X-Shells: A new class of deployable beam structures

JULIAN PANETTA, EPFL
MINA KONAKOVIĆ-LUKOVIĆ, EPFL
FLORIN ISVORANU, EPFL
ETIENNE BOULEAU, Ingeni SA
MARK PAULY, EPFL

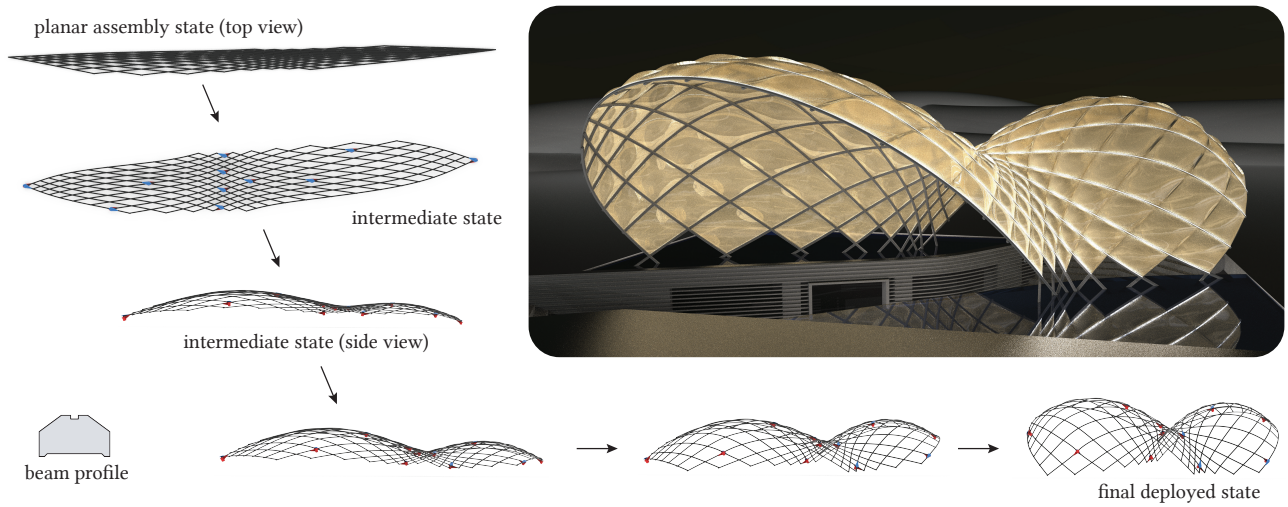


Fig. 1. An *X-shell* is a deformable mechanism that can be assembled from linear beam elements in a flat configuration and deployed to a desired 3D target form. Our algorithm computes the layout and parameters of the flexible beam network as well as a sparse pattern of actuation forces required to deploy the structure. The sequence illustrates the deployment process, where torque actuators at the joints are stylized in red and blue. The top right image shows a design study for a potential architectural application with additional cladding.

We present *X-shells*, a new class of deployable structures formed by an ensemble of elastically deforming beams coupled through rotational joints. An X-shell can be assembled conveniently in a flat configuration from standard elastic beam elements and then deployed through force actuation into the desired 3D target state. During deployment, the coupling imposed by the joints will force the beams to twist and buckle out of plane to maintain a state of static equilibrium. This complex interaction of discrete joints and continuously deforming beams allows interesting 3D forms to emerge. Simulating X-shells is challenging, however, due to unstable equilibria at the onset of beam buckling. We propose an optimization-based simulation framework building on a discrete rod model that robustly handles such difficult scenarios by analyzing and appropriately modifying the elastic energy Hessian. This real-time simulation method forms the basis of a computational design tool for X-shells that enables interactive design space exploration by varying and optimizing design parameters to achieve a specific design intent. We jointly optimize the assembly state and the deployed configuration to ensure the geometric and structural integrity of the deployable X-shell. Once a design is finalized, we also optimize for a sparse distribution of actuation forces to efficiently deploy it from its flat assembly state to its 3D target state. We demonstrate the effectiveness of our design approach with a number of design studies that highlight the richness of the X-shell design space, enabling new forms not possible with existing approaches. We validate our computational model with several physical prototypes that show excellent agreement with the optimized digital models.

CCS Concepts: • **Computing methodologies** → **Shape Modeling**; **Simulation**.

Additional Key Words and Phrases: grid shells, deployable structures, physics-based simulation, numerical optimization, computational design

ACM Reference Format:
Julian Panetta, Mina Konaković-Luković, Florin Isvoranu, Etienne Bouleau, and Mark Pauly. 2019. X-Shells: A new class of deployable beam structures. *ACM Trans. Graph.* 38, 4, Article 83 (July 2019), 15 pages. https://doi.org/10.1145/3306346.3323040

Authors' addresses: J. Panetta, EPFL, julian.panetta@epfl.ch; M. Konaković- Luković, EPFL, mina.konakovic@epfl.ch; F. Isvoranu, EPFL, florin.isvoranu@epfl. ch; E. Bouleau, Ingeni SA, etienne.bouleau@ingeni.ch; M. Pauly, EPFL, mark. pauly@epfl.ch.

1 INTRODUCTION

Modeling and simulation of deformable objects is a core topic in computer graphics. Recent efforts have focused on predictive simulation and design exploration of physical structures for computational design and fabrication [Attene et al. 2018; Bickel et al. 2018].

An interesting class of deformable objects are deployable structures that can transition between two or more distinct geometric configurations [Guseinov et al. 2017; Kilian et al. 2017; Konaković-Luković et al. 2018b; Pérez et al. 2017]. In architectural design, *gridshells* are a particularly intriguing example. Pioneered by V. Shukhov in 1896, gridshells have later been refined by Frei Otto in iconic designs such as the Mannheim Multihalle [Liddell 2015]. Gridshells are assembled on the ground as a regular quadrilateral grid of flexible beams that are connected at rotational joints. The joints allow the quads to shear which in addition to the elastic deformation of the beams enables the structure to be deployed to a curved surface (see Figure 2, top). Gridshells are attractive in architecture because they are lightweight and structurally efficient [Mesnil 2013]. However, relatively few realizations exist today, mainly due to the complexity of deployment: a gridshell assumes its desired shape when the boundary nodes are forced towards pre-defined positions, which necessitates tailor-made erection equipment and significant temporary formwork [Quinn and Gengnagel 2014]. In addition, traditional gridshells have a fairly limited space of realizable geometries and often suffer from stress concentrations that can lead to material failure [Tayeb et al. 2013].

We address these drawbacks and propose a new deployable structure called *X-shell*. Similar to gridshells, an X-shell is formed by a network of interconnected beams that are assembled in a flat configuration. In contrast to gridshells, however, these beams do not form a regular grid, nor are they necessarily straight in the flat assembly state. More importantly, the deployment of X-shells is achieved through intrinsic actuation, i.e. by applying a torque at certain joints to expand the initially flat beam network (Figure 1). The target shape of an X-shell is encoded in the flat rest configuration and does not rely on constraints imposed on the boundary nodes. Therefore, the deployment does not require any formwork or complex support structures. Note that a traditional gridshell actuated in this way would simply shear in the plane and not assume a curved 3D shape.

In our work, we leverage the "incompatibility" of the beam network in terms of a linkage mechanism to force the beams to buckle out of plane and assume the desired curved target shape. As such, X-shells can be considered as a special kind of mechanical linkage where the commonly used rigid elements are replaced by flexible elements that can bend and twist. This unique dynamic behavior poses significant challenges for robust numerical simulation and design optimization that we address in this paper.

Computer graphics research has studied numerous other types of shape-shifting or deployable structures that we discuss more thoroughly in Section 2. What is common to these methods is that the kinematics of deployment as well as considerations about materiality, fabrication, and assembly impose numerous geometric and physical constraints. These constraints typically define intricate design trade-offs that can be extremely difficult to navigate by hand. As a consequence, effective design exploration is often only possible with computational support. This is certainly true in our case, where the coupling of elastic beams in the network defines a complex deformation behavior of the structure that is very hard to predict or control manually.

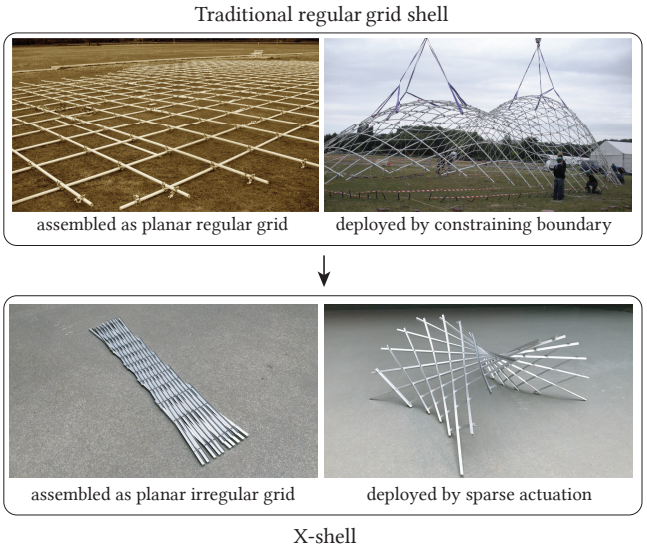


Fig. 2. A traditional gridshell is formed by a regular grid of elastic beams that assumes a curved shape when lifted and fixed along the boundary to a pre-defined curve. In contrast, X-shells have their target shape encoded in the flat non-uniform layout and can be deployed to an equilibrium state via actuation at a small set of joints. *Top images from the Solidays' Festival [Caron et al. 2012] by permission of Cyril Douthe.*

Contributions. To facilitate predictive form-finding and interactive design of X-shells we introduce the following core technical contributions:

- An efficient and numerically robust simulation algorithm to compute the 3D equilibrium states of an X-shell under torque actuations at the joints;
- an optimization approach that adapts the design parameters of an initial flat assembly state so that it deploys into a low-stress state that remains close to a given 3D reference geometry; and
- an algorithm to compute a sparse set of actuated joints for driving the deployment from flat to target shape.

We combine these methods into a computational tool for interactive exploration of X-shell designs. We show that X-shells offer a rich design space that enables the construction of new deployable structures not possible with existing methods. We highlight applications in architecture and demonstrate the validity of our approach through several design studies and physical prototypes.

Overview. The rest of the paper is organized as follows: After discussing related work, we define in Section 3 our mathematical representation that encodes all the relevant geometric and physical parameters of an X-shell. Sections 4 to 6 then present the core technical contributions listed above, i.e. algorithms for forward simulation,

design optimization, and actuation sparsification. In Section 7 we show results created with our approach, before concluding with a discussion of limitations and ideas for future research.

We discuss additional implementation aspects of our numerical optimization methods in the appendix and have released a C++ implementation of our simulation algorithms, along with Jupyter notebooks for simulating and visualizing the deployment for all X-shells shown in the paper, at http://lgg.epfl.ch/XShells. Please also refer to the supplemental material, where we provide detailed derivations of our energy and constraint functions, gradients, and Hessians.

2 RELATED WORK

Computer graphics and related disciplines have recently seen increasing interest in computational design for digital fabrication. For a general overview we refer to recent surveys [Attene et al. 2018; Bermano et al. 2017; Bickel et al. 2018]. The main application domain of our work is architectural design and we refer [Pottmann et al. 2015] for a broad discussion on recent advances in the field of architectural geometry. Here we focus the discussion on prior art that is most closely related to our work.

Curved Surfaces from Flat Materials. An important aspect of X-shells is that they can be assembled in the plane and then deployed to the desired 3D target shape. A series of recent papers explore different ways to form 3D shapes from easy-to-fabricate planar materials. [Schüller et al. 2018] propose a method for the design and fabrication of complex surfaces with a single, ribbon-like piece of flat fabric. When zippered up along its boundary, the fabric approximates a given 3D shape. Malomo and colleagues [2018] design flat, flexible panels from spiraling microstructures. The panels are optimized to be in a static equilibrium when assembled to match a desired 3D surface. [Konaković et al. 2016] leverage conformal geometry to rationalize curved surfaces with auxetic materials, created by cutting inextensible sheet material with a fixed patterns to enable limited uniform stretching. [Garg et al. 2014] utilize Chebyshev nets to create 3D designs from woven metal wires arranged in a regular grid.

