CurveUps:
Shaping Objects from Flat Plates with Tension-Actuated Curvature

RUSLAN GUSEINOV, EDER MIGUEL, and BERND BICKEL, IST Austria

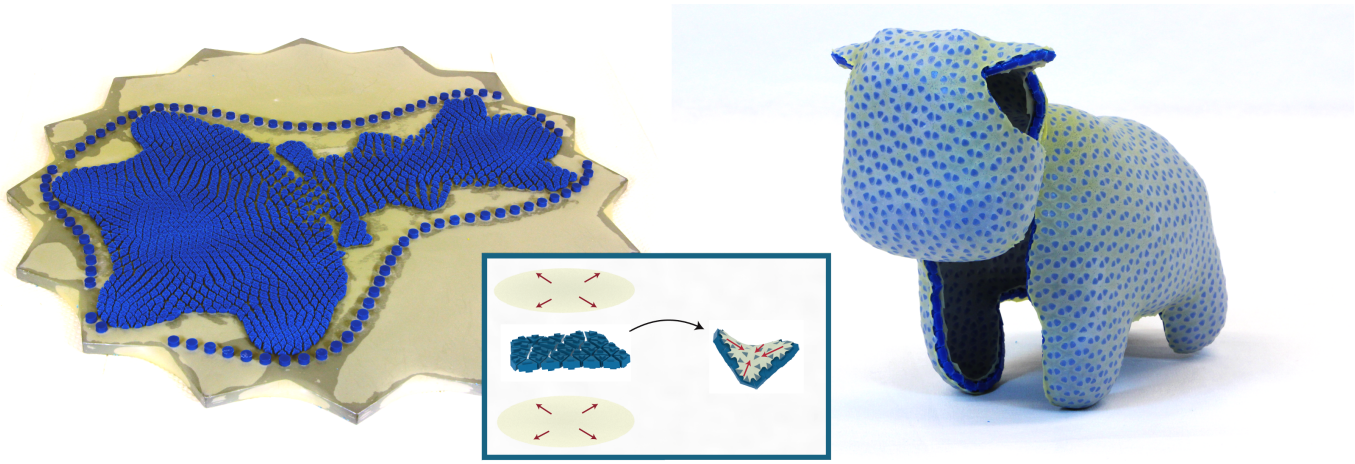


Fig. 1. CurveUps are printed flat and actuated into doubly curved 3D shapes using pre-stretched elastic membranes.

We present a computational approach for designing CurveUps, curvy shells that form from an initially flat state. They consist of small rigid tiles that are tightly held together by two pre-stretched elastic sheets attached to them. Our method allows the realization of smooth, doubly curved surfaces that can be fabricated as a flat piece. Once released, the restoring forces of the pre-stretched sheets support the object to take shape in 3D. CurveUps are structurally stable in their target configuration. The design process starts with a target surface. Our method generates a tile layout in 2D and optimizes the distribution, shape, and attachment areas of the tiles to obtain a configuration that is fabricable and in which the curved up state closely matches the target. Our approach is based on an efficient approximate model and a local optimization strategy for an otherwise intractable nonlinear optimization problem. We demonstrate the effectiveness of our approach for a wide range of shapes, all realized as physical prototypes.

CCS Concepts: • **Computing methodologies** → **Shape modeling**; • **Applied computing** → Computer-aided design;

ACM Reference format:
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.
DOI: http://dx.doi.org/10.1145/3072959.3073709

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 645599 - Soft-bodied intelligence for Manipulation (SoMa) and from the European Research Council (ERC) under grant agreement No 715767 - MATERIALIZABLE: Intelligent fabrication-oriented Computational Design and Modeling.

1 INTRODUCTION

The potential of self-transforming structures with complex geometries has inspired numerous researchers from science, engineering, and art to explore novel materials, multi-material fabrication techniques, and material programmability as enabling technologies. The underlying transformation mechanism is usually the result of a complex interaction between materials exerting forces, anisotropic material distribution, and mechanical stability. Designing such structures is an active research challenge in computational fabrication, as it requires finding a physically realizable configuration that satisfies functional constraints while facing a huge number of degrees of freedom.

We explore a type of structure that starts from a flat initial configuration and has a static equilibrium that resembles a desired three-dimensional shape. We propose to use flat pre-stretched elastic sheets as the actuating material, combined with an anisotropic distribution of *disconnected* rigid tiles that resemble the geometric shape of frustums. The tiles, attached to these elastic sheets, embody the implementation of a transforming mechanism. This transforming mechanism is ideal for building robust, cost-efficient models while enabling the reproduction of a wide range of shapes. As the initial configuration is flat, we can use widely available standard materials with excellent deformation properties, such as latex, as base materials in the fabrication process. By adjusting the distribution and shape of the tiles, the resulting local curvature can be influenced and thereby the resulting global shape controlled. The tiles are rigidly attached to the pre-stretched sheets, which exert contracting forces on the tiles once released. These forces can be adjusted by the layout of the tiles as well as by the parameters that control the attachment location. This construction enables shaping

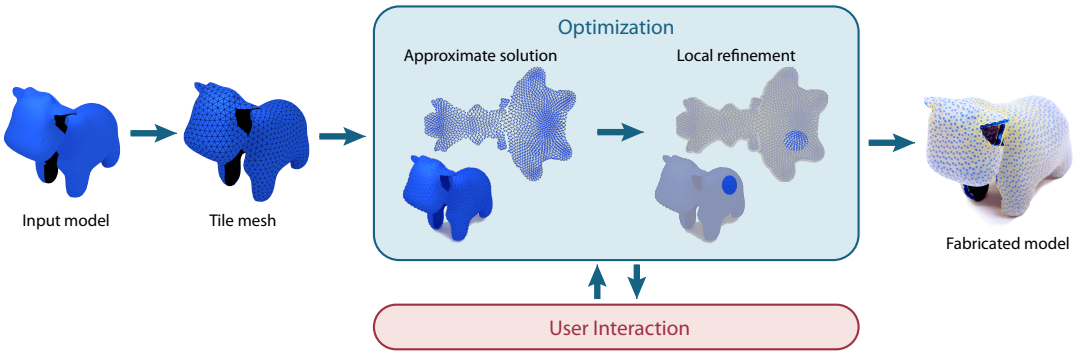


Fig. 2. Overview of our workflow: the user provides a target mesh, the system builds the initial tile layout and finds an approximate configuration of tiles while allowing the user to make cuts in the 2D layout. The approximate solution is then refined locally using the physical model. Finally, the structure is fabricated as a flat piece that is structurally stable in its actuated configuration.

nearly smooth surfaces without developability restrictions. As a result, a complex self-transforming structure can be fabricated with simple rapid prototyping technologies.

