig. 10a), and the tent examples (Fig. 11c), or trace two- and three-direction frame fields [Panozzo et al. 2014] for the arched and regular shells (Fig. 11a,d respectively) and the kagome pattern example respectively (Fig. 7a). We modeled the initial curve network on the tower example (Fig. 10d) in a modeling software.

To generate a frame field, we use the representation and the optimization of PolyVector fields [Diamanti et al. 2014]. We use PolyVector polynomials of degree 4 or 6, to capture 2 or 3 directions

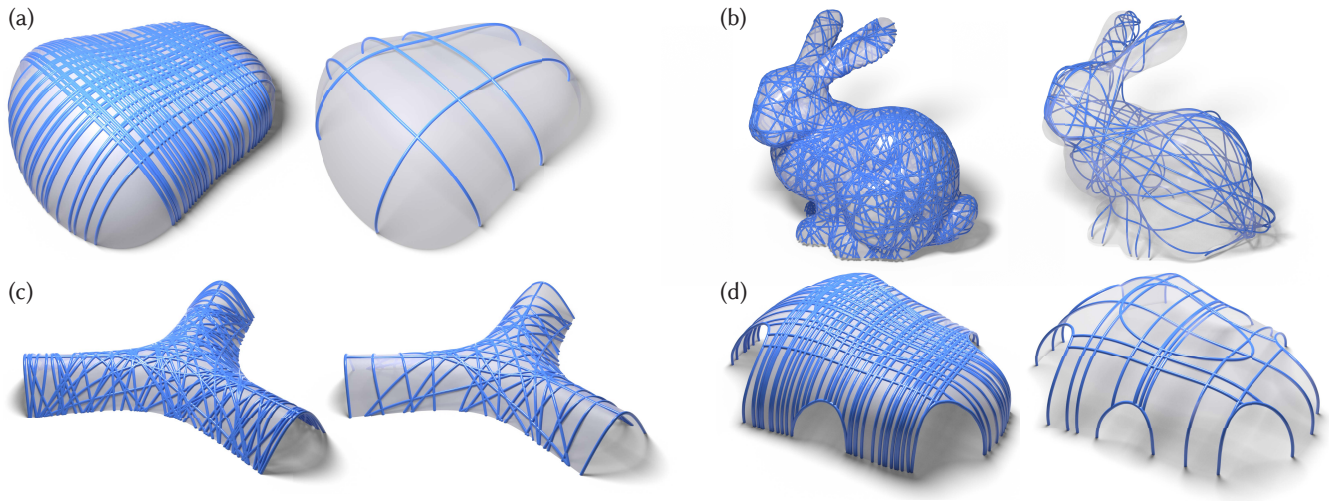


Fig. 11. A gallery of additional results.

Table 1. Algorithm statistics for different curve networks.

Surface	Fig.	initial # of curves	initial # of vertices	final # of curves	time (min.)
hill	1	73	6319	9	17.7
kagome	7a	114	10224	23	53.9
roof	7b	75	2813	7	4.62
stadium	10a	100	6646	25	16.9
tower	10d	33	5699	9	2.46
shell	11a	63	5504	5	7.06
bunny	11b	75	25442	7	266
tent	11c	112	6107	42	24.7
arched shell	11d	67	3514	21	2.60

per triangle respectively. We then sample the surface boundaries and trace the curves using the classical Euler integration method [Polthier and Schmies 2006]. To trace the frame fields, we use principal matching, removing any traced curve that hits a frame field singularity. We follow the standard definition of a singularity as either a zero of one of the directions of a field, or an inconsistency of matching directions around a given vertex.

Equilibrium Configurations. We explicitly compute the equilibrium configuration for all the curve networks via Newton's method. Most of the initial configurations are close to the equilibrium (Fig. 8 and Supplementary). The final curve networks at their equilibrium might deviate from the target surface (e.g., Fig. 8b). If undesired, this effect can be reduced for fabrication by adding pre-bent stiff curves to the initial curve network (yellow curve in Fig. 8c).