Deployment-aware Design. Other works have explored fabrication-aware design that also takes into account the deployment process. [Kilian et al. 2017] propose a method for curved folded surfaces that transition from planar sheets to freeform shapes actuated by a network of strings, making the actuation process an integral part of the structure. Rigid-foldable origami can also be used to design deployable shells at architectural scale [Tachi 2013]. Inspired by the mechanism of the Hoberman Sphere, [Zheng et al. 2016] design deployable rigid scissor linkages that approximate 3D models while ensuring a collision-free expansion path.

Several previous works have studied how to encode a target 3D shape into a flat sheet of material augmented with an actuation logic. [Guseinov et al. 2017] and [Pérez et al. 2017] use prestressing as a driving force. They embed 3D printed elements in a pre-stretched membrane which contracts to a prescribed 3D form when released. [Konaković-Luković et al. 2018a,b] work with varying scale auxetic materials and rely on inflation and gravitational loading to deform a flat sheet towards the desired 3D configuration. [Celli et al. 2018] show how curved shapes can arise through buckling of planar elastic sheets when suitable architected patterns are cut into the material.

In our approach we simulate and optimize a network of elastic beams coupled with rotational joints. This setup is fundamentally different from the above methods, both in the physical composition of the structure and the applied deployment mechanism. Consequently, we need a new representation and optimization approach to accurately model the behavior of X-shells.

Mechanical Linkages. Kinematic linkages have been used to create animated structures whose rigid parts trace out user-specified planar curves [Bächer et al. 2015; Ceylan et al. 2013; Thomaszewski et al. 2014]. Our X-shells' buckling behavior suggests a possible avenue to extend these methods to generate out-of-plane motion by harnessing the buckling of deformable, geometrically "incompatible" parts.

Gridshells. The closest form of deployable structures to our X-shells are elastic gridshells. Here we do not consider *static* gridshells, e.g. [Mesnil et al. 2017; Pietroni et al. 2015; Tang et al. 2014; Tonelli et al. 2016] that are not deployable and thus need to be constructed incrementally in place.

Traditional elastic gridshells are composed of straight elastic beams linked to form a regular quadrilateral grid. They achieve their final shape by active bending [Du Peloux 2017; Lienhard 2014]. The main methods of erection are *pull up* with cranes and cables, *push up* with static framework and jacking towers, and *ease down* with hydraulic and mechanical formwork [Quinn and Gengnagel 2014]. Contrary to our setup, the final shape of a gridshell is largely determined by its boundary, where boundary nodes have to be explicitly fixed to a prescribed location. During erection, a gridshell often has to go through high-energy configurations until it settles in the final shape when the boundary is fixed. To increase safety and reduce the time and cost of manufacturing and erecting an elastic gridshell, [Liuti et al. 2017; Quinn and Gengnagel 2014] study erection with pneumatic formwork.

[Hernández et al. 2012] and [Hernández et al. 2013] present a variational approach to modeling elastic gridshells, where joints are allowed to deviate from the regular grid locations. Their goal is to improve the structural performance of gridshells by reducing the curvature of beams, while our focus is on accurate and robust simulation of the physical behavior of flexible linkages that are directly optimized to minimize elastic energy.

Rod Simulation. Various approaches for elastic rod simulation have been studied in [Bertails et al. 2006; Pai 2002; Spillmann and Teschner 2007; Umetani et al. 2014] based on Cossarat theory and in [Rosenblum et al. 1991; Selle et al. 2008; Iben et al. 2013] using mass-spring models. Our numerical simulation of X-shells is based on the discrete elastic rods model first proposed in [Bergou et al. 2008]. Specifically, we use the efficient framed curve representation from [Bergou et al. 2010], which updates the frame using parallel transport over time to ensure sparse Hessians.

This elastic rods model was extended by [Pérez et al. 2015] to model elastically deforming connections between rods in order to simulate and design flexible rod meshes that closely approximate target deformations. A similar elastic joint model has been used to

model networks of elastic rods in [Zehnder et al. 2016] to design attractive and robust curves that fill in a target surface, in [Malomo et al. 2018] as a homogenized model for flexible spiral microstructures, and in [Schumacher et al. 2018] to determine the homogenized mechanical properties of isohedral polygonal tilings of rods. Baek et al. [2017] use the discrete elastic rods model to simulate traditional elastic gridshells, modeling the joint constraints with positional springs (rods remain twist free). They use Chebyshev nets to design the gridshell boundaries to achieve simple surface building blocks and show how these can assembled into more interesting structures. All these methods have shown the discrete elastic rods model's excellent agreement with physical prototypes.

We extend the elastic rods model with constraints to properly capture the connection of rod ends at the joints. Furthermore, we can handle arbitrary rod cross-section profiles, which provides additional degrees of freedom for design and enables us to incorporate important fabrication requirements.

3 X-SHELL REPRESENTATION

X-shells are networks of elastic beams that are linked at rotational joints (see Figure 3). We define the topology of the network with a graph, where each node represents a joint and each edge denotes a beam segment. We currently restrict our representation to quadrilateral topologies, where every interior node has valence four (hence the name *X-shell*) and boundary nodes have valence three, two, or one. A node with valence one denotes a free end of a beam.

We base our numerical simulation of X-shells on the discrete elastic rods model of [Bergou et al. 2010]. This model allows us to capture the elastic forces induced by bending, twisting, and stretching the structure's beams. Every beam segment connecting two joints is represented as a distinct elastic rod $r \in \mathcal{R}$, discretized with k linear elements. Constraints are added to properly couple the rod ends at the joints.

The joints themselves store no elastic energy; they simply constrain the incident beam segments to pass through a common point with compatible orientations and material axes. We impose these joint constraints exactly by constructing a reduced set of variables $\mathbf{x} \in \mathbb{R}^m$ that parametrize the space of properly joined rods. Specifically, for each joint we define nine variables controlling its position, orientation, opening angle, and two edge lengths (Figure 3). These variables fully determine the material axis angle and the two centerline positions for the terminal edge of each incident elastic rod (dark gray dots on the left in Figure 3). Note that two rods connecting across a joint have overlapping terminal edges; we halve the stretching stiffnesses of these edges to avoid double-counting their energy. All of the remaining rod centerline positions and material angles (for internal edges and free ends) are included as reduced variables in $\mathbf{x}$. The position and orientation of a single joint is fixed (removed from $\mathbf{x}$) to pin down the structure's rigid motion.

We can calculate the elastic energy stored in a given X-shell configuration with reduced variables $\mathbf{x}$ by summing the elastic energies of each rod:

$$E(\mathbf{x}, \mathbf{p}) := \sum_{r=1}^{|\mathcal{R}|} E_s(\mathbf{v}_r(\mathbf{x}), \mathbf{p}) + E_b(\mathbf{v}_r(\mathbf{x}), \mathbf{p}) + E_t(\mathbf{v}_r(\mathbf{x}), \mathbf{p}). \quad (1)$$

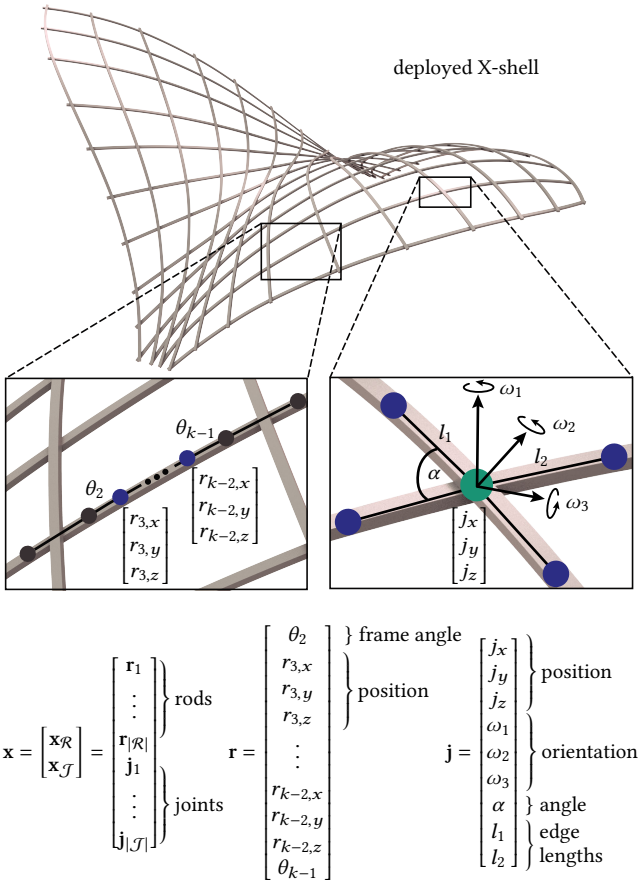


Fig. 3. X-shell representation. The network of elastic beams is modeled as a collection of individual rods coupled at rotational joints. The vector $\mathbf{x}$ aggregates all the variables of our reduced model representation designed to facilitate efficient and robust numerical computations. Note that two rod position vectors and one frame angle at each end of a rod are defined by the variables of the corresponding joint.

Here E_s, E_b, and E_t are the stretching, bending, and twisting energies, respectively, $\mathbf{v}_r$ is a nonlinear function computing the centerline position and material angle variables for rod r from the reduced variables $\mathbf{x}$, and $\mathbf{p}$ is a vector that stores the rest lengths for each beam segment. The per-edge stretching and per-vertex twisting energy expressions are taken from [Bergou et al. 2010]. For the per-vertex bending energy term, we note that averaging the material axes onto the vertices, as effectively done in [Bergou et al. 2010] and the recent computational fabrication papers using this method, yields a non-orthonormal material frame in the presence of twist. We prefer the original, more physically meaningful bending energy of [Bergou et al. 2008], which averages the two curvature *energies* computed with each incident edge's quadratic form. Our framework implements both bending energies, and we provide detailed expressions for all energy terms in the supplemental material. Our supplement also points out potential pitfalls in the use of the analytic gradients and Hessians provided in [Bergou et al. 2010] and provides alternative formulas with derivations.

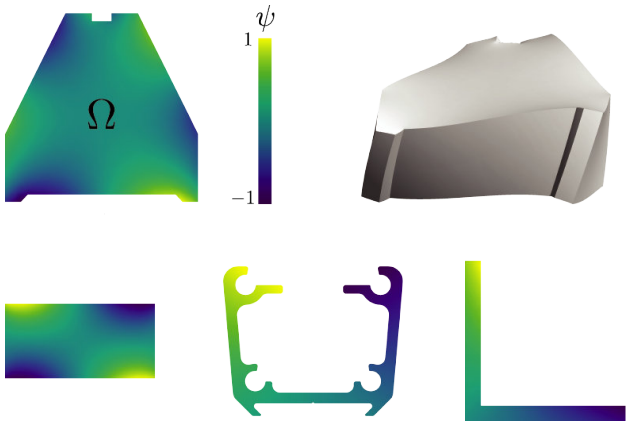


Fig. 4. Computing the twisting stiffness k_t for an arbitrary beam cross-section profile Ω. Colors show the warping displacement function Ψ. The examples in the bottom row are re-scaled to match the same color map.

Material Parameters. The deformation behavior of an X-shell as described by (1) depends on the geometric structure of the linkage and the elastic properties of the beams. For a single beam of a given homogeneous material, the stiffness parameters for stretching k_s, bending k_b, and twisting k_t depend on the Young's modulus E and shear modulus G of the material, as well as the cross-section profile Ω of the beam. Following [Landau et al. 1989], we can compute the stiffness parameters as:

- Stretching stiffness $k_s = EA$, where A is the area of Ω;
- Bending stiffness $k_b = EI$, where $I = \int_\Omega \begin{bmatrix} y^2 & -xy \\ -xy & x^2 \end{bmatrix} \mathrm{dxdy}$ is the moment of inertia tensor;
- Twisting stiffness $k_t = G \int_\Omega \| \begin{pmatrix} -y \\ x \end{pmatrix} + \nabla\psi \|^2 \, \mathrm{d}A$, where ψ is a scalar field defined on Ω that represents the out-of-plane displacement under an applied torsion; it satisfies the Laplace equation $\triangle\psi = 0$ in Ω with Neumann boundary conditions $\mathbf{n} \cdot \nabla\psi = \mathbf{n} \cdot \begin{pmatrix} y \\ -x \end{pmatrix}$ on $\partial\Omega$.

We compute these stiffnesses by meshing the cross-section with Triangle [Shewchuk 1996] and using exact quadrature. We solve the Laplace equation for the warping displacement ψ using quadratic finite elements [Hughes 2012].