In this paper, we introduce a computational approach for designing shapes from flat plates with tension-actuated curvature. Our method starts with a desired input shape and automatically computes a fabricable configuration that best approximates the shape. As a basic building block, we propose a novel type of element, which has the shape of a frustum and is connected via pins to two uniformly stretched elastic sheets. At the core of our method is an optimization approach that solves a complicated layout and shape optimization problem: how do we best place the elements in a 2D layout, what is the optimal shape of these elements, and how do we connect them to the elastic sheet to obtain the required forces to reach a physically valid and stable state that closely approximates the desired input shape? Due to the nonlinear relationship of the element configurations to the resulting forces in the target configuration, this is a very challenging inverse problem, highly dependent on the initial guess. Motivated by simple construction rules observed in practice, we introduce a highly effective approximation to compute an initial guess. We then propose a physics-based model and employ a dedicated optimization method for computing the configuration of the elements.

We verify our approach by both simulating the resulting structures and fabricating several example models, ranging from simple shapes including parts of an ellipsoid and of a hyperbolic paraboloid to design studies of popular models from architecture and computer graphics.

2 RELATED WORK

Self-Actuating Materials. Previous works have developed different self-actuating materials and applied them to generate 3D shapes. Based on unidimensional components with discrete elements encoding the prescribed behavior, Raviv et al. [2014] produce grids that deform from an initially planar configuration to a curved surface using hydrophilic materials for actuation. Gladman et al. [2016] generalize this approach, encoding information in a continuous domain. They generate biologically inspired deforming structures using unidimensional composite hydrogel elements that swell anisotropically when immersed in water and interact in complex ways when arranged in networks. Similarly, Kim et al. [2012] use photopatterning polymer films to create temperature-responsive gel sheets that can transform from a flat configuration to a prescribed 3D shape. Moving into 2D domains, methods based on self-folding origami [An et al. 2014; Hawkes et al. 2010; Kwok et al. 2015; Tolley et al. 2014] are able to exploit the information encoded in the crease patterns to create self-folding sheets. While the design space of these approaches is usually constrained to developable surfaces, hinge-based designs, or near constant Gaussian curvature designs, our approach uses an initially flat layer of disconnected rigid tiles, which allows us to avoid hinges and go beyond developable surfaces.

A key element in self-actuating materials is the actuation mechanism. Its goal is to provide the energy necessary to drive the prescribed change in properties. Previous works have explored a wide variety of actuation mechanisms, from temperature-driven Shape Memory Alloys (SMA) [Felton et al. 2013; Hawkes et al. 2010] and gel sheets [Kim et al. 2012] to hydrophilic materials that change properties when immersed in water [Gladman et al. 2016; Raviv et al. 2014], including pneumatic actuation [Shepherd et al. 2011] or light-responsive materials [Ryu et al. 2012].

Here, we present a novel actuation mechanism based on elastic energy stored in pre-stretched membranes. When relaxed, these membranes bring the rigid tiles together, driving the shape change. While our approach is based on a uniformly stretched elastic sheet and does not rely on external stimuli, our approach might theoretically be extensible with some of the previously mentioned actuating materials.

Fabrication-Aware Surface Design. The field of fabrication-aware surface design has received considerable attention in recent years. Paper folding is one of the most common techniques for generating surfaces. Previous works in this area have studied origami patterns [Demaine and O'Rourke 2007; Dudte et al. 2016; Massarwi et al. 2007] and folding based on curved creases [Kilian et al. 2008; Mitani and Igarashi 2011]. Recently, Tang et al. [2016] added interactivity

with their design tool for exploration of the space of developable surfaces. Our method builds on research in this area to compute unfolded designs of 3D models, but it explores the design space of non-developable surfaces.

Many other methods and tools have been developed that leverage a wide variety of material and structural properties to design fabricable surfaces. Skouras et al. use stretching [2012] and bending [2014] of thin sheets to create inflatable structures with complex shapes. Similarly, Garg et al. [2014] exploit the shearing ability of regular wire grids to design aesthetic wire mesh sculptures. More recently, Konaković et al. [2016] present a computational method for surface design that introduces a cutting pattern analogous to a triangular linkage to allow spatially varying stretching of originally inextensible flat sheets. These methods employ computational optimization strategies to find nontrivial configurations that approximate doubly curved surfaces. Ou et al. [2016] introduce a universal bending mechanism that creates programmable, air-pressure-activated shape changes from a flat sheet with custom-shaped air pouches. Their design approach supports user-adjusted geometries of air pouches and a forward simulation of its transformation. Our approach shares similar goals but introduces the concept of self-actuation. We neither require an additional form to shape a plastic material nor any external stimuli such as air pressure. Furthermore, we explicitly solve an inverse problem to find a configuration that is physically stable.

Tile-Based Design. In a broader scope, the exploration of tile and element-based design spaces has also been thoroughly studied. Examples include surface design using interlocking 3D [Cignoni et al. 2014; Skouras et al. 2015] and planar [Hildebrand et al. 2012; Schwartzburg and Pauly 2013] elements, beadwork-based designs [Igarashi et al. 2012], self-supporting masonry structures [Deuss et al. 2014; Vouga et al. 2012], or panel-based surfaces [Eigensatz et al. 2010; Pottmann et al. 2008]. Often, fabrication of these structures requires finding an appropriate assembly sequence to produce the final surface. In contrast, our method relies on the actuation mechanism to automatically perform the assembly, thus simplifying both the design and fabrication processes.