Quantitative Evaluation. We numerically compare our algorithm against two baseline algorithms: exhaustive and random searches.

The exhaustive search simply tries all subsets of the given curves and selects the most stable combination within a budget. Clearly, the algorithm has exponential complexity, yet produces the true optimum. We plotted the approximation ratio of our algorithm compared to this true optimum, i.e. the ratio of λ_{ours}, the stability of our result, and λ_{optimum} in Fig. 9a. We tested 200 random curves on each of 4 simple test surfaces and averaged the ratios. We were able to run the exhaustive search in a reasonable time for 17 curves maximum. As seen from the plot, our approximation ratio is increasing with the number of curves and reaches roughly 85%. For comparison, for 19 curves, our algorithm takes less than 1 second, while exhaustive search takes around 45 minutes.

We furthermore display the percentile of our solution among all the combinations (Fig. 9b). Our solution is always in the top $90-99\%$ percentile for a sufficient number of curves.

Finally, we compare our algorithm to a random search algorithm, which generates uniformly distributed random combinations within the budget and chooses the most stable one using the definition in Eq. 10. For a fair comparison, we run the random search for the same time as our algorithm. The ratio of λ_{ours} divided by the best λ of the random search is presented in Fig. 9c. We run this experiment 400 times, each time generating a fixed number of random geodesics on a given surface, and display average (blue) and median values (orange). As indicated by the plot, our algorithmic solution is roughly $20\% - 40\%$ more stable than the solution obtained by the random search, and this ratio increases as the space of possible combinations grows.

Performance. The performance of our algorithm depends on the number of the initial curves, the material budget, and the vertex sampling density of the polylines. On a desktop computer, our computation time typically varies from 2 minutes for the tower (Fig. 10f) to 1 hour for the kagome pattern example (Fig. 7a). The only exception is the bunny (Fig. 11b) that took 4.5 hours. See Table 1 for the full statistics.

Limitations and Future Work. A natural extension of our work is to optimize the shapes of the selected curves to further improve the network's stability while preserving the artist intent; we only

focus on selecting a subset of curves with fixed geometry. Another exciting direction is to perform a simultaneous simplification and stylization of curve networks, akin to sketch stylization.

6 CONCLUSIONS

We presented a novel formulation and an efficient method for simplification of 3D curve networks focusing on their worst-case stability. Our method is compatible with standard interfaces for creating 3D curves and thus has immediate applications in fabrication-aware modeling. We hope that it will form the core of a future fabrication-aware design system allowing users to interactively create, edit, and simplify curve networks.

ACKNOWLEDGMENTS

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grant No.: RGPIN-2019-05097 ("Creating Virtual Shapes via Intuitive Input"), the Fonds de recherche du Québec - Nature et technologies (FRQNT) under Grant No.: 2020-NC-270087, and the NSERC-FRQNT NOVA Grant No. 314090.

This research was also supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant No. 866480).

REFERENCES

Wolfgang Achtziger and Michal Kočvara. 2007. Structural topology optimization with eigenvalues. *SIAM Journal on Optimization* 18, 4 (2007), 1129–1164. https://doi.org/10.1137/060651446

Ergun Akleman, Jianer Chen, Qing Xing, and Jonathan L. Gross. 2009. Cyclic Plain-Weaving on Polygonal Mesh Surfaces with Graph Rotation Systems. *ACM Trans. Graph.* 28, 3, Article 78 (jul 2009), 8 pages. https://doi.org/10.1145/1531326.1531384

MOSEK ApS. 2020. *The MOSEK Fusion API for C++ manual. Version 8.1.* https://docs.mosek.com/8.1/cxxfusion/index.html

Rahul Arora, Alec Jacobson, Timothy R. Langlois, Yijiang Huang, Caitlin Mueller, Wojciech Matusik, Ariel Shamir, Karan Singh, and David I.W. Levin. 2019. Volumetric michell trusses for parametric design & fabrication. *Proceedings: SCF 2019 - ACM Symposium on Computational Fabrication* (2019). https://doi.org/10.1145/3328939.3328999