An important benefit of this approach is that we can compute stiffness parameters for arbitrary beam profile geometries (see Figure 4). As we discuss in more detail in Section 7, this allows us to incorporate important architectural features into the design of X-shell beams, such as cable canals or frame fixtures. In addition, the cross-section geometry of the beams provides additional degrees of freedom for design, since the deployed state of an X-shell depends directly on the beam stiffness parameters (see also bottom row of Figure 14).

4 FORWARD SIMULATION

With the above representation for the geometry and material properties of an X-shell structure, we can now formulate our optimization algorithm to simulate the deployment from a given planar assembly state towards a 3D target state.

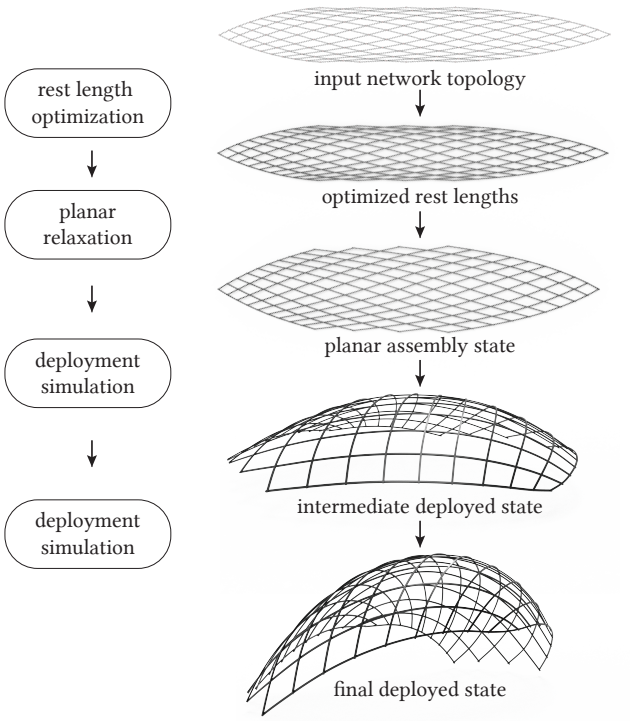


Fig. 5. Forward simulation: Given a user-provided network topology, our method first solves for a low-energy planar configuration that matches the desired joint locations, effectively determining the rest lengths of each beam. Minimizing elastic energy with released joint constraints, while keeping the structure planar, yields the assembly state. This state can then be deployed by solving for actuation torques for the joints that achieve a user-specified average opening angle.

We assume as input the graph topology of the beam network and the 2D locations of the joints in the flat configuration. These are the principal design parameters that we allow the user to interactively manipulate as explained in Section 7. The forward simulation process is split into two parts: an initialization that finds a low-energy planar assembly state, and the deployment algorithm that computes the deformation of the X-shell when applying torques at the joints (see Figure 5).

Rest-length Optimization. Interesting deployed shapes emerge if the joint positions deviate from a regular grid. In such cases, the rigid scissor mechanisms of the linkage become incompatible and rods deform and buckle out of plane when the joints are actuated. Irregular joint spacings mean that the rods potentially need to be bent in the flat assembly to smoothly interpolate the joint locations in a low-energy state. This poses the challenge of determining their rest lengths, which are essential parameters of the simulation algorithm. We collect the rest length of each rod into a vector $\mathbf{p} \in \mathbb{R}^{|\mathcal{R}|}$, which will also serve as the design parameters for the subsequent design optimization Section 5. The rest length of a rod is distributed

evenly across all the k edges making up the rod (to promote a uniform discretization). We infer the initial structure's rest lengths by simultaneously minimizing the elastic energy of (1) over both the rest-lengths of beam segments and their equilibrium configurations, while holding the joint positions fixed at their input locations:

$$\begin{aligned} &\min_{\mathbf{x},\mathbf{p}} E(\mathbf{x},\mathbf{p}) \\ &\text{s.t. } S_\mathsf{j}\mathbf{x} = S_\mathsf{j}\mathbf{x}_\text{input} \\ &\qquad \mathbf{p} > \epsilon. \end{aligned} \tag{2}$$

Here, S_j is a matrix selecting each joint's position variables from the full vector of state variables $\mathbf{x}$, and ϵ is a lower bound on the rest length variables. We set this lower bound equal to $1/100$ the size of the smallest distance between segments. Effectively, this optimization allows the beams to assume rest lengths that minimize the elastic energy for a given set of 2D joint locations. We further optimize this state as described in Section 5 to obtain a low-energy planar state in which the beams can be assembled with minimal required pre-stressing (see Figure 5).

Deployment Simulation. The planar beam network can now be deployed by imposing a uniform torque at every joint in the set of joints $\mathcal{J}'$ selected for actuation. Initially $\mathcal{J}' = \mathcal{J}$ includes all joints. In Section 6 we will discuss how to sparsify the actuation to obtain a small set of actuation points for physical deployment.

By increasing the torque acting on the joints, we expand the structure towards its target shape. Instead of prescribing torques directly, which would be difficult to quantify by the user, we solve for the torques that achieve a user-specified average opening angle of $\overline{\alpha}$ at the joints. Specifically, we simulate the deployment process by minimizing elastic energy while imposing a linear equality constraint on the average of the joints' angles to match $\overline{\alpha}$:

$$\begin{aligned} \mathbf{x}^*(\mathbf{p},\overline{\alpha}) = &\operatorname*{argmin}_{\mathbf{x}} E(\mathbf{x},\mathbf{p}) \\ &\text{s.t. } \tfrac{1}{|\mathcal{J}'|}\textstyle\sum_{j\in\mathcal{J}'} \alpha_j = \overline{\alpha}. \end{aligned} \tag{3}$$

The single Lagrange multiplier for this equality constraint tells us the magnitude of torque that must be applied to each actuation joint. To avoid inversions during the optimization, we also add the same lower-bound constraint as in (2) on the joint edge length variables. As we increase $\overline{\alpha}$, the structure is driven open as illustrated in Figure 1. At some point in this deployment, the planar equilibrium of the X-shell becomes unstable (a saddle point of the elastic energy), and the structure abruptly pops out of the plane into a lower energy configuration, realizing a curved surface in 3D.

Numerical Solver. We solve both minimization problems—inferring rest lengths and simulating deployment—using our own Newton-based solver, as we found off-the-shelf solvers like Knitro [Artelys 2019] neither fast enough nor sufficiently robust in dealing with the many saddle points in the energy landscape of our X-shells. We use an active set method to handle the bound constraints on the $\mathbf{p}$ variables and implement hard constraints (like the joint positions in the rest length solve) by removing the corresponding rows and columns of the Hessian.

We focus our discussion on the deployment simulation problem (3) because its linear equality constraint for actuation requires extra attention; the method for solving rest length is identical except for the removal of the extra system row and column used to impose the linear equality constraint. Each step of our Newton solver requires solving the linear system:

$$\begin{bmatrix} \tilde{H} & \mathbf{a} \\ \mathbf{a}^T & 0 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} -\mathbf{g} \\ \Delta\overline{\alpha} \end{bmatrix}, \tag{4}$$

for steps $\Delta\mathbf{x}$ and $\Delta\lambda$ of the state variables and the Lagrange multiplier for the linear equality constraint, respectively. Here, $\mathbf{g}$ is the gradient of the elastic energy (i.e., the net force acting on the X-shell), $\mathbf{a} \in \mathbb{R}^m$ is a vector with value $1/|\mathcal{J}'|$ in each actuated joint angle variable and 0 elsewhere, and $\tilde{H}$ is the Hessian of the elastic energy with respect to all optimization variables (including $\mathbf{p}$ in the rest length solve) after it has been modified to be positive definite as discussed below. $\Delta\overline{\alpha}$ is the requested increase in actuation angle.

To simulate the full deployment sequence, we subdivide the interval between the closed and open actuation angles into individual increments of size h. For each increment, we solve (4) once with $\Delta\overline{\alpha} = h$ and apply the full step $\Delta\mathbf{x}$ to the linkage. Now the linkage has the requested average opening angle but is not at a minimum energy configuration. Next we run several steps of our Newton solver with $\Delta\overline{\alpha} = 0$ to place the structure back in equilibrium.

Hessian modification. During the course of the simulation, and particularly at the onset of buckling, the Hessian H of the linkage's elastic energy frequently becomes indefinite. Simply solving (4) in this situation will often obtain a direction that *increases* the energy. Even if a descent direction is found, the linkage will still be attracted towards saddle points, which is not desirable. We therefore always check if H is positive definite (by attempting a Cholesky factorization) and modify it if not. A standard modification approach is to add a scaled multiple of the identity matrix $H + \sigma I$. [Nocedal and Wright 2006]. Applying the inverse of this modified matrix to $-\mathbf{g}$ can be interpreted as finding the step of a fixed length (depending on σ) that minimizes the local quadratic model; note that this local model is unbounded from below, so we need to fix the step length. However, since our state variables $\mathbf{x}$ have different meanings (Figure 3) and vastly different typical magnitudes, the Euclidean norm is a poor measure of distance. Instead, we use the linkage's mass matrix M as our metric; then applying the inverse of the modified matrix $\tilde{H} = H + \sigma M$ to $-g$ has a physical meaning of computing the step of fixed kinetic energy that minimizes elastic energy. We compute the full linkage's mass matrix M by assembling the per-rod lumped mass matrices using the Jacobian of our mapping from reduced state variables $\mathbf{x}$ to the individual rod variables. Note that we cannot use a lumped mass matrix for the full linkage (e.g., by summing the rows) because this generally has negative entries. However, M's sparsity pattern is a subset of H's, so no additional work is added to the factorization step.

We determine a reasonable initial guess for σ based on the largest eigenvalues of H and M (user-specified multiple of $\frac{\lambda_{\max}(H)}{\lambda_{\max}(M)}$) and employ a strategy similar to Algorithm 3.3 in [Nocedal and Wright 2006] for updating σ (multiplying it by 4 repeatedly until $\tilde{H}$ becomes positive definite and incrementally halving it when the modification is successful.)

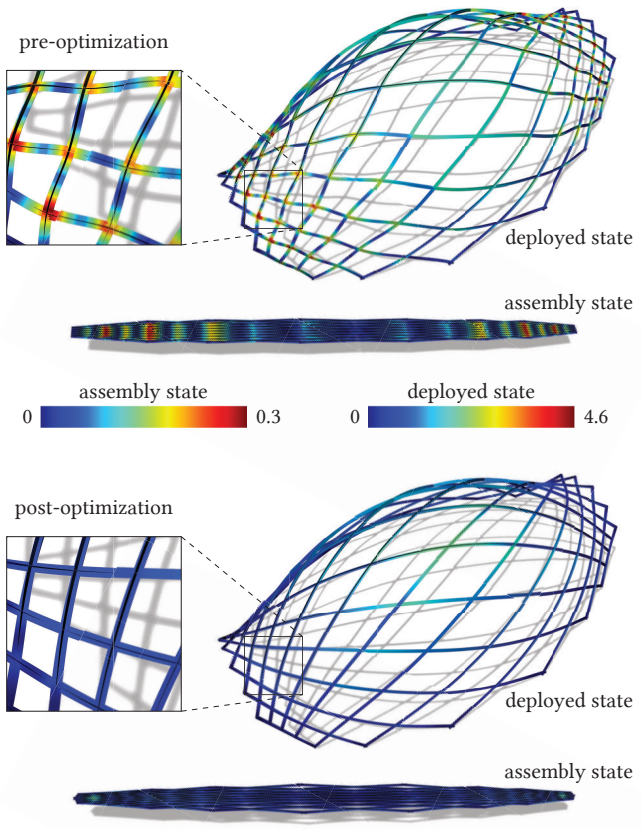


Fig. 6. Design optimization. A user-provided initial beam layout leads to high internal stresses (top), which manifests itself in undulations of the beams in the deployed state. After optimization, this undesirable behavior is eliminated while staying geometrically close to the desired target shape. The models are colored by the square root of their pointwise bending energies. The optimization reduces the total elastic energy of the X-shell by more than 5× in the deployed state and more than 7× in the assembly state.