3 OVERVIEW

Our goal is to approximate an input 3D model from an initially *flat* tile-based assembly that can be shaped into a *stable 3D configuration* without the need for any connectors or manual assembly. Our structures are made of flat, rigid tiles, which have a frustum-like shape. When aligned in 3D, they form a shell where all faces between neighboring elements are in contact. The individual elements are only connected via two thin elastic sheets, which represent the front and back covers of the shell. The elastic sheets are pre-stretched in the initial flat configuration and, once released, exert contraction forces between the elements. Finding an optimal design for such a flat assembly is very challenging, as the shape in 3D has to be stable under the contracting forces: the forces resulting from the contraction of the elastic sheet must balance the contact forces of the elements. To solve this problem efficiently, we propose a two-step optimization procedure. Fig. 2 illustrates the workflow. First, we compute an initial flat configuration and its corresponding 3D shape. For this, we uniformly remesh the input surface into triangles with the target edge length L that is well suited for our output device. Then, we thicken the triangles, which provides the initial shape of our elements, and pack them into a 2D layout. We use an approximate model that represents resulting forces between elements as linear springs, with empirically determined rules for a favorable 2D layout formulated as soft constraints and hard constraints that guarantee fabricability. This model also takes the shape and placement of connectors into account, which greatly influences the resulting forces. As shown in our results section, this model provides a high-quality approximation, which in most cases is already ready for fabrication. However, to guarantee physical validity, we switch to a more accurate finite element model with a Neo-Hookean material model for the elastic forces and compute a configuration that simultaneously optimizes the 2D layout, the shape of the individual elements, and the connectors to the elastic sheet. Finally, the obtained tile distribution is ready for fabrication and assembly.

4 MODEL

In this section, we formally introduce our model and design parameters. We start by introducing the geometry of the tiles. Then, we describe the representation used for the flat and actuated configurations. Finally, we explain the force balance as an essential feature of the actuated configuration.

Tiles and Pins. Each tile consists of its body and the pins attached to it. The body is a *frustum* defined by a pair of triangles with pairwise parallel edges representing its front and back sides. Thus, the body can be either a prism or a truncated pyramid. The distance between the triangles is the tile body thickness H_{body}. Each tile body triangle has a pin attached to it, which is a right triangular prism of height H_{pin} with the base edges parallel to the tile triangle edges pairwise. In this construction, we parametrize each pin by the pairwise distances from its edges to the edges of the corresponding tile triangle. Hence, for each tile, there are six parameters, which are denoted as

$$\mathbf{q} = q_{it}^{s},$$

where $s \in \{\mathrm{f}, \mathrm{b}\}$ denotes the front and back sides of tiles, $i \in [1, M]$ denotes tiles, M is the total number of tiles, and $t \in \{1, 2, 3\}$ denotes the edges of the tiles' triangular faces. A detailed illustration can be seen in Fig. 3.

Flat Configuration. In the flat configuration, the layout of triangular tiles is represented by the set of maps from the actuated to the flat configuration for each tile i:

$$\omega_i : \hat{\mathbf{v}}_{it}^{\mathrm{f}} \mapsto \bar{\mathbf{v}}_{it}^{\mathrm{f}},$$

where $\hat{\mathbf{v}}_{it}^{\mathrm{f}} \in \mathbb{R}^3$ and $\bar{\mathbf{v}}_{it}^{\mathrm{f}} \in \mathbb{R}^2$ are vertex coordinates of tile front faces in the actuated and flat configurations, respectively. In order to avoid deformation of the faces by the given mapping, we constrain the edge lengths of the triangles so that, effectively, each tile remains rigid and can be represented by only three degrees of freedom.

Actuated Configuration. Tiles in the actuated configuration are represented by a triangle mesh $\mathcal{M}$, containing M faces. Each face defines the front face of a tile, while the opposite face is defined solely by the tile thickness H_{body} and dihedral angles formed with

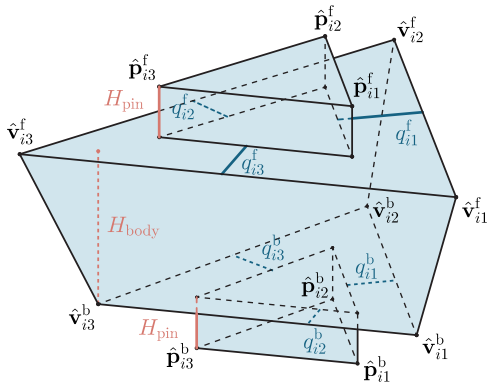


Fig. 3. One tile with notation.

neighboring faces. This parameterization implies that the front faces of the tiles should perfectly match in the actuated configuration. We denote N as the number of vertices in $\mathcal{M}$ with coordinates $\mathbf{x}_j$ for $j \in [1, N]$. Note that following this construction might produce small intersections between tiles in the actuated configuration. We handle this problem by removing intersecting parts in a post-process prior to fabrication.

In summary, as illustrated in Fig. 4, the design space of our model is defined by the following:

- the vertices in the actuated configuration, $\mathbf{x} \in \mathbb{R}^{3N}$,
- the maps of faces of $\mathcal{M}$ from the actuated to flat configuration, ω_i, effectively in $\mathbb{R}^{3M}$,
- the pin parameters, $\mathbf{q} \in \mathbb{R}^{6M}$,

which gives a total number of degrees of freedom of $3N + 9M$.

During actuation, the pre-stretched elastic membranes push the tiles together. Eventually, tiles collide, producing contact forces that balance out the elastic forces. We use this equilibrium condition to compute the contact forces as a least squares problem:

$$\begin{aligned} &\text{minimize} && \sum \|\mathbf{f}^{\mathrm{c}}\|^2 \\ &\text{subject to} && \text{static equilibrium,} \end{aligned} \tag{1}$$

where static equilibrium is expressed as

$$\begin{aligned} \mathbf{f}^{\mathrm{c}}_i + \mathbf{f}^{\mathrm{e}}_i &= 0, \\ \mathbf{t}^{\mathrm{c}}_i + \mathbf{t}^{\mathrm{e}}_i &= 0, \end{aligned} \tag{2}$$

for each i-th tile, $\mathbf{f}^{\mathrm{c}}_i$ and $\mathbf{f}^{\mathrm{e}}_i$ represent the total contact and elastic forces, respectively, and $\mathbf{t}^{\mathrm{c}}_i$ and $\mathbf{t}^{\mathrm{e}}_i$ represent the total torques with respect to any given point generated by the contact and elastic forces, respectively. Since the choice of the point is arbitrary, in practice, we compute the torques relative to a preselected front face vertex for each tile. Given a point $\mathbf{r}_i$, the torques are computed as $\mathbf{t}^{\mathrm{c}}_i = \mathbf{r}_i \times \mathbf{f}^{\mathrm{c}}_i$ and $\mathbf{t}^{\mathrm{e}}_i = \mathbf{r}_i \times \mathbf{f}^{\mathrm{e}}_i$.