Rahul Arora and Karan Singh. 2020. Mid-Air Drawing of Curves on 3D Surfaces in AR/VR. *CoRR* abs/2009.09029 (2020). arXiv:2009.09029 https://arxiv.org/abs/2009.09029

Autodesk. 2022. Maya 2022. https://autodesk.com/maya

A. Ben-Tal and A. Nemirovski. 1997. Robust truss topology design via semidefinite programming. *SIAM Journal on Optimization* 7, 4 (1997), 991–1016. https://doi.org/10.1137/S1052623495291951

Miklós Bergou, Basile Audoly, Etienne Vouga, Max Wardetzky, and Eitan Grinspun. 2010. Discrete Viscous Threads. *ACM Trans. Graph.* 29, 4, Article 116 (jul 2010), 10 pages. https://doi.org/10.1145/1778765.1778853

Miklós Bergou, Max Wardetzky, Stephen Robinson, Basile Audoly, and Eitan Grinspun. 2008. Discrete Elastic Rods. *ACM Transactions on Graphics (SIGGRAPH)* 27, 3 (aug 2008), 63:1–63:12.

Gaurav Bharaj, David I. W. Levin, James Tompkin, Yun Fei, Hanspeter Pfister, Wojciech Matusik, and Changxi Zheng. 2015. Computational Design of Metallophone Contact Sounds. *ACM Trans. Graph.* 34, 6, Article 223 (oct 2015), 13 pages. https://doi.org/10.1145/2816795.2818108

Pengbo Bo, Helmut Pottmann, Martin Kilian, Wenping Wang, and Johannes Wallner. 2011. Circular arc structures. *ACM Transactions on Graphics* 30, 4 (2011), 1–12. https://doi.org/10.1145/1964921.1964996

Marcel Campen and Leif Kobbelt. 2014. Dual Strip Weaving: Interactive Design of Quad Layouts Using Elastica Strips. *ACM Trans. Graph.* 33, 6, Article 183 (nov 2014), 10 pages. https://doi.org/10.1145/2661229.2661236

Adelaide Cerveira, Agostinho Agra, Fernando Bastos, and Joaquim A. S. Gromicho. 2010. New branch and bound approaches for truss topology design with discrete areas.

Desai Chen, David I. W. Levin, Wojciech Matusik, and Danny M. Kaufman. 2017. Dynamics-Aware Numerical Coarsening for Fabrication Design. *ACM Trans. Graph.* 36, 4, Article 84 (jul 2017), 15 pages. https://doi.org/10.1145/3072959.3073669

Elena Cherkaev and Andrej Cherkaev. 2004. *Principal Compliance and Robust Optimal Design.* Springer Netherlands, Dordrecht, 169–196. https://doi.org/10.1007/1-4020-2308-1_14

Qiaodong Cui, Timothy Langlois, Pradeep Sen, and Theodore Kim. 2020. Fast and Robust Stochastic Structural Optimization. *Computer Graphics Forum* 39, 2 (2020), 385–397. https://doi.org/10.1111/cgf.13938 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13938

Fernando de Goes, Pierre Alliez, Houman Owhadi, and Mathieu Desbrun. 2013. On the Equilibrium of Simplicial Masonry Structures. *ACM Trans. Graph.* 32, 4, Article 93 (jul 2013), 10 pages. https://doi.org/10.1145/2461912.2461932

Frederic de Gournay, Grégoire Allaire, and François Jouve. 2008. Shape and topology optimization of the robust compliance via the level set method. *ESAIM: COCV* 14, 1 (2008), 43–70. https://doi.org/10.1051/cocv:2007048

Olga Diamanti, Daniele Panozzo, and Olga Sorkine-hornung. 2014. Designing N-PolyVector Fields with Complex Polynomials. *Eurographics Symposium on Geometry Processing* 33, 5 (2014).

William S. Dorn, Ralph E. Gomory, and Harvey J. Greenberg. 1964. Automatic design of optimal structures.