Because $\tilde{H}$ is positive definite, we can compute its Cholesky factorization and solve (4) with block elimination:

$$\Delta\lambda = \frac{\Delta\overline{\alpha} - \mathbf{a}^T\tilde{H}^{-1}\mathbf{g}}{\mathbf{a}^T\tilde{H}^{-1}\mathbf{a}}, \quad \Delta\mathbf{x} = \tilde{H}^{-1}(-\mathbf{g} - \Delta\lambda\mathbf{a}).$$

If we start at or step near a saddle point, $\mathbf{g}$ will be approximately zero even though the energy is not at a minimum. In this case, we need to move in a direction of negative curvature [Gill et al. 1981] to quickly escape the saddle point. Thankfully, due to our Hessian modification, we have already computed the factorization needed to efficiently compute the eigenvector corresponding to the most negative eigenvalue (for the generalized eigenvalue problem $H\mathbf{d} = \lambda M\mathbf{d}$) using inverse power iteration. Whenever we detect that H is indefinite and $\|\mathbf{g}\|$ is below a user-defined multiple of the Newton solver's gradient tolerance, we use the C++ library Spectra to compute this eigenvector and add a scaled version of it to $\Delta\mathbf{x}$ to compute our line search direction $\mathbf{d}$. We pick the scaling so that the largest physical velocity it induces on the rod geometry's surface

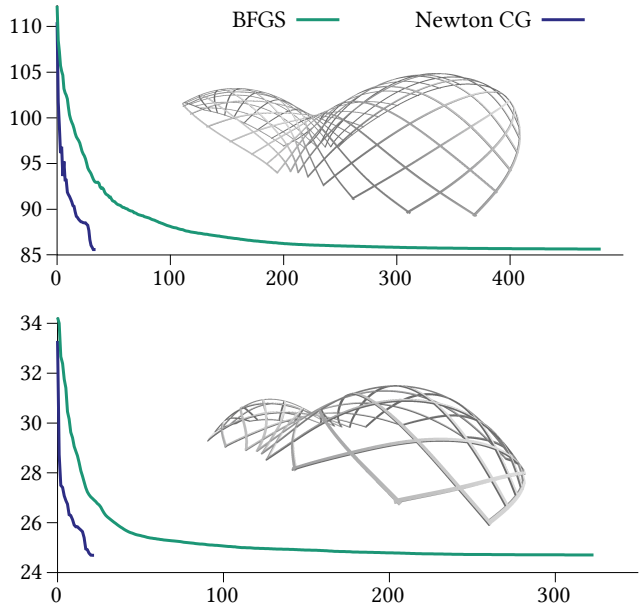


Fig. 7. Convergence of the design optimization for two different X-shells. The plots show the objective function of (5) over the number of iterations. A quasi-Newton BFGS solver requires significantly more iterations than our Newton-CG trust region approach.

(i.e. the maximum velocity of points on the swept cross-sections) is 1/100 the minimum arc length of any rod in the structure.

Our solver then does a standard backtracking Armijo line search [Nocedal and Wright 2006] to compute a step along $\mathbf{d}$ that reduces elastic energy.

Figure 1 and Figure 5 illustrate the deployment of an X-shell computed with our optimization. See also the accompanying video for more examples of the simulation algorithm and deployments of physical models.

5 DESIGN OPTIMIZATION

The forward simulation approach described above is an essential tool to explore the design space of X-shells. From an initial user-provided planar beam network, we can quickly compute deployed states to evaluate design alternatives. However, the complex deformation behavior of X-shells makes it unlikely that the deployed state has low internal stress. In practice we often observe high-frequency undulations in the beams, because the initial joint locations are not conducive to a low-energy deployed state (see Figure 6, top). Manually adjusting the rest configuration is ineffective, because small variations in the beam rest lengths can have complex global influence on the shape and elastic energy of the deployed state. Optimization of design parameters is thus crucial to obtain high-quality X-shell designs.

This optimization problem is challenging, not only because simulating the deployment is already a highly nonlinear optimization problem in itself, but also because we care about two states of the structure: the flat assembly state and the curved deployed state. We

want the assembly state to be truly flat, the deployed state to produce a desired 3D shape, and both states to experience reasonable stresses/elastic energy. We formulate these goals as a minimization of the following objective function:

$$J(\mathbf{p}) = \frac{\gamma}{E_0} E(\mathbf{x}^*_{2\mathrm{D}}(\mathbf{p}), \mathbf{p}) + \frac{(1-\gamma)}{E_0} E(\mathbf{x}^*_{3\mathrm{D}}(\mathbf{p}), \mathbf{p}) + \frac{\beta}{2l_0^2} T(\mathbf{x}^*_{3\mathrm{D}}(\mathbf{p})), \quad (5)$$

$$\mathbf{x}^*_{2\mathrm{D}}(\mathbf{p}) \coloneqq \mathbf{x}^*(\mathbf{p}, \overline{\alpha}_0) \qquad \text{(flat configuration state vars.)},$$
$$\mathbf{x}^*_{3\mathrm{D}}(\mathbf{p}) \coloneqq \mathbf{x}^*(\mathbf{p}, \overline{\alpha}_{\mathrm{tgt}}) \quad \text{(deployed configuration state vars.)},$$

subject to the following planarity and minimum angle constraints on the flat configuration:

$$c(\mathbf{p}) = \|S_z \mathbf{x}^*_{2\mathrm{D}}(\mathbf{p})\|^2 = 0$$
$$\alpha_{\min}(\mathbf{p}) = KS(S_\alpha \mathbf{x}^*_{2\mathrm{D}}(\mathbf{p}), s) \geq \epsilon.$$

Here $\gamma \in [0, 1]$ trades off between preferring low energy in the deployed or flat state, $\beta > 0$ controls how closely we want to fit the deployed structure's joints to user-provided target geometry, target fitting energy T (defined below) penalizes the deployed structure's distance to this target geometry, and E_0 and l_0 are normalization constants (the initial deployed structure's energy and the length of its bounding box diagonal, respectively).

S_z and S_α are matrices selecting the joints' z coordinates and opening angles α from the full vector of state variables (these $|\mathcal{J}| \times m$ matrices have a single 1 in each row and zeros everywhere else).

We use the popular constraint aggregation function $KS(\alpha, s) = -s \log(\sum_i e^{-\alpha_i/s})$ [Kreisselmeier and Steinhauser 1979] as a smooth, conservative approximation to the minimum. This smooth function converges to the exact, non-differentiable minimum as $s \to 0$, and we found $s = 0.01$ to be a good trade-off between smoothness and approximation fidelity.

The initial actuation angle $\overline{\alpha}_0$ is chosen just large enough to keep the structure's minimum opening angle greater than some threshold ϵ to avoid interpenetrations of neighboring beams in the assembly state. The target actuation angle α_{tgt} is specified by the user and can be adjusted interactively during the optimization process.

Surface Fitting. The fitting energy term T penalizes deviation of the joint positions from a user-provided target surface. This allows the structure to slide along the design surface to minimize internal stress and thus provide a more efficient X-shell. However, without further constraints, joints displacements could be significant, potentially even closing up the structure to reduce the elastic energy. We therefore also incorporate in the fitting term a set of user-specified target positions $\mathbf{x}_{\mathrm{tgt}}$ for each joint, to which the joints are fit with a smaller weight. This is analogous to the commonly used combination of point-to-surface and point-to-point distance terms for non-rigid registration [Chang et al. 2012]. The full fitting term is then given by

$$T(\mathbf{x}) \coloneqq \frac{1}{2}\|\mathbf{x} - P_{\mathrm{surf}}(\mathbf{x})\|^2_{W_{\mathrm{surf}}} + \frac{1}{2}\|\mathbf{x} - \mathbf{x}_{\mathrm{tgt}}\|^2_W, \quad (6)$$

where $P_{\mathrm{surf}}(\mathbf{x})$ projects the joint positions of the deployed structure configuration $\mathbf{x}$ onto their closest points on the target surface. The diagonal matrix W_{surf} holds user-provided weights that indicate the importance of fitting each individual joint to the surface (only entries corresponding to joint position state variables are nonzero). Likewise, W is a diagonal matrix of fitting weights for each target joint position in $\mathbf{x}_{\mathrm{tgt}}$. We ensure all of the fitting weights (in W_{surf} and W) sum to one so that β directly controls the over-all importance of fitting.

Numerical Solver. Especially for high values of β, the three conflicting terms of the objective function lead to an ill-conditioned optimization problem. This makes it is essential to incorporate accurate Hessian information from the start. We found that a quasi-Newton method such as BFGS [Nocedal and Wright 2006] requires too many iterations to converge (see Figure 7); recall that each iteration of the design optimization requires re-solving for both the flat and deployed equilibria (3). On the other hand, computing the full, dense Hessian of J and c with respect to the design variables $\mathbf{p}$ is intractable. We therefore propose to solve this optimization problem with a Newton-CG trust region method. This approach allows larger step sizes and thus converges in fewer iterations, but only requires computing analytic gradients of J and c with respect to the design variables, as well as Hessian-vector products (the gradients' directional derivatives). We show in Appendix B how to efficiently compute this required derivative information and provide more implementation details in Appendix A.

Optimizing X-shells. This design optimization of (5) facilitates an inverse design mode where the user edits the target geometry and alters the fitting weights. It can also be used as an automated tool for improving an existing structure created with our interactive design interface described in Section 7; in this mode we choose $\mathbf{x}_{\mathrm{tgt}}$ as the joint positions of the original deployed structure and construct a target surface by triangulating the linkage's quads. We refine this mesh with two steps of Loop subdivision. For all examples shown in the paper we choose W_{surf} to have a total weight of 0.9 spread equally across all joints. The remaining 0.1 is distributed among the target position fitting weights in W; we found it helpful to set a 10× higher weight in W for the joints of valence 2 (sharp corners) than the rest to better preserve the extremities of a design.

Figure 6 illustrates how this optimization significantly improves the quality of a given X-shell design. All examples shown in the paper employ this method to jointly optimize flat and deployed state.

6 OPTIMIZING ACTUATION

A key benefit of X-shells is that they can be intrinsically actuated at the joints to deploy to the desired target shape, in contrast to traditional grid shells whose boundary nodes need to be explicitly forced towards pre-defined target positions.

The deployment simulation described in Section 4 applies the same torque on each of the X-shell's joints. While this is ideal for fully opening all parts of the structure and avoiding large localized deformations and forces, it is not practical for most physical deployment scenarios to equip each joint with an actuator. We note that while X-shells generally deploy into similar shapes regardless of the actuation forces applied due to their resemblance to 1-DoF scissor linkage structures, the precise deployed shape still depends on where forces are applied since they must elastically deform the beams. We therefore propose an optimization algorithm to solve for

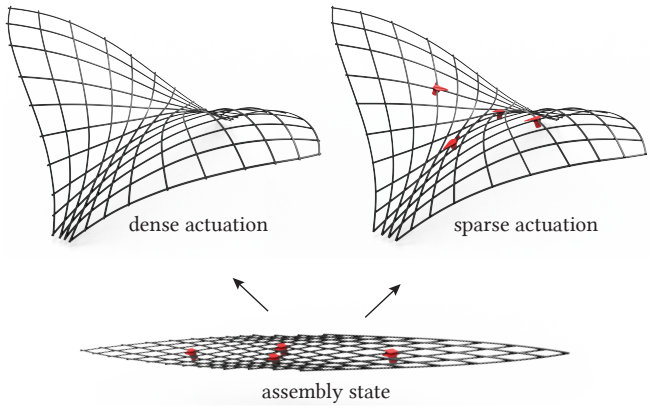


Fig. 8. Sparse deployment. A set of just four actuators computed by our sparsifying algorithm deploys this X-shell to a target state that is very close to the densely actuated model, where each joint would need to apply an expansive torque.

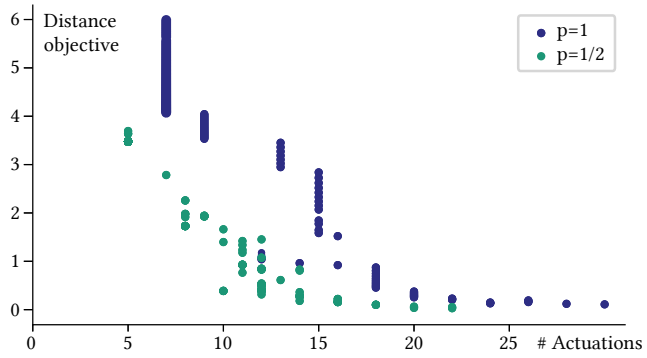


Fig. 9. Performance of the ℓ_p regularization term for optimizing actuation. Each dot corresponds to a different value for η that controls the trade-off between fitting term and sparseness of the actuation. $p = 1/2$ consistently leads to fewer actuators *and* smaller deviation from the target shape.

a sparse set of actuation torques that best deploy the X-shell to the desired target state.