We compute the elastic forces per pin vertex by solving a quasi-static problem on the thin sheet, as described in Appendix A.1. The contact forces are computed for each of the four contact points of each contact side of the tile (see Appendix A.2 for computation of the contact points). We enforce Newton's third law for each pair of contact forces applied to the same contact point shared between two neighboring tiles.

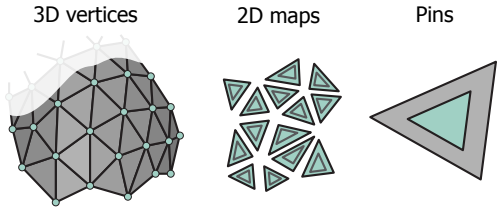


Fig. 4. Degrees of freedom highlighted with green color (from left to right): vertex coordinates $\mathbf{x}$ of the triangle mesh $\mathcal{M}$, maps of triangles from the actuated to flat configuration ω_i, and pin parameters $\mathbf{q}$.

Note that while the front faces of the tiles always match perfectly, the back faces might need to be cut in order to avoid self-intersections with neighboring tiles (see Fig. 14). Thus, we cut each tile at each back face vertex with a plane so that the tiles no longer intersect each other (see Appendix A.2).

While this model ensures physically valid contact forces, they might be not strong enough or inverted (meaning that they are pulling tiles apart instead of pushing them together), producing an invalid design. Thus, in order to find a set of model parameters that ensures a valid design, we formulate a constrained optimization problem, which is described in the next section.

5 OPTIMIZATION

The goal of our system is to approximate an input 3D model with the surface generated by the tiles in the actuated configuration. We formulate this goal as a constrained nonlinear optimization problem, where soft constraints represent desired properties of the final assembly while hard constraints are used to capture fabrication and actuation limitations:

$$\begin{aligned} &\underset{\mathbf{P}}{\text{minimize}} && E_{\text{target}} + w_{\text{tight}} E_{\text{tight}} \\ &\text{subject to} && g_a(\mathbf{P}) \le 0, h_b(\mathbf{P}) = 0, \end{aligned} \tag{3}$$

where E_{target} measures the deviation of the actuated mesh $\mathcal{M}$ from the target mesh $\mathcal{T}$, E_{tight} represents the tightness of bonds between tiles, $g_a(\mathbf{P})$ and $h_b(\mathbf{P})$ are the hard inequality and equality constraints, respectively, and $\mathbf{P}$ is a concatenation of all model parameters $\mathbf{x}_j$, $\bar{\mathbf{v}}^{\mathrm{f}}_{it}$, q^s_{it}. In order to allow the user to control the behavior of the optimization, we introduce a weight, w_{tight}, which defines the tradeoff between structural tightness and deviation from the target shape.

Target Energy. Our goal is to obtain a shape that closely resembles the target shape. The target energy measures the deviation of the surfaces and is measured as an area-weighted vertex to vertex distance:

$$E_{\text{target}}(\mathbf{x}) = \sum_j A_j \|\mathbf{x}_j - \mathbf{c}_j\|^2, \tag{4}$$

where $\mathbf{c}_j$ denotes the vertex coordinates of $\mathcal{T}$ and A_j is the sum of the areas of all triangles of $\mathcal{T}$ sharing the vertex j.

Tightness Energy. The tiles should tightly press against each other in the final configuration. The tightness energy supports a minimum normal contact force magnitude. Assuming infinite friction between tiles, we measure this structural tightness as the sum of the magnitudes of the normal contact forces smaller than $f_{\min}$ for each pair of contact tiles in the actuated mesh, $\mathcal{M}$:

$$E_{\text{tight}}(\mathbf{P}) = \sum_{k} \sum_{l=1}^{4} \left(\max\left(0, f_{\min} - \mathbf{f}^{\text{c}}_{kl} \cdot \mathbf{n}^{\text{c}}_{kl}\right)\right)^2, \tag{5}$$

where $\mathbf{n}^{\text{c}}_{kl}$ and $\mathbf{f}^{\text{c}}_{kl}$ are the l-th contact normal and contact force, respectively, of the k-th edge and $f_{\min}$ is the minimal required normal force magnitude.

Fabrication and Actuation Limitations. These are expressed as hard constraints. Specifically, we set lower and upper bounds on the pin parameters to ensure proper actuation:

$$p_{\min} \leq p^s_{it} \leq p_{\max},$$

where $p_{\min}$ and $p_{\max}$ are constant. The minimal value constrains the pins to stay within the limits of the base triangle they are attached to, while the maximal value prevents the pins from being too far away from the contacts between tiles.

We also set a lower bound $A^{\text{pin}}_{\min}$ on the pin areas so that they can be reliably attached to the actuating elastic membranes:

$$g^{\text{pin area}}_{si} = 2A^{\text{pin}}_{\min} - \left\| (\bar{\mathbf{p}}^s_{i2} - \bar{\mathbf{p}}^s_{i0}) \wedge (\bar{\mathbf{p}}^s_{i1} - \bar{\mathbf{p}}^s_{i0}) \right\| \leq 0, \tag{6}$$

where we use the wedge product notation $\wedge$ as a "two-dimensional analog" of the cross product.

Additionally, for each pair of tiles in contact, we introduce constraints for collisions, taking into account fabrication limits on the minimal distance between disconnected printed objects:

$$g^{\text{gap}}_{skl} = d^{\text{gap}}_{\min} - \frac{(\bar{\mathbf{v}}^s_{k12} - \bar{\mathbf{v}}^s_{k11}) \wedge (\bar{\mathbf{v}}^s_{k2l} - \bar{\mathbf{v}}^s_{k11})}{\|\bar{\mathbf{v}}^s_{k12} - \bar{\mathbf{v}}^s_{k11}\|} \leq 0, \quad l \in \{1,2\}, \tag{7}$$

where the indexing is shown in Fig. 5.