Michael Eigensatz, Martin Kilian, Alexander Schiftner, Niloy J. Mitra, Helmut Pottmann, and Mark Pauly. 2010. Paneling architectural freeform surfaces. *ACM SIGGRAPH 2010* 1, 212 (2010), 1–10. https://doi.org/10.1145/1778765.1778782

Koos Eissen and Roselien Steur. 2007. *Sketching : drawing techniques for product designers* (paperback ed.). BIS Publishers, Amsterdam, Netherlands.

Chi Wing Fu, Chi Fu Lai, Ying He, and Daniel Cohen-Or. 2010. K-set tilable surfaces. *ACM Transactions on Graphics* 29, 4 (2010), 1–6. https://doi.org/10.1145/1778765.1778781

Tristan Gally, Marc E. Pfetsch, and Stefan Ulbrich. 2018. A framework for solving mixed-integer semidefinite programs. *Optimization Methods and Software* 33, 3 (2018), 594–632. https://doi.org/10.1080/10556788.2017.1322081 arXiv:https://doi.org/10.1080/10556788.2017.1322081

Akash Garg, Andrew O. Sageman-Furnas, Bailin Deng, Yonghao Yue, Eitan Grinspun, Mark Pauly, and Max Wardetzky. 2014. Wire Mesh Design. *ACM Trans. Graph.* 33, 4 (2014), 66:1–66:12. https://doi.org/10.1145/2601097.2601106

Caigui Jiang, Chengcheng Tang, Hans Peter Seidel, Renjie Chen, and Peter Wonka. 2019. Computational design of lightweight trusses. *arXiv* (2019). arXiv:1901.05637

Mina Konaković-Luković, Julian Panetta, Keenan Crane, and Mark Pauly. 2018. Rapid deployment of curved surfaces via programmable auxetics. *ACM Transactions on Graphics* 37, 4 (2018), 1–13. https://doi.org/10.1145/3197517.3201373

Shahar Z. Kovalsky, Noam Aigerman, Ronen Basri, and Yaron Lipman. 2014. Controlling Singular Values with Semidefinite Programming. *ACM Trans. Graph.* 33, 4, Article 68 (jul 2014), 13 pages. https://doi.org/10.1145/2601097.2601142

Michal Kočvara. 2015. Truss topology design with integer variables made easy. January 2010 (2015). http://www.optimization-online.org/DB_FILE/2010/05/2614.pdf

Timothy Langlois, Ariel Shamir, Daniel Dror, Wojciech Matusik, and David I. W. Levin. 2016. Stochastic Structural Analysis for Context-Aware Design and Fabrication. *ACM Trans. Graph.* 35, 6, Article 226 (nov 2016), 13 pages. https://doi.org/10.1145/2980179.2982436

Julian Lienhard and Jan Knippers. 2015. Bending-active structures. *Bautechnik* 92, 6 (2015), 394–402. https://doi.org/10.1002/bate.201500007

Hsueh-Ti Derek Liu, Alec Jacobson, and Maks Ovsjanikov. 2019. Spectral Coarsening of Geometric Operators. *ACM Trans. Graph.* 38, 4, Article 105 (jul 2019), 13 pages. https://doi.org/10.1145/3306346.3322953

Yang Liu, Hao Pan, John Snyder, Wenping Wang, and Baining Guo. 2013. Computing self-supporting surfaces by regular triangulation. *ACM Transactions on Graphics* 32, 4 (2013). https://doi.org/10.1145/2461912.2461927

J. Löfberg. 2004. YALMIP : A Toolbox for Modeling and Optimization in MATLAB. In *In Proceedings of the CACSD Conference.* Taipei, Taiwan.

Lin Lu, Andrei Sharf, Haisen Zhao, Yuan Wei, Qingnan Fan, Xuelin Chen, Yann Savoye, Changhe Tu, Daniel Cohen-Or, and Baoquan Chen. 2014. Build-to-Last: Strength to Weight 3D Printed Objects. *ACM Trans. Graph.* 33, 4, Article 97 (jul 2014), 10 pages. https://doi.org/10.1145/2601097.2601168

Kristo Mela. 2014. Resolving Issues with Member Buckling in Truss Topology Optimization Using a Mixed Variable Approach. *Struct. Multidiscip. Optim.* 50, 6 (dec 2014), 1037–1049. https://doi.org/10.1007/s00158-014-1095-x

Eder Miguel, Mathias Lepoutre, and Bernd Bickel. 2016. Computational Design of Stable Planar-Rod Structures. *ACM Transactions on Graphics (SIGGRAPH 2016)* 35, 4 (2016).