Arbitrary Torque Actuation. We first introduce a slightly different formulation of Equation (3) for the deployed linkage's equilibrium solve:

$$\tilde{\mathbf{x}}(\boldsymbol{\tau}) = \underset{\mathbf{x}}{\operatorname{argmin}}\, E(\mathbf{x}) - \boldsymbol{\tau} \cdot S_\alpha \mathbf{x}$$

Here we drop the dependence on the design parameters $\mathbf{p}$, which we assume are known at this stage. Recall that S_α is a matrix that selects the joints' opening angles α from the full vector $\mathbf{x}$ of state variables. Note that the solution to (3) can simply be written as $\mathbf{x}^*(\overline{\alpha}) = \tilde{\mathbf{x}}(\boldsymbol{\tau}^*(\overline{\alpha}))$, where $\boldsymbol{\tau}^*(\overline{\alpha})$ is a vector with all entries equal to the Lagrange multiplier for actuation in the angle constraint of (3). This formulation allows us to simulate the equilibrium corresponding to any vector of torque magnitudes $\boldsymbol{\tau} = [\tau_1, \ldots, \tau_{|\mathcal{J}|}]^T \in \mathbb{R}^{|\mathcal{J}|}$ applied to the set $\mathcal{J}$ of 1-DoF rotational joints.

Sparse Actuation. Our goal is now to find a sparse vector $\boldsymbol{\tau}$ that deploys the X-shell to the same target state as the dense actuation at each joint, so that only few joints need to be equipped with physical actuators. For this purpose we solve the following minimization problem:

$$\min_{0 \le \tau_i \le \tau_{\max}} T(\tilde{\mathbf{x}}(\boldsymbol{\tau})) + \eta \sum_i \tau_i^p, \tag{7}$$

where $\tau_{\max}$ is the maximum torque that can be safely applied by an actuator, η is the weight for the sparsifying regularization term, $0 < p \le 1$ is the power for approximating the ℓ_0 "norm," and T is the fitting energy from (6). Here, the target joint positions are taken from the densely actuated deployed equilibrium, and the target surface is constructed by triangulating the linkage's quads and performing Loop subdivision. For the examples shown in the paper, we opted to keep all joints of the sparsely deployed structure as close as possible to their original positions, so we set $W_{\text{surf}} = 0$ and weighted each joint equally in W.

When $p = 0$, the regularization term directly counts the number of actuators, but results in an extremely challenging optimization problem. The common approach is to replace this non-convex ℓ_0 "norm" with the convex approximation obtained by setting $p = 1$. Choosing the ℓ_1 norm indeed leads to a sparse vector of torques in our setting, but does not drive the smaller torques to zero quite as aggressively as we want. Also, since our problem is already a nonconvex minimization, it is less essential to use a convex regularization term.

Several techniques have been proposed to improve on ℓ_1 with nonconvex regularization terms. For instance, one can iteratively weight the ℓ_1 norm to place more emphasis on the small entries [Candes et al. 2008] or reformulate the ℓ_0 term using complementarity constraints [Feng et al. 2013]. The former is not efficient in our setting since each weighted instance of the ℓ_1 sparsification problem is still a nonconvex minimization, and the latter did not work well in our experiments. Instead we chose to minimize (7) with $0 < p < 1$. For this choice of p, calculating the derivatives of the regularization term involves nearly dividing by zero as torques approach zero. This is avoided in [Chartrand 2007] by adding an epsilon term to the denominator, but we prefer to introduce a change of variables, defining a new variable vector $\mathbf{t} \in \mathbb{R}^{|\mathcal{J}|}$ with $t_i = \tau_i^p$. Then we can write the minimization problem as:

$$\min_{0 \le t_i \le \tau_{\max}^p} T(\tilde{\mathbf{x}}(\mathbf{t}^{1/p})) + \eta \sum_i t_i, \tag{8}$$

where the regularization term has now become a standard ℓ_1 norm in the new variables. We note the similarity of this formulation to the SIMP method used to encourage binary designs in topology optimization [Sigmund 2001].

Figure 8 shows an example where only four actuators are sufficient to deploy the X-shell to its desired target configuration. By varying η, we can explore the trade-off between preserving the deployed shape and using fewer actuators. To compare the various choices of regularization terms, we ran a sweep over η and plotted the approximation distance as a function of the number of actuators; see Figure 9 for the results. We found that $p = 1/2$ leads to consistently better results compared to the standard ℓ_1 norm ($p = 1$) in all our experiments. The other values for p that we tested

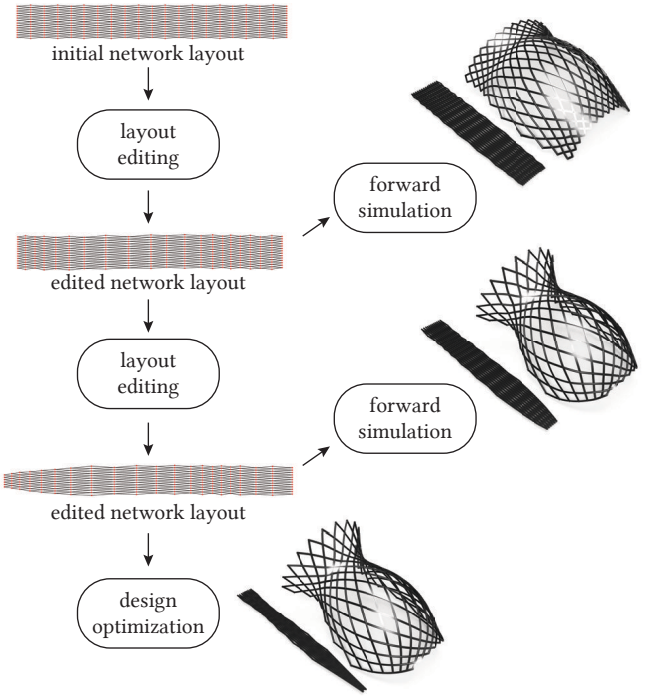


Fig. 10. Overview of our design tool. Starting from a regular quadrilateral grid, the designer adds or deletes nodes in the graph, controls the positioning of joints using a variety of direct manipulation tools, and evaluates the resulting deployed shape computed with our forward simulation algorithm. Once an overall satisfactory shape has been found, the optimization method refines the design to reduce the internal elastic energy of both the flat assembly state and deployed target state.

$(1/3, 1/4, 1/8, \ldots)$ gave indistinguishable results from $p = 1/2$ but took longer to converge.

Note that since the deployed equilibrium shape is closely preserved, the sparse set of actuators still need to output the same amount of total work to deploy the structure as the original dense actuators. Therefore the torque each actuator must apply is roughly inversely proportional to the number of actuators. The torque bound $\tau_{\max}$ will automatically stop further sparsification once the actuators' force output limits are reached.

7 RESULTS

We integrate forward simulation (Section 4) and design optimization (Section 5) into a form-finding tool that enables interactive exploration of X-shell designs. As illustrated in Figure 10 and the accompanying video, our tool allows the designer to manipulate both the topology of the beam network and the initial joint locations. Our implementation is integrated in the professional modeling software Rhinoceros3D in order to leverage direct manipulation tools such as cage-based editing, smart brushes, and smooth warping functions to manipulate joint locations.

Figure 14 shows a collection of X-shells to give a sense of the kind of designs possible with our approach. Note that none of these examples could be realized with traditional grid shells, highlighting how our approach offers new form typologies for design. The examples in the bottom row of the figure illustrate how the deformation behavior of an X-shell depends on the cross-section profile of the beams, which in turn determines the stiffness parameters for stretching, bending, and twisting as discussed in Section 3.

Table 1. Statistics for some of our designs. Timings for forward simulation (sim) and design optimization (opt) are in seconds, measured on a MacBook Pro (2017), 3.1 GHz Intel core i7-7920HQ.

X-shell	segments	joints	DoF	sim	opt
Fig. 6	264	146	8970	1.618	97.56
Fig. 7 top	440	237	14893	4.023	58.68
Fig. 7 bot.	216	122	7362	2.202	20.5
Fig. 10	438	238	14844	3.00	167
Fig. 13	288	157	9765	2.09	17.36

Table 1 lists some performance data and other statistics of the shown examples.

Physical Prototypes. Figure 2 and Figures 11 to 13 show physical prototypes that we fabricated from different standard materials: wood, aluminum, and fiberglass. These examples indicate how different graph layouts and beam material properties can achieve interesting freeform structures that correspond closely to the simulated digital models.

All our physical prototypes use simple riveting to implement rotational joints for the beams. The model in Figure 2 is assembled from L-shaped aluminum profiles of 10mm × 10 mm with 1 mm material thickness and deploys into a negatively curved target state approximating a hyperboloid. Observe how the beams are bent in the flat state, but almost straight but twisted in the deployed state.

For deployment, we did not have torque actuators at our disposal, so we instead opted for a simpler strategy that pulls the structure open using strings (see Figure 12 and accompanying video). Even though the corresponding forces differ significantly from the calculated torque actuation, the optimized actuation joint set still informed us where to attach the ropes, and the deployed shape was closely reproduced. This points to a key advantage of X-shells: essentially, a well-designed X-shell behaves like a 1-DoF linkage, with just a single low energy deployment path. In this sense, the target shape is directly encoded in the flat assembly state, which makes the deployment robust under deviations from the prescribed actuation mechanism. There is one subtlety though: Since the assembly state is flat, there is ambiguity (mirror symmetry) in the vertical direction when deploying the structure. Friction and gravity typically break this symmetry in the desired way, but sometimes explicit control is necessary as in Figure 12.

Architectural Design Study. Figure 1 shows a speculative design studies to illustrate potential applications in architecture. X-shells offer a number of key advantages for construction: simple assembly on the ground from linear beam elements joined with identical rotational joints; fast and robust deployment; no need for any temporary formwork. This makes them ideally suited for temporary spaces, cost-constrained projects, or environments where involved

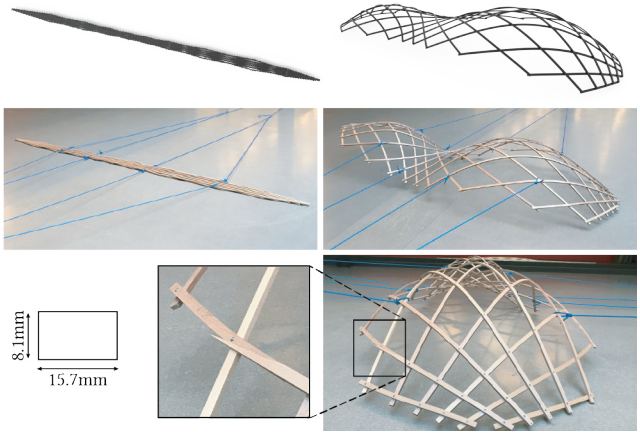


Fig. 11. Positive and negative curvature in a double-dome structure composed of wooden slats with a rectangular profile of 15.7mm × 8.1mm. As shown in the zoom, a material defect in one of the beams—a poorly aligned wood grain direction—led to a fracture at the last step of opening.