Since the maps ω_i defined by the variables $\hat{\mathbf{v}}^{\text{f}}_{it}$ and $\bar{\mathbf{v}}^{\text{f}}_{it}$ might deform the shapes of the tiles, we constrain the rigidity of the tiles as follows:

$$h^{\text{rig}}_{it} = \|\hat{\mathbf{v}}^{\text{f}}_{it} - \hat{\mathbf{v}}^{\text{f}}_{it^{+1}}\| - \|\bar{\mathbf{v}}^{\text{f}}_{it} - \bar{\mathbf{v}}^{\text{f}}_{it^{+1}}\| = 0, \tag{8}$$

where $t^{+1} = (t \mod 3) + 1$.

Solving the optimization problem in Eq. 3 requires the evaluation of the contact forces, which in turn requires solving a quasi-static problem to find the equilibrium configuration of the elastic membranes, as shown in Eq. 1. This operation is computationally expensive and unaffordable for any moderate-sized problem in an interactive tool. Hence, in order to efficiently solve this problem, we propose a two-step optimization approach, where we first solve a coarse approximate optimization problem and then locally refine the obtained solution.

5.1 Coarse Optimization

We have analyzed the behavior and properties of desirable solutions and synthesized physics-based energy terms that capture them. Then, we have used these energy terms to construct an approximation model that requires no expensive operation to be evaluated

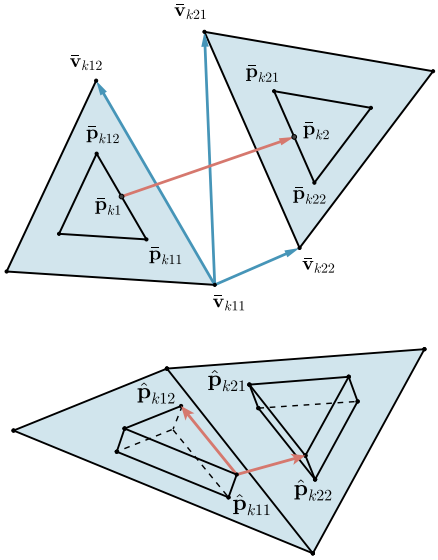


Fig. 5. Two contact tiles in the flat (top) and actuated (bottom) configurations.

but closely represents the original optimization landscape. This allows us to find an approximate solution to the original optimization problem in a cost-efficient way.

The new optimization problem is formulated as:

$$\begin{aligned} \underset{\bar{\mathbf{v}},\mathbf{p}}{\text{minimize}} \quad E_{\text{flat}} = \quad & w_{\text{r}}E_{\text{r}} + w_{\text{c}}E_{\text{c}} + w_{\text{pa}}E_{\text{pa}} + w_{\text{a}}E_{\text{a}} + \\ & + w_{\text{p}}E_{\text{p}} + w_{\text{i}}E_{\text{i}} + w_{\text{s}}E_{\text{s}}, \end{aligned} \tag{9}$$

where each term in the cost function E_{flat} contributes to obtaining desirable properties in the actuation, fabrication, or assembly processes.

Tile Rigidity. In order to minimize the deformation of individual tiles with respect to the remeshed model, we introduce a tile rigidity term, E_{r}, that measures the deformation of tile faces:

$$E_{\text{r}} = L^{-2} \sum_{i,t} \left(h^{\text{rig}}_{it}\right)^2, \tag{10}$$

where h^{rig}_{it} is the tile rigidity constraint function defined in Eq. 8 and L^{-2} is a normalizer based on the edge length L to make the energy term dimensionless.

Collision Avoidance. In order to avoid collisions between neighboring tiles in the flat layout, the term E_{c} is added:

$$E_{\text{c}} = L^{-2} \sum_{k,s,l} \left(\max\left(0, g^{\text{gap}}_{ksl}\right)\right)^2, \tag{11}$$

where g^{gap}_{ksl} is the inequality constraint defined in Eq. 7.

Pin Area. We penalize pin areas below the defined minimum:

$$E_{\text{pa}} = L^{-4} \sum_{s,i} \left(\max\left(0, g^{\text{pin area}}_{si}\right)\right)^2, \tag{12}$$

where $g^{\text{pin area}}_{si}$ is the inequality constraint defined in Eq. 6.

Tile Alignment. We aim to place the tiles in a flat configuration such that the resulting elastic forces are aligned as much as possible with the contact surface normals. This decreases the friction contact forces, thus significantly improving the stability of the structure. As illustrated in Fig. 6, we introduce an alignment term, E_a, that penalizes tile misalignments:

$$E_\mathrm{a} = L^{-2} \sum_k \left(\|\bar{\mathbf{v}}^\mathrm{f}_{k11} - \bar{\mathbf{v}}^\mathrm{f}_{k21}\| - \|\bar{\mathbf{v}}^\mathrm{f}_{k12} - \bar{\mathbf{v}}^\mathrm{f}_{k22}\| \right)^2. \tag{13}$$

The effectiveness of this term is illustrated in Fig. 7; it reduces the relative friction forces on average from 18% to 11% in the *Half-Sphere* model.

Tile Packing. In order to maximize the normal contact forces and decrease the size of the flat layout, we introduce a tile packing term, E_p, that helps pack the tiles in the flat layout as closely as possible:

$$E_\mathrm{p} = L^{-4} \sum_k \|\bar{\mathbf{v}}^\mathrm{f}_{k11} - \bar{\mathbf{v}}^\mathrm{f}_{k22}\|^4 + \|\bar{\mathbf{v}}^\mathrm{f}_{k12} - \bar{\mathbf{v}}^\mathrm{f}_{k21}\|^4. \tag{14}$$

We use the fourth order to severely penalize large distances between the contact tiles. This term also helps to reduce the computational cost of the elastic membrane simulation, which is determined by the number of degrees of freedom. For a given discretization resolution, this depends directly on the free area (not attached to any pin) in the elastic membranes. This term minimizes the free areas and the domains between tiles, therefore reducing the number of simulation degrees of freedom. As a side effect, it also allows us to prevent buckling of the membranes, which tends to produce invalid contact forces.