Przemyslaw Musialski, Christian Hafner, Florian Rist, Michael Birsak, Michael Wimmer, and Leif Kobbelt. 2016. Non-Linear Shape Optimization Using Local Subspace Projections. *ACM Transactions on Graphics* 35, 4 (2016), 87:1–87:13. https://doi.org/10.1145/2897824.2925886

Julian Panetta, Mina Konaković-Luković, Florin Isvoranu, Etienne Bouleau, and Mark Pauly. 2019. X-Shells: A new class of deployable beam structures. *ACM Transactions on Graphics* 38, 4 (2019). https://doi.org/10.1145/3306346.3323040

Julian Panetta, Abtin Rahimian, and Denis Zorin. 2017. Worst-Case Stress Relief for Microstructures. *ACM Trans. Graph.* 36, 4, Article 122 (jul 2017), 16 pages.

https://doi.org/10.1145/3072959.3073649

Daniele Panozzo, Philippe Block, and Olga Sorkine-Hornung. 2013. Designing unreinforced masonry models. *ACM Transactions on Graphics* 32, 4 (2013). https://doi.org/10.1145/2461912.2461958

D Panozzo, E Puppo, M Tarini, and O Sorkine-Hornung. 2014. Frame Fields: Anisotropic and Non-Orthogonal Cross Fields. *Acm Transactions on Graphics* 33, 4 (2014), 11. https://doi.org/10.1145/2601097.2601179

Davide Pellis, Martin Kilian, Helmut Pottmann, and Mark Pauly. 2021. Computational design of weingarten surfaces. *ACM Transactions on Graphics* 40, 4 (2021). https://doi.org/10.1145/3450626.3459939

Jesús Pérez, Miguel A. Otaduy, and Bernhard Thomaszewski. 2017. Computational Design and Automated Fabrication of Kirchhoff-Plateau Surfaces. *ACM Trans. Graph.* 36, 4, Article 62 (jul 2017), 12 pages. https://doi.org/10.1145/3072959.3073695

Jesus Perez, Bernhard Thomaszewski, Stelian Coros, Bernd Bickel, José A. Canabal, Robert Sumner, and Miguel A. Otaduy. 2015. Design and Fabrication of Flexible Rod Meshes. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH)* 34, 4 (2015). http://www.gmrv.es/Publications/2015/PTCBCSO15

Stefan Pillwein and Przemyslaw Musialski. 2021. Generalized Deployable Elastic Geodesic Grids. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2021)* 40, 6 (2021), in press. https://doi.org/10.1145/3478513.3480516 arXiv:arXiv:2111.08883v1

Konrad Polthier and Markus Schmies. 2006. Straightest Geodesics on Polyhedral Surfaces. In *ACM SIGGRAPH 2006 Courses* (Boston, Massachusetts) *(SIGGRAPH '06)*. Association for Computing Machinery, New York, NY, USA, 30–38. https://doi.org/10.1145/1185657.1185664

Yingying Ren, Switzerland Julian Panetta, Uc Davis, Usa Tian Chen, Switzerland Florin Isvoranu, Switzerland Samuel Poincloux, Switzerland Christopher Brandt, Switzerland Alison Martin, Independent Researcher, Italy Mark Pauly, Julian Panetta, Tian Chen, Florin Isvoranu, Samuel Poincloux, Christopher Brandt, Alison Martin, and Mark Pauly. 2021. 3D Weaving with Curved Ribbons. *ACM Transactions on Graphics* 40, 4 (2021), 1–15. https://doi.org/10.1145/3450626.3459788

Eike Schling, Wang Hui, Jonas Schikore, and Helmut Pottmann. 2018. Design and Construction of Curved Support Structures with Repetitive Parameters.