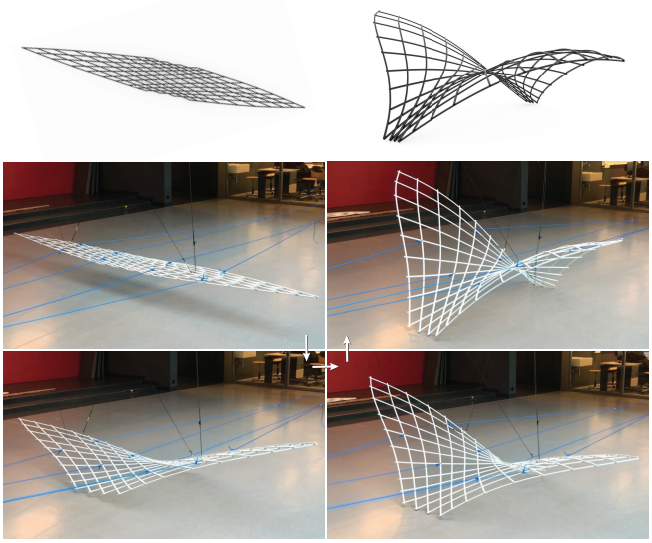


Fig. 12. Resting on only few ground contact points, this double-wing X-shell naturally supports complex free boundaries. The fiberglass beams have a rectangular profile of 12mm × 8 mm. In this example, we use additional ropes to suspend the structure, which ensures that the vertical symmetry of the planar assembly state is broken in the desired way so that the "wings" are deployed upwards. The bottom images show intermediate stage of the deployment process.

construction processes are difficult to perform. Additional facade elements, such as glass panels, can be added after deployment. Certain types of cladding could even be integrated during assembly. For example, the flexible EFTE membranes shown in Figure 1 could be attached to the beams in the flat state, then be inflated to form

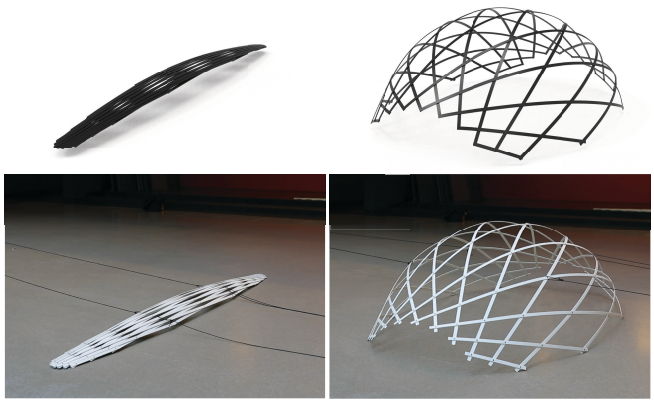


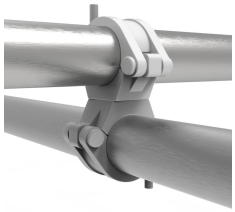
Fig. 13. This positively curved surface is assembled from fiberglass beams with rectangular profile of 15mm × 3mm. Due to the beams' weak resistance to out-of-plane bending, the assembly configuration does not lie flat, neither in our simulation nor in the fabricated prototype.

stiff cushions after deployment. Potentially, deployment could even be driven by the inflation process, providing a fairly uniform and distributed expansive actuation across the structure.

While our focus in this paper is on large-scale applications in architecture, our physical prototypes inspired other use cases, such as kitchen utensils (deployable pasta drainer), a retractable children's playpen, or deployable furniture such as foldable chairs or coat hangers (see inset). For these applications, the compactness of the assembly state offers additional advantages for transport and storage.

8 LIMITATIONS AND FUTURE WORK

Our current optimization-based design approach makes several simplifying assumptions with respect to the physical realizations of our prototypes or potential industrial applications. Joints in our prototypes are realized by rivets to support the required rotational motion. We do not currently take into account distortions in the beam stiffnesses at the joints due to the hole required to accommodate the shaft passing through both connected beams. These holes can create weak points that led to material failure in the prototype using wooden slats (Figure 11).



The geometric representation of X-shells assumes that crossing beams spatially coincide at the joints, while physical realizations require an offset distance between joined beams. For the rectangular and L-shaped beam profiles of our physical prototypes, we observed that this discrepancy did not lead to significant error between simulated and observed model. For large-scale architectural deployments, joints such as the one shown in the inset could be used. However, in this case, the offset distance between joined beams could be significant and should be accounted for in the model.

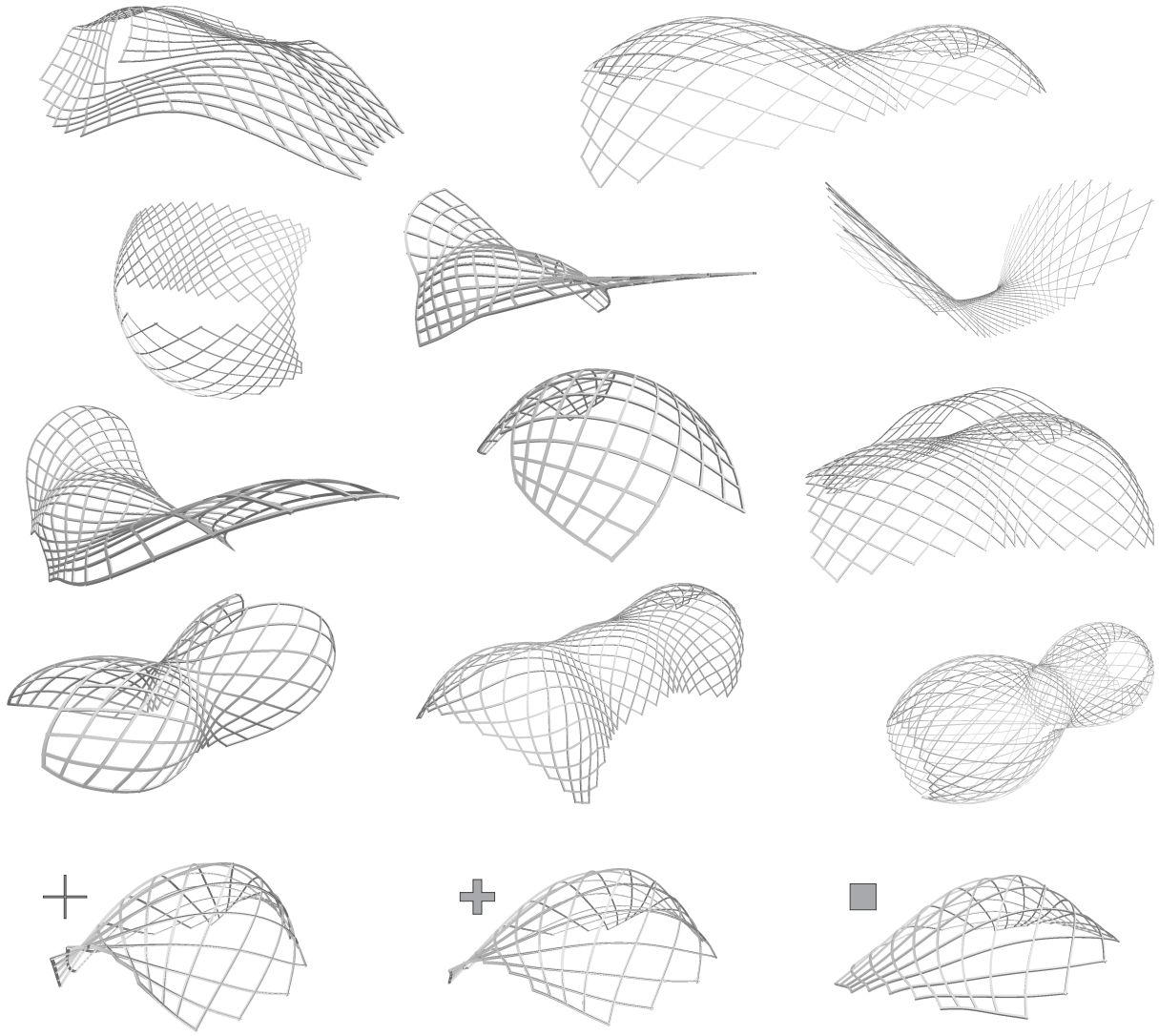


Fig. 14. X-shell design zoo. A large variety of double curved geometries can be deployed from flat assembly states. The bottom row shows how different cross-section geometries of elastic beams (sketched next to the models) of the same material lead to different stiffness parameters, and consequently different deployed states for the same assembly configuration. This offers additional control for form finding and design optimization.

Our current representation is restricted to valence-4 interior joints. Extensions to more general networks require more complex joints, which poses an interesting avenue for future research. The core of our simulation approach should extend naturally though.

The simulation framework supports varying the cross-sectional profile and material properties along a beam. Currently this functionality is not exposed in the design framework and none of our prototypes explores these additional degrees of freedom for design. Since modern production processes such as laser cutting or extrusion casting allow varying the cross-sectional profile of beams, this could be an interesting extension to explore in the future. Similarly, we currently assume that all beams have a straight rest shape. Curved rest shapes offer even more design flexibility and might help to improve the structural performance of an X-shell.

Each X-shell design has to negotiate trade-offs in geometric and physical complexity, structural efficiency, and artistic expression. Interesting questions arise in how to best navigate these trade-offs and more research is needed to develop effective design tools that can incorporate all the above mentioned additional design parameters not currently considered in our system.

Not all X-shells perform equally well. Certain target shapes cannot be approximated well or would require X-shells with high internal stress, which would make them impractical to build. At this stage we do not have a formal classification of the space of shapes that permit efficient X-shell realizations, nor do we have a solution for

the general inverse design problem. Our optimization method requires a reasonable initialization that is currently provided by the forward simulation approach. How to automatically find such an initialization for arbitrary design surfaces remains an open question. Similarly, a comprehensive structural analysis of X-shells under different load scenarios offers interesting avenues for future research.

9 CONCLUSION

We introduced X-shells as a new type of deployable structure and presented novel computational methods to explore X-shell designs. The key aspect of our work is a careful analysis and design of numerical solvers based on a reduced model representation and suitable parameterization of the design space. We found this to be essential to obtain robust and efficient algorithms for handling the complex buckling behavior of interconnected elastic beams that give rise to a geometrically rich space of deployable forms.

X-shells are just one instance of a broader class of deployable structures that are distinguished by a coupled interaction of continuously deforming elements. For example, bird wings are composed of individual flexible feathers that interact in complex ways to optimize flight performance. These and other examples pose fascinating new research challenges in terms of mathematical and geometric modeling, robust and efficient numerical simulation and optimization, as well as novel interactive design metaphors.

REFERENCES

Eugene L Allgower and Kurt Georg. 2012. *Numerical continuation methods: an introduction*. Vol. 13. Springer Science & Business Media.

Artelys. 2019. Artelys Knitro - Nonlinear optimization solver. (2019). https://www.artelys.com/en/optimization-tools/knitro

Marco Attene, Marco Livesu, Sylvain Lefebvre, Thomas Funkhouser, Stefano Ellero, Szymon Rusinkiewicz, Jonàs Martínez, and Amit Haim Bermano. 2018. *Design, Representations, and Processing for Additive Manufacturing*. Vol. 10. Morgan & Claypool Publishers. 146 pages. https://hal.inria.fr/hal-01836525

Moritz Bächer, Stelian Coros, and Bernhard Thomaszewski. 2015. LinkEdit: Interactive Linkage Editing Using Symbolic Kinematics. *ACM Trans. Graph.* 34, 4, Article 99 (July 2015), 8 pages. https://doi.org/10.1145/2766985

Changyeob Baek, Andrew O. Sageman-Furnas, Mohammad K. Jawed, and Pedro M. Reis. 2017. Form finding in elastic gridshells. *Proceedings of the National Academy of Sciences* (2017). https://doi.org/10.1073/pnas.1713841115 arXiv:http://www.pnas.org/content/early/2017/12/13/1713841115.full.pdf

Miklós Bergou, Basile Audoly, Etienne Vouga, Max Wardetzky, and Eitan Grinspun. 2010. Discrete Viscous Threads. *ACM Trans. Graph.* 29, 4, Article 116 (July 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 Trans. Graph.* 27, 3, Article 63 (Aug. 2008), 12 pages. https://doi.org/10.1145/1360612.1360662

Amit H. Bermano, Thomas Funkhouser, and Szymon Rusinkiewicz. 2017. State of the Art in Methods and Representations for Fabrication-Aware Design. *Comput. Graph. Forum* 36, 2 (May 2017), 509–535. https://doi.org/10.1111/cgf.13146

Florence Bertails, Basile Audoly, Marie-Paule Cani, Bernard Querleux, Frédéric Leroy, and Jean-Luc Lévêque. 2006. Super-helices for Predicting the Dynamics of Natural Hair. *ACM Trans. Graph.* 25, 3 (July 2006), 1180–1187. https://doi.org/10.1145/1141911.1142012

Bernd Bickel, Paolo Cignoni, Luigi Malomo, and Nico Pietroni. 2018. State of the Art on Stylized Fabrication. *Computer Graphics Forum* (2018). https://doi.org/10.1111/cgf.13327

Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. 2008. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier analysis and applications* 14, 5-6 (2008), 877–905.

Jean-François Caron, Olivier Baverel, Frédéric Tayeb, and Lionel Du Peloux. 2012. Gridshells in composite materials : Construction of a 300 m(2) Forum for the Solidays' Festival in Paris. *Physical Review E : Statistical, Nonlinear, and Soft Matter Physics* 22, 3 (2012), 408–414. https://hal-enpc.archives-ouvertes.fr/hal-00799095

Paolo Celli, Connor McMahan, Brian Ramirez, Anton Bauhofer, Christina Naify, Douglas Hofmann, Basile Audoly, and Chiara Daraio. 2018. Shape-morphing architected sheets with non-periodic cut patterns. *Soft matter* (2018).