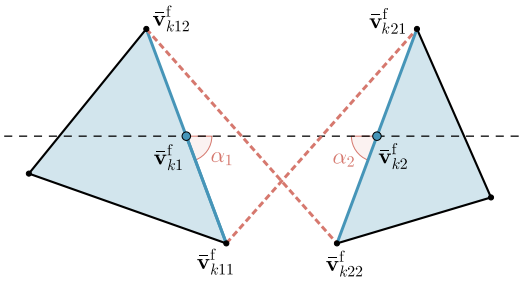


Fig. 6. The alignment energy term aims to equalize the angles α_1 and α_2 between the edges of the tiles and the dashed line, which connects their centers $\bar{\mathbf{v}}^\mathrm{f}_{k1}$ and $\bar{\mathbf{v}}^\mathrm{f}_{k2}$. This is achieved by minimizing the difference between the lengths of the diagonals (red dashed segments).

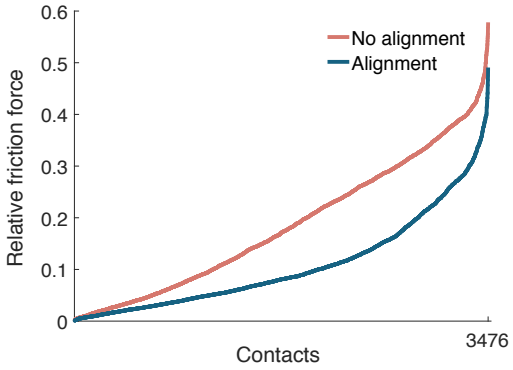


Fig. 7. Relative friction force magnitude per contact in ascending order with and without the alignment energy term for *Half-Sphere*.

Tile Inversion. In order to prevent potential inversion of tile faces when transformed by the mappings ω_i, we introduce the following energy to penalize wrong orientation:

$$E_\mathrm{i} = L^{-4} \sum_i \left(\min\left(0, (\bar{\mathbf{v}}^\mathrm{f}_{i2} - \bar{\mathbf{v}}^\mathrm{f}_{i1}) \wedge (\bar{\mathbf{v}}^\mathrm{f}_{i3} - \bar{\mathbf{v}}^\mathrm{f}_{i1})\right)\right)^2. \tag{15}$$

Spring-Like Approximation. Our goal is to find an equilibrium configuration with no pulling contact forces and elastic forces evenly distributed over the structure. This requires the evaluation of elastic forces in the actuated configuration, which is computationally expensive due to the required global physical simulation using nonlinear membrane deformation models. We have developed a spring-like approximation, which significantly reduces the computational cost and allows us to obtain the initial pin parameters that fulfill this goal.

We define the spring-like energy as:

$$E_\mathrm{s} = \sum_{k,s} \left(\frac{\|\bar{\mathbf{p}}^s_{k2} - \bar{\mathbf{p}}^s_{k1}\| \|\hat{\mathbf{p}}^s_{k12} - \hat{\mathbf{p}}^s_{k11}\|}{\tau \|(\hat{\mathbf{p}}^s_{k12} - \hat{\mathbf{p}}^s_{k11}) \times (\hat{\mathbf{p}}^s_{k22} - \hat{\mathbf{p}}^s_{k11})\|} \right)^2, \tag{16}$$

with τ being the uniform stretch ratio, $\bar{\mathbf{p}}^s_{k1} = \frac{1}{2}(\bar{\mathbf{p}}^s_{k11} + \bar{\mathbf{p}}^s_{k12})$ and $\bar{\mathbf{p}}^s_{k2} = \frac{1}{2}(\bar{\mathbf{p}}^s_{k21} + \bar{\mathbf{p}}^s_{k22})$. This energy term is the sum of squares of ratios A/B, where A is the distance between the pin centers in the flat configuration scaled by τ^{-1} and B is the distance between the pins in the actuated configuration. Minimizing such ratios in the least squares sense leads to the following observations:

- A is minimized, meaning that the rest area of the membrane between the tiles decreases and the centers of pins get closer to each other, which decreases the membrane shearing along the contact edge and thus friction;
- B is maximized, meaning that the membrane gets more stretched in the actuated configuration;
- the opposite sides of one pair of tiles in contact tend to be stretched equally due to the least squares formulation.

Note that although Eq. 9 includes the desired model properties as soft constraints, they can easily be turned into hard constraints by setting their weights to infinity. In practice, we exploit this feature by first solving an unconstrained optimization using finite weights for all terms, which rapidly provides an initial solution, and then refining the initial solution using hard rigidity, collision, and pin area constraints.

5.2 Local Refinement

The designs obtained after solving the approximate model in Eq. 9 display the desired properties, but due to the limited precision of the approximation, invalid contacts may still be present. We could resolve them by running the full optimization problem in Eq. 3, but it would generate performance issues due to the computational cost of elastic membrane simulation.

ALGORITHM 1: Optimization Algorithm

Function *Coarse Optimization* ($\mathbf{x}$, ω_i)
  while *not Accepted* **do**
    Solve Eq. 9 for ω_i, $\mathbf{q}$, only soft constraints;
    User Places Cuts
  end
  Solve Eq. 9 for ω_i, $\mathbf{q}$, hard and soft constraints;
  Result: ω_i, $\mathbf{q}$
Function *Local Refinement* ($\mathbf{P}$, $\mathbf{f}^e$, c)
  Define local domains as in Fig. 8;
  Solve Eq. 3 for local dofs and membrane;
  Update $\mathbf{P}$ in Parameter update domain;
  Update $\mathbf{f}^e$ in Elastic update domain;
  Result: $\mathbf{P}$, $\mathbf{f}^e$
Data: Mesh vertices $\mathbf{x}$; `/* Uniformly remeshed user input */`
ω_i := Init Conformal ($\mathbf{x}$) ; `/* Initialization by conformal map */`
ω_i, $\mathbf{q}$:= Coarse Optimization ($\mathbf{x}$, ω_i);
$\mathbf{f}^e$:= Global Elastic Forces for $\mathbf{P}$;
$\mathbf{f}^c$:= Solve Eq. 1 for $\mathbf{P}$, $\mathbf{f}^e$;
C := Set of contacts with nonzero terms of Eq. 5 for $\mathbf{x}$, $\mathbf{f}^c$;
for *each* $c \in C$ **do**
  $\mathbf{P}$, $\mathbf{f}^e$:= Local Refinement ($\mathbf{P}$, $\mathbf{f}^e$, c)
end
Result: $\mathbf{P}$

We solve this problem by exploiting the local dependence of elastic forces on model parameters. We have found that, in practice, for a given pair of tiles in contact, a change in their design parameters affects only the elastic forces on the neighboring tiles. We use this observation to run the elastic membrane simulations required in Eq. 3 on a smaller subdomain, significantly reducing the computational cost. It is important to note that contact forces do not show this local dependence and therefore need to be recomputed globally.