Christian Schüller, Roi Poranne, and Olga Sorkine-Hornung. 2018. Shape representation by zippables. *ACM Transactions on Graphics* 37, 4 (2018). https://doi.org/10.1145/3197517.3201347

Christian Schumacher, Jonas Zehnder, and Moritz Bächer. 2018. Set-in-Stone: Worst-Case Optimization of Structures Weak in Tension. *ACM Trans. Graph.* 37, 6, Article 252 (dec 2018), 13 pages. https://doi.org/10.1145/3272127.3275085

Ondrej Stava, Juraj Vanek, Bedrich Benes, Nathan Carr, and Radomír Měch. 2012. Stress Relief: Improving Structural Strength of 3D Printable Objects. *ACM Trans. Graph.* 31, 4, Article 48 (jul 2012), 11 pages. https://doi.org/10.1145/2185520.2185544

Mathias Stolpe. 2016. Truss optimization with discrete design variables: a critical review. *Structural and Multidisciplinary Optimization* 53, 2 (2016), 349–374. https://doi.org/10.1007/s00158-015-1333-x

Masahito Takezawa, Takuma Imai, Kentaro Shida, and Takashi Maekawa. 2016. Fabrication of Freeform Objects by Principal Strips. *ACM Trans. Graph.* 35, 6, Article 225 (nov 2016), 12 pages. https://doi.org/10.1145/2980179.2982406

Ye Tao, Nannan Lu, Caowei Zhang, Guanyun Wang, Cheng Yao, and Fangtian Ying. 2016. CompuWoven: A Computer-Aided Fabrication Approach to Hand-Woven Craft. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI EA '16)*. Association for Computing Machinery, New York, NY, USA, 2328–2333. https://doi.org/10.1145/2851581.2892293

Ye Tao, Guanyun Wang, Caowei Zhang, Nannan Lu, Xiaolian Zhang, Cheng Yao, and Fangtian Ying. 2017. *WeaveMesh: A Low-Fidelity and Low-Cost Prototyping Approach for 3D Models Created by Flexible Assembly*. Association for Computing Machinery, New York, NY, USA, 509–518. https://doi.org/10.1145/3025453.3025699

Toshikazu Torigaki, Ichiro Hagiwara, Yuichi Kitagawa, Maki Ueda, Zheng-Dong Ma, and Noboru Kikuchi. 1994. Development and Application of a Shape-Topology Optimization System Using a Homogenization Method. *SAE Transactions* 103 (1994), 1217–1223. http://www.jstor.org/stable/44611836

Lieven Vandenberghe and Stephen Boyd. 1999. Applications of semidefinite programming. *Applied Numerical Mathematics* 29, 3 (1999), 283–299.

Amir Vaxman, Christian Müller, and Ofir Weber. 2017. Regular meshes from polygonal paterns. *ACM Transactions on Graphics* 36, 4 (2017). https://doi.org/10.1145/3072959.3073593

Josh Vekhter, Jiacheng Zhuo, Luisa F.Gil Fandino, Qixing Huang, and Etienne Vouga. 2019. Weaving geodesic foliations. *ACM Transactions on Graphics* 38, 4 (2019). https://doi.org/10.1145/3306346.3323043

Etienne Vouga, Mathias Höbinger, Johannes Wallner, and Helmut Pottmann. 2012. Design of Self-Supporting Surfaces. *ACM Trans. Graph.* 31, 4, Article 87 (jul 2012), 11 pages. https://doi.org/10.1145/2185520.2185583

Jonas Zehnder, Stelian Coros, and Bernhard Thomaszewski. 2016. Designing structurally-sound ornamental curve networks. *ACM Transactions on Graphics* 35, 4 (2016). https://doi.org/10.1145/2897824.2925888

Qingnan Zhou, Julian Panetta, and Denis Zorin. 2013. Worst-Case Structural Analysis. *ACM Trans. Graph.* 32, 4, Article 137 (jul 2013), 12 pages. https://doi.org/10.1145/2461912.2461967