Duygu Ceylan, Wilmot Li, Niloy J. Mitra, Maneesh Agrawala, and Mark Pauly. 2013. Designing and Fabricating Mechanical Automata from Mocap Sequences. *ACM Trans. Graph.* 32, 6, Article 186 (Nov. 2013), 11 pages. https://doi.org/10.1145/2508363.2508400

Will Chang, Hao Li, Niloy Mitra, Mark Pauly, and Michael Wand. 2012. Dynamic Geometry Processing. In *Eurographics 2012 - Tutorials*. The Eurographics Association. https://doi.org/10.2312/conf/EG2012/tutorials/t3

R. Chartrand. 2007. Exact Reconstruction of Sparse Signals via Nonconvex Minimization. *IEEE Signal Processing Letters* 14, 10 (Oct 2007), 707–710. https://doi.org/10.1109/LSP.2007.898300

Xiang Chen, Changxi Zheng, Weiwei Xu, and Kun Zhou. 2014. An Asymptotic Numerical Method for Inverse Elastic Shape Design. *ACM Trans. Graph.* 33, 4, Article 95 (July 2014), 11 pages.

Yanqing Chen, Timothy A Davis, William W Hager, and Sivasankaran Rajamanickam. 2008. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software (TOMS)* 35, 3 (2008), 22.

Lionel Du Peloux. 2017. *Modeling of bending-torsion couplings in active-bending structures. Application to the design of elastic gridshells.* Ph.D. Dissertation. Université Paris Est, École des Ponts Paris Tech.

Mingbin Feng, John E Mitchell, Jong-Shi Pang, Xin Shen, and Andreas Wächter. 2013. Complementarity formulations of ℓ_0-norm optimization problems. *Industrial Engineering and Management Sciences. Technical Report. Northwestern University, Evanston, IL, USA* (2013).

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, Article 66 (July 2014), 12 pages. https://doi.org/10.1145/2601097.2601106

Philip E Gill, Walter Murray, and Margaret H Wright. 1981. Practical optimization. (1981).

Ruslan Guseinov, Eder Miguel, and Bernd Bickel. 2017. CurveUps: Shaping Objects from Flat Plates with Tension-actuated Curvature. *ACM Trans. Graph.* 36, 4, Article 64 (July 2017), 12 pages.

Elisa Hernández, Stefan Sechelmann, Thilo Rörig, and Christoph Gengnagel. 2012. Topology Optimisation of Regular and Irregular Elastic Gridshells by Means of a Non-linear Variational Method. *Advances in Architectural Geometry, 2012*. https://doi.org/10.1007/978-3-7091-1251-9_11

Elisa Lafuente Hernández, Olivier Baverel, and Christoph Gengnagel. 2013. On the Design and Construction of Elastic Gridshells with Irregular Meshes. *International Journal of Space Structures* 28, 3-4 (2013), 161–174. https://doi.org/10.1260/0266-3511.28.3-4.161

Thomas JR Hughes. 2012. *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation.

Martin Kilian, Aron Monszpart, and Niloy J. Mitra. 2017. String Actuated Curved Folded Surfaces. *ACM Trans. Graph.* 36, 4, Article 64a (May 2017). https://doi.org/10.1145/3072959.3015460

Mina Konaković, Keenan Crane, Bailin Deng, Sofien Bouaziz, Daniel Piker, and Mark Pauly. 2016. Beyond Developable: Computational Design and Fabrication with Auxetic Materials. *ACM Trans. Graph.* 35, 4, Article 89 (July 2016), 11 pages.

Mina Konaković-Luković, Pavle Konaković, and Mark Pauly. 2018a. Computational Design of Deployable Auxetic Shells. In *Advances in Architectural Geometry 2018*. 94–111.

Mina Konaković-Luković, Julian Panetta, Keenan Crane, and Mark Pauly. 2018b. Rapid Deployment of Curved Surfaces via Programmable Auxetics. *ACM Trans. Graph.* (Aug. 2018).

G. Kreisselmeier and R. Steinhauser. 1979. Systematic Control Design by Optimizing a Vector Performance Index. *IFAC Proceedings Volumes* 12, 7 (1979), 113 – 117. https://doi.org/10.1016/S1474-6670(17)65584-8

L.D. Landau, E.M. Lifshitz, and J.B. Sykes. 1989. *Theory of Elasticity*. Pergamon Press. https://books.google.ch/books?id=YjDiQwAACAAJ

Ian Liddell. 2015. Frei Otto and the development of gridshells. *Case Studies in Structural Engineering* 4 (2015), 39 – 49. https://doi.org/10.1016/j.csse.2015.08.001

J. Lienhard. 2014. *Bending-Active Structures: Form-finding Strategies Using Elastic Deformation in Static and Kinetic Systems and the Structural Potentials Therein*. Universität Stuttgart Inst. f. Tragkonstr.

Alessandro Liuti, Alberto Pugnale, and Sofia Colabella. 2017. The Airshell prototype: a timber gridshell erected through a pneumatic formwork. (09 2017).

Luigi Malomo, Jesús Peréz, Emmanuel Iarussi, Nico Pietroni, Eder Miguel, Paolo Cignoni, and Bernd Bickel. 2018. FlexMaps: Computational Design of Flat Flexible Shells for Shaping 3D Objects. *ACM Trans. on Graphics - Siggraph Asia 2018* 37, 6 (dec 2018), 14. https://doi.org/10.1145/3272127.3275076.

JRRA Martins and Nicholas MK Poon. 2005. On structural optimization using constraint aggregation. In *VI World Congress on Structural and Multidisciplinary Optimization WCSMO6, Rio de Janeiro, Brasil*.

Romain Mesnil. 2013. *Stability of elastic gridshells*. Ph.D. Dissertation. https://doi.org/10.13140/RG.2.1.1727.3449

Romain Mesnil, Cyril Douthe, and Olivier Baverel. 2017. Non-standard patterns for gridshells: fabrication and structural optimization. 4 (12 2017), 277–286.
Jorge Nocedal and Stephen J. Wright. 2006. *Numerical Optimization* (second ed.). Springer, New York, NY, USA.
Dinesh K. Pai. 2002. STRANDS: Interactive Simulation of Thin Solids using Cosserat Models. *Computer Graphics Forum* (2002). https://doi.org/10.1111/1467-8659.00594
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 (July 2017), 12 pages.
Jesús Pérez, 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. Graph.* 34, 4, Article 138 (July 2015), 12 pages.
Nico Pietroni, Davide Tonelli, Enrico Puppo, Maurizio Froli, Roberto Scopigno, and Paolo Cignoni. 2015. Statics Aware Grid Shells. *Comput. Graph. Forum* 34, 2 (May 2015), 627–641. https://doi.org/10.1111/cgf.12590
Helmut Pottmann, Michael Eigensatz, Amir Vaxman, and Johannes Wallner. 2015. Architectural Geometry. *Computers and Graphics* 47 (2015), 145–164. https://doi.org/10.1016/j.cag.2014.11.002
G Quinn and C Gengnagel. 2014. A review of elastic grid shells, their erection methods and the potential use of pneumatic formwork. *Mob Rapidly Assem Struct IV* 136 (2014), 129–143.
Christian Schüller, Roi Poranne, and Olga Sorkine-Hornung. 2018. Shape Representation by Zippables. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 37, 4 (Aug. 2018).
Christian Schumacher, Steve Marschner, Markus Cross, and Bernhard Thomaszewski. 2018. Mechanical Characterization of Structured Sheet Materials. *ACM Trans. Graph.* 37, 4, Article 148 (July 2018), 15 pages. https://doi.org/10.1145/3197517.3201278
Jonathan Richard Shewchuk. 1996. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *WACG (Lecture Notes in Computer Science)*, Vol. 1148. Springer, 203–222.
Ole Sigmund. 2001. A 99 line topology optimization code written in Matlab. *Structural and multidisciplinary optimization* 21, 2 (2001), 120–127.
J. Spillmann and M. Teschner. 2007. CORDE: Cosserat Rod Elements for the Dynamic Simulation of One-Dimensional Elastic Objects. In *Eurographics/SIGGRAPH Symposium on Computer Animation*. https://doi.org/10.2312/SCA/SCA07/063-072
Tomohiro Tachi. 2013. Composite Rigid-Foldable Curved Origami Structure.
Chengcheng Tang, Xiang Sun, Alexandra Gomes, Johannes Wallner, and Helmut Pottmann. 2014. Form-finding with Polyhedral Meshes Made Simple. *ACM Trans. Graph.* 33, 4, Article 70 (July 2014), 9 pages. https://doi.org/10.1145/2601097.2601213
Frédéric Tayeb, Jean-François Caron, Olivier Baverel, and Lionel Du Peloux. 2013. Stability and robustness of a 300 m2 Composite Gridshell Structure. *Construction and Building Materials* (2013).
Bernhard Thomaszewski, Stelian Coros, Damien Gauge, Vittorio Megaro, Eitan Grinspun, and Markus Gross. 2014. Computational Design of Linkage-based Characters. *ACM Trans. Graph.* 33, 4, Article 64 (July 2014), 9 pages. https://doi.org/10.1145/2601097.2601143
Davide Tonelli, Nico Pietroni, Paolo Cignoni, and Roberto Scopigno. 2016. Design and Fabrication of Grid-shells Mockups. In *Proceedings of the Conference on Smart Tools and Applications in Computer Graphics (STAG '16)*. 21–27. https://doi.org/10.2312/stag.20161360
Nobuyuki Umetani, Ryan Schmidt, and Jos Stam. 2014. Position-based Elastic Rods. In *ACM SIGGRAPH 2014 Talks (SIGGRAPH '14)*. ACM, New York, NY, USA, Article 47, 1 pages. https://doi.org/10.1145/2614106.2614158
Jonas Zehnder, Stelian Coros, and Bernhard Thomaszewski. 2016. Designing Structurally-sound Ornamental Curve Networks. *ACM Trans. Graph.* 35, 4, Article 99 (July 2016), 10 pages. https://doi.org/10.1145/2897824.2925888
Changxi Zheng, Timothy Sun, and Xiang Chen. 2016. Deployable 3D Linkages with Collision Avoidance. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '16)*. 179–188. http://dl.acm.org/citation.cfm?id=2982818.2982843

A IMPLEMENTATION ASPECTS

In this section, we comment on additional implementation aspects of our numerical solvers for forward simulation, design optimization, and actuation sparsification presented above.

Elastic energy Hessians and their derivatives. The full expressions of (1) are provided in the supplemental material, which also gives a derivation of the gradients and Hessians and points out terms omitted in [Bergou et al. 2010] that are needed in certain settings. The Hessian-vector product formulas needed to run Newton-CG on the design optimization and actuation sparsification problems require computing directional derivatives of the elastic energy Hessian (i.e., third derivative information). We compute these efficiently using automatic differentiation of our hand-coded Hessian formulas.

3D Rotations. Our simulation algorithm for X-shells requires optimizing over the 3D rotation of each joint. We choose an approach that avoids singularities (gimbal lock) and leads to relatively simple and efficient formulas for the gradients and Hessians of the rotation matrix: the tangent space to $SO(3)$ at a particular reference rotation (i.e., the infinitesimal rotations). By updating this reference rotation to the current rotation at each Newton step (resetting the corresponding optimization variables to zero), singularities like gimbal lock are eliminated and the derivative formulas are simplified–only variations around the identity need to be computed, and many terms of the derivatives vanish. However, we still need the full derivative formulas to enable comparison against an off-the-shelf optimizer that does not support manually resetting the optimization variables at each iteration and to calculate third derivatives by automatic differentiation. We provide these full gradient and Hessian formulas in the supplemental material.

Numerically robust formulas. Several formulas used in our paper must be implemented with care. To prevent floating point overflows, the KS function used to aggregate the minimum angle constraints must actually be implemented in the following mathematically equivalent form [Martins and Poon 2005]:

$$KS(\alpha, s) = \min(\alpha) - s \log\left(\sum_i e^{\frac{\min(\alpha)-\alpha_i}{s}}\right).$$

For small magnitudes of the infinitesimal rotation variables—extremely common due to our strategy of resetting the rotation parametrization—several terms in the rotations' gradient and Hessian formulas degenerate to $\frac{0}{0}$. We provide robust formulas based on Taylor expansions of the problematic terms in the supplemental material.