Specifically, in order to detect any possible invalid contacts, we start by evaluating the elastic and contact forces on the complete domain. Then, for every invalid contact, we define three subdomains. First, a *parameter update subdomain*, corresponding to the two tiles in contact. Second, an *elastic update subdomain*, which includes the previous subdomain and all neighboring tiles. And third, an *elastic simulation subdomain*, defined by the n-ring of surrounding tiles (including the previous subdomain), where in our examples $n = 4$. Fig. 8 illustrates the different subdomains.

We solve Eq. 3 for the design parameters associated with the *parameter update subdomain*. To update the elastic forces on pins, we simulate the elastic membranes in the *elastic simulation subdomain*, but we only update those in the *elastic update domain*, keeping the values of the elastic forces computed in the initial evaluation everywhere else. Then, we compute the contact forces by solving Eq. 1 on the complete domain (but using the locally updated elastic forces).

6 IMPLEMENTATION

In this section, we explain implementation details, including the computational design tool and the fabrication process.

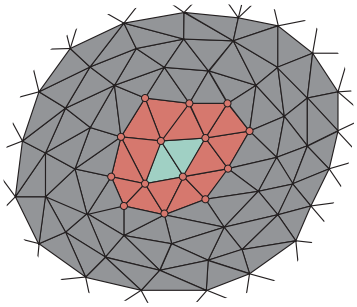


Fig. 8. We define 3 subdomains for local refinement. First, the *parameter update domain* (green), where design parameters are updated. Second, the *elastic update domain* (red), where elastic forces acting on tile vertices are updated. And third, the *elastic simulation domain* (gray), where the quasi-static elastic problem is solved.

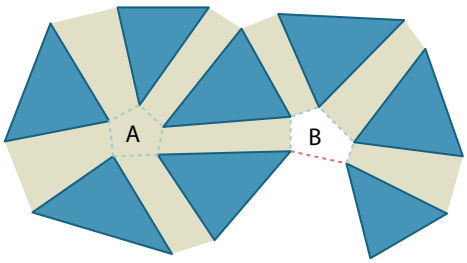


Fig. 9. Defining the elastic sheet simulation domain based on pins (shown as blue triangles): region A is added, since it forms a full polygon, but region B is not added, since it is missing one side.

6.1 Computational Design Tool

Simulation. We use the incompressible Neo-Hookean material model (see Appendix A.1 for details), which is able to accurately capture the behavior of the elastic sheets, even for large deformations. We aim to generate a simulation mesh with minimal area to avoid unpredictable behavior of non-stretched sheet areas and minimize the computational cost. To compute the domain, we bridge each pair of actuated pins with quadrilateral regions and include all polygons enclosed by them (see Fig. 9 for an example). Then, we mesh the domain with triangles constraining minimal angles and maximal areas, obtaining a simulation-ready mesh.

Optimization. We solve two optimization problems: first, a global optimization, which uses the approximate model as explained in Section 5.1, and second, a series of local refinements (Section 5.2) using the accurate force model in Eq. 3.

In order to obtain the initial guess for the first, we build a conformal map of $\mathcal{T}$ using the *Least-Squares Conformal Maps* method [Lévy et al. 2002] as a fast way to initialize the problem, and we scale it so that the area of each triangle fits the area of the corresponding triangle in $\mathcal{T}$. Then, we construct the tiles: the front face of each tile is placed inside the corresponding triangle of the conformal map, while the back face is defined by the front face, the tile thickness,

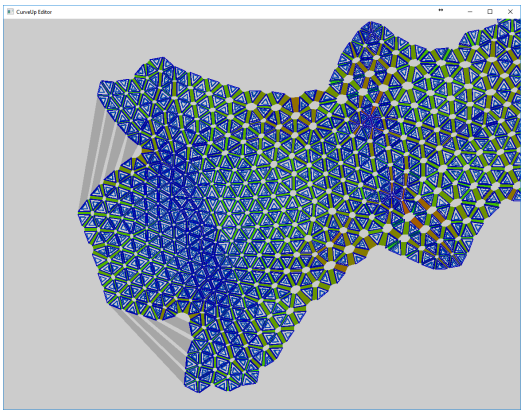


Fig. 10. Our user interface visualizes the current configuration, highlights problematic areas, and allows for placing cuts interactively. Editing the *Spot* model: two cuts introduced by the user are highlighted in gray. Red colors indicate potentially problematic contacts according to the approximation.

and the dihedral angles with neighboring elements. If necessary, we preprocess the input model and remesh it to obtain a regular, isotropic triangle mesh with given average edge length.

To solve the coarse optimization problems, we use the Active-set method of the *Knitro* optimization toolbox, and for the local refinements, we use the SQP method. Both methods require gradients of the cost function with respect to the design parameters. For all geometric variables, we use the algorithmic differentiation packages *CppAD* and *TOMLAB /MAD* to compute their derivatives. For the elastic forces and the contact forces, we compute analytic derivatives as shown in the supplemental material.

As mentioned in Sec. 4, we cut the tiles in order to avoid their intersections and compute the exact positions of the new contact points on the back side. Taking into account that the deviations from the target mesh are small, we avoid overloading our main optimization problem with additional internal dependencies by fixing the contact points' locations as their barycentric coordinates with respect to the corresponding front edge.

Another potential problem is the possibility of collisions between tile bases and the elastic sheet domains. We tackle this problem by choosing a sufficiently large value for the pin height.