Trust regions and iterate evaluation. In both the design optimization and actuation sparsification algorithms, evaluating the objective and constraints at a new set of design parameters (recomputing the closed and deployed equilibria) is efficient provided the change in parameters is not too great: only a few iterations of Newton's method are needed. However, large steps will require many Newton iterations in the inner equilibrium solves and, worse, often cause the elastic energy Hessian to become indefinite, further slowing the equilibrium solver. To keep the optimization fast we use Knitro's trust region optimizer [Artelys 2019]. We found that, in practice, after one or two iterations Knitro has determined a reasonable trust region radius that keeps subsequent iterations fast.

We can also exploit our existing elastic energy Hessian factorizations for the flat and deployed states to further improve the performance of our inner Newton solver—thereby enabling the outer Newton CG algorithm to take larger steps. Instead of simply applying the new design parameters/actuation forces to the linkage and rerunning the equilibrium solver initialized with the previous iterate's equilibrium configuration, we can use the prefactored Hessians to predict the equilibrium deformation under the new settings. Given a parameter step $\delta\mathbf{p}$ requested by the outer optimization, we construct a second order Taylor expansion of the

equilibrium displacements along this direction. Considering the deployed equilibrium as an example,

$$\mathbf{x}^*_{3\mathrm{D}}(\mathbf{p}+\delta\mathbf{p}) = \mathbf{x}^*_{3\mathrm{D}}(\mathbf{p}) + \frac{\partial \mathbf{x}^*_{3\mathrm{D}}}{\partial \mathbf{p}}\delta\mathbf{p} + \frac{1}{2}\delta\mathbf{p}^T\frac{\partial^2 \mathbf{x}^*_{3\mathrm{D}}}{\partial\mathbf{p}\partial\mathbf{p}}\delta\mathbf{p} + O(\delta\mathbf{p}^3).$$

In Section B, we compute the first order change $\frac{\partial \mathbf{x}^*_{3\mathrm{D}}}{\partial \mathbf{p}}\delta\mathbf{p}$ as the quantity $\delta\mathbf{x}^*_{3\mathrm{D}}$ and show how to find second order change $\delta\mathbf{p}^T\frac{\partial^2 \mathbf{x}^*_{3\mathrm{D}}}{\partial\mathbf{p}\partial\mathbf{p}}\delta\mathbf{p}$.

This strategy particularly benefits the design optimization, where it reduces the likelihood of the elastic energy Hessians in the inner equilibrium solves becoming indefinite: applying even mild changes to the design parameters while holding the equilibrium fixed often immediately puts the structure in an indefinite configuration. In principle, higher order or multiple-step continuation methods could also be applied [Allgower and Georg 2012; Chen et al. 2014] at the cost of additional computational and implementation complexities (e.g., computing fourth derivatives of elastic energy). For the purposes of accelerating the outer optimization, this simple second-order continuation method strikes a good balance of reducing the subsequent Newton corrector iterations with low overhead: just two additional backsolves for each configuration.

Sparse factorizations. We compute sparse Cholesky factorizations of the elastic energy Hessians H using Cholmod [Chen et al. 2008], which also informs us when H is indefinite so that we can apply our modification scheme to form $\tilde{H}$. Since the sparsity pattern of $\tilde{H}$ remains fixed across all equilibrium solves (and is identical for both the flat and deployed linkages), we perform a single symbolic factorization that we re-use throughout the entire design optimization. We update the numeric factorizations of the flat and deployed structure Hessians at the end of each Newton iteration (4). We finally use the factorizations computed at the end of the final iteration to compute the adjoint state variables (for the design problem's gradients), the directional derivatives of the equilibrium and adjoint state variables (for the Hessian-vector products), and the second directional derivatives of the equilibrium (for predicting the equilibria at the next design optimization iterate) with back substitution.

Constants. The number of linear elements k comprising each rod is set to 10 for all examples shown in the paper. This was chosen by comparing the accuracy of a discrete rod with k segments against an analytical solution for a planar elastica curve; 10 subdivisions gives less than 1% error in the bending forces over the range of bending magnitudes an X-shell might reasonably experience.

B GRADIENTS AND HESSIANS

We describe how to compute the derivative information required to apply Newton CG to the design optimization (5) proposed in Section 5. We present only the objective function's derivatives here, since the process for differentiating the constraints is analogous, but we provide the full set of formulas in the supplementary material.

For gradients, we solve for the adjoint deployed state vector $\mathbf{w}$:

$$\underbrace{\begin{bmatrix} H_{3\mathrm{D}} & \mathbf{a} \\ \mathbf{a}^T & 0 \end{bmatrix}}_{K_{3\mathrm{D}}} \begin{bmatrix} \mathbf{w} \\ w_\lambda \end{bmatrix} = \begin{bmatrix} W\left(\mathbf{x}^*_{3\mathrm{D}} - \mathbf{x}_{\mathrm{tgt}}\right) + W_{\mathrm{surf}}\left(\mathbf{x}^*_{3\mathrm{D}} - P_{\mathrm{surf}}\left(\mathbf{x}^*_{3\mathrm{D}}\right)\right) \\ 0 \end{bmatrix},$$

where $H_{3\mathrm{D}} = \frac{\partial^2 E}{\partial \mathbf{x}\partial\mathbf{x}}(\mathbf{x}^*_{3\mathrm{D}}(\mathbf{p}), \mathbf{p})$ is the elastic energy Hessian for the deployed X-shell equilibrium, and scalar w_λ is ignored. With this adjoint state, we can efficiently evaluate the objective's gradient,

$$\frac{\partial J}{\partial \mathbf{p}} = \frac{\gamma}{E_0}\frac{\partial E}{\partial \mathbf{p}}(\mathbf{x}^*_{2\mathrm{D}}, \mathbf{p}) + \frac{(1-\gamma)}{E_0}\frac{\partial E}{\partial \mathbf{p}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p}) - \frac{\beta}{l_0^2}\mathbf{w}^T\frac{\partial^2 E}{\partial \mathbf{x}\partial\mathbf{p}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p}).$$

To compute Hessian-vector products, we compute the variations of forward and adjoint state vectors $\delta\mathbf{x}^*_{2\mathrm{D}}$, $\delta\mathbf{x}^*_{3\mathrm{D}}$, and $\delta\mathbf{w}$ that arise from parameter perturbation $\delta\mathbf{p}$ by solving:

$$K_{3\mathrm{D}}\begin{bmatrix}\delta\mathbf{x}^*_{3\mathrm{D}} \\ \delta\lambda_{3\mathrm{D}}\end{bmatrix} = \begin{bmatrix} -\frac{\partial^2 E}{\partial\mathbf{x}\partial\mathbf{p}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p})\delta\mathbf{p} \\ 0 \end{bmatrix},$$

$$K_{3\mathrm{D}}\begin{bmatrix}\delta\mathbf{w} \\ \delta w_\lambda\end{bmatrix} = \begin{bmatrix} W\delta\mathbf{x}^*_{3\mathrm{D}} + W_{\mathrm{surf}}\left(\delta\mathbf{x}^*_{3\mathrm{D}} - \frac{\partial P_{\mathrm{surf}}}{\partial\mathbf{x}}\delta\mathbf{x}^*_{3\mathrm{D}}\right) \\ 0 \end{bmatrix} - \begin{bmatrix} \left(\frac{\partial^3 E}{\partial\mathbf{x}\partial\mathbf{x}\partial\mathbf{x}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p})\delta\mathbf{x}^*_{3\mathrm{D}} + \frac{\partial^3 E}{\partial\mathbf{x}\partial\mathbf{x}\partial\mathbf{p}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p})\delta\mathbf{p}\right)\mathbf{w} \\ 0 \end{bmatrix},$$

where the derivative of the closest point projection $\frac{\partial P_{\mathrm{surf}}}{\partial\mathbf{x}}$ can be computed as $I - \mathbf{n}\otimes\mathbf{n}$ when the closest point lies on a surface triangle with normal $\mathbf{n}$, $\mathbf{e}\otimes\mathbf{e}$ when it lies on a surface edge with normalized edge vector $\mathbf{e}$, and 0 when it lies on a vertex. The formula for $\delta\mathbf{x}^*_{2\mathrm{D}}$ is analogous to $\delta\mathbf{x}^*_{3\mathrm{D}}$ and is given in the supplement.

The Hessian-vector products for J can now be calculated:

$$\begin{aligned}\frac{\partial^2 J}{\partial\mathbf{p}\partial\mathbf{p}}\delta\mathbf{p} =& \frac{\gamma}{E_0}\left(\frac{\partial^2 E}{\partial\mathbf{p}\partial\mathbf{x}}(\mathbf{x}^*_{2\mathrm{D}}, \mathbf{p})\delta\mathbf{x}^*_{2\mathrm{D}} + \frac{\partial^2 E}{\partial\mathbf{p}\partial\mathbf{p}}(\mathbf{x}^*_{2\mathrm{D}}, \mathbf{p})\delta\mathbf{p}\right) \\ &+ \frac{(1-\gamma)}{E_0}\left(\frac{\partial^2 E}{\partial\mathbf{p}\partial\mathbf{x}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p})\delta\mathbf{x}^*_{3\mathrm{D}} + \frac{\partial^2 E}{\partial\mathbf{p}\partial\mathbf{p}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p})\delta\mathbf{p}\right) \\ &- \frac{\beta}{l_0^2}\begin{bmatrix}\frac{\partial^2 E}{\partial\mathbf{p}\partial\mathbf{x}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p}) \\ 0\end{bmatrix}\delta\mathbf{w} \\ &- \frac{\beta}{l_0^2}\begin{bmatrix}\frac{\partial^3 E}{\partial\mathbf{p}\partial\mathbf{x}\partial\mathbf{x}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p})\delta\mathbf{x}^*_{3\mathrm{D}} + \frac{\partial^3 E}{\partial\mathbf{p}\partial\mathbf{x}\partial\mathbf{p}}(\mathbf{x}^*_{3\mathrm{D}}, \mathbf{p})\delta\mathbf{p} \\ 0\end{bmatrix}\mathbf{w}.\end{aligned}$$

Notice that we can reuse the factorization of $K_{3\mathrm{D}}$ that was computed to solve the adjoint equation, so the added cost for computing a Hessian-vector product for the objective is simply three additional back substitutions for $\delta\mathbf{x}^*_{2\mathrm{D}}$, $\delta\mathbf{x}^*_{3\mathrm{D}}$, and $\delta\mathbf{w}$. While these formulas involve third derivatives of the elastic energy, really only the *directional derivative* of the elastic energy Hessian is needed. Since we have efficient analytic Hessian expressions for our forward simulation, these directional derivatives can be calculated inexpensively with automatic differentiation (differentiating a function that implements the elastic energy Hessian-vector product).

Finally, the second order change in the deployed equilibrium induced by perturbing the design parameters by $\delta\mathbf{p}$ is:

$$K_{3\mathrm{D}}\begin{bmatrix}\delta\mathbf{p}^T\frac{\partial^2\mathbf{x}^*_{3\mathrm{D}}}{\partial\mathbf{p}\partial\mathbf{p}}\delta\mathbf{p} \\ \delta\mathbf{p}^T\frac{\partial^2\lambda_{3\mathrm{D}}}{\partial\mathbf{p}\partial\mathbf{p}}\delta\mathbf{p}\end{bmatrix} = -\begin{bmatrix}\left(\frac{\partial^3 E}{\partial\mathbf{x}\partial\mathbf{x}\partial\mathbf{x}}\delta\mathbf{x}^*_{3\mathrm{D}} + \frac{\partial^3 E}{\partial\mathbf{x}\partial\mathbf{x}\partial\mathbf{p}}\delta\mathbf{p}\right)\delta\mathbf{x}^*_{3\mathrm{D}} \\ 0\end{bmatrix} - \begin{bmatrix}\left(\frac{\partial^3 E}{\partial\mathbf{x}\partial\mathbf{p}\partial\mathbf{x}}\delta\mathbf{x}^*_{3\mathrm{D}} + \frac{\partial^3 E}{\partial\mathbf{x}\partial\mathbf{p}\partial\mathbf{p}}\delta\mathbf{p}\right)\delta\mathbf{p} \\ 0\end{bmatrix},$$

and analogously for $\mathbf{x}^*_{2\mathrm{D}}$.