User Interaction. Not all input shapes are reproducible, and some might require modifications that impact the aesthetics of the model. To address this problem, our system provides an interactive user interface that keeps the user in the design loop. Given an input model, the system aims to compute a valid design. If no valid design can be found, the user is informed and visual feedback is provided. As shown in Fig. 10, problematic areas where contact forces are not within the required tolerances are highlighted, allowing the user to modify the input model accordingly with an external modeling tool or introduce cuts within our user interface. These cuts disconnect tiles that would otherwise be in contact, thus reducing the complexity of the required actuation. We display the updates performed by the optimization while the user can interactively place and refine cuts. This process allows the design space to be intuitively explored until the fabrication constraints are fulfilled and the user is satisfied with the obtained design.

6.2 Fabrication

Our fabrication process, as illustrated in Fig. 11, starts by 3D printing the rigid tiles and a support structure, which guarantees that the computed tile layout is kept during post-printing manipulation. We also add small friction bumps at the interfaces of neighboring tiles, which allows us to use the infinite friction assumption explained in Eq. 5. In our experiments, we use a Stratasys J750 3D printer and its support material, which can be easily removed using a water jet.

We use latex sheets as elastic membranes. Since the actuation depends on the relative forces on both membranes and these depend linearly on the thickness, we can safely use latex sheets of different thicknesses for each model. This is useful to minimize the risk of cracks and other defects in latex when a larger stretching factor is required. To uniformly stretch the latex sheets, we use a stretching plate with regularly distributed teeth, which produce enough friction to keep the sheet stretched without any additional fixation. In order to ensure uniform stretch and the specified stretching factor, we add markers to the latex sheet and manually stretch the latex to bring the markers to the teeth of the stretching device.

Next, a uniform layer of glue is applied over the tiles' pins on one side of the sheet, and one of the pre-stretched latex sheets is glued to them. The layout is now preserved by the latex sheet, which is kept stretched on the device, and we can safely remove the support material using a water jet. The structure is then dried, and the second pre-stretched latex sheet is glued on the other side. Finally, we detach the latex sheets from the stretching devices, obtaining the final actuated 3D shape, and cut away the latex surplus. Before the surplus is removed, it might exert very large contraction forces at the boundary of the structure. In order to protect CurveUps from those forces potentially leading to the detaching of latex from the pins, we print a set of cylinders around the flat layout (see Fig. 1). The area of each cylinder is large enough to keep latex glued even when subject to very large latex surplus forces. After latex contraction, these cylinders are cut away together with the surplus.

7 RESULTS

We have tested our design and fabrication approach with a broad variety of shapes, including mathematical shapes, such as the *Half-sphere* and a subpart of a *Hyperboloid*, architectural shapes, such as the *Lilium Tower*, artist-designed models, such as the *Turtle*, the *Bump Cap*, and the *Mask*, and more standard computer graphics

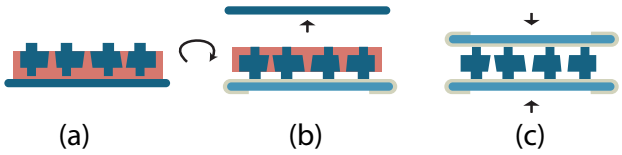


Fig. 11. Illustration of our fabrication process. (a) 3D printout (red support, blue tiles). (b) Gluing one side to the stretched latex sheet and washing away the support. (c) Gluing the other side.

models, such as *Spot*, the cow. All models were fabricated as a single patch in the flat configuration.

Table 1 shows details for each model as well as timings for the design computation and fabrication.

Half-sphere and *Hyperboloid* demonstrate the capabilities of our design system to produce simple models with positive and negative Gaussian curvatures, respectively. The *Lilium Tower* illustrates the behavior of the system when working with a more complex architectural shape. The *Turtle*, the *Bump Cap*, and the *Mask* are custom models that illustrate the capabilities of the design approach when used by a professional designer. Finally, *Spot*, is our most complex model. Its locally high curvature as well as challenging geometry for flattening, such as the horns, make it impossible to exactly reproduce as a single piece. We smooth the horns of the original model since their initial shape is too extreme for our purposes, and we manually remove the bottom part. Therefore, *Spot's* geometry exemplifies a use case where a non-artist is able to interact with our system, tuning an existing model and introducing cuts, to obtain a layout that can be properly actuated into the final 3D shape. Fig. 12 shows the input model, the optimized 2D layout, and the fabricated prototypes.

Fixed Parameters. The minimum height of the tile body H_{body} and the height of the pins H_{pin} depend on the resolution of our fabrication device. For all of our fabricated objects, we have chosen $H_{\text{body}} \approx 3$ mm and $H_{\text{pin}} \approx 1$ mm, summing up to approximately 5 mm of total structure thickness. To assure a good bond between the pins and the latex sheets, we set the minimal pin area $A^{\text{pin}}_{\text{min}} \approx 4\,\text{mm}^2$ and $p_{\text{max}} \approx 3$ mm. In order to leave some space for the friction bumps between tiles in contact, we set $d^{\text{gap}}_{\text{min}} \approx 0.6$ mm and $p_{\text{min}} \approx 0.6$ mm. We uniformly stretch the latex sheets to 900% of the original area and therefore set the stretch ratio $\tau = 3$ for all examples.

Performance of Optimization. We evaluate the performance of our optimization starting from a uniform triangulation remeshing with an average edge length $L \approx 7$ mm, and we measure the timing to obtain a valid configuration. Our simplest model, *Half-Sphere*, consists of 594 tiles, and our most complex model, *Spot*, contains 2545 tiles. All computations were performed on a standard desktop computer with 3.50 Ghz and 8 cores. We present statistics for all of our models in Table 1. The coarse optimization time is divided into two parts: only soft constraints (the user interaction might also take place) and with hard constraints. We then evaluate the exact forces, which requires meshing and simulating two latex sheets and a least squares solution for the contact forces. While for some input shapes (*Half-sphere*, *Hyperboloid*, *Lilium*, and *Mask*) the coarse optimization approach already provided a valid configuration, we noticed that the other models had several invalid contacts (below 10), mostly along the boundary. For each invalid contact, we perform local refinement by running three iterations of the optimization problem (3). We observed significantly larger computation times in the presence of membrane buckling. This is a well-known problem, which could be addressed by using the relaxed energy density model from Tension Field Theory [Pipkin 1986; Steigmann 1990]. The convergence of this problem in our approach cannot be guaranteed in general, since the main optimization problem is highly nonlinear and non-convex, but in practice we observed that three optimization steps usually provide a satisfying solution.

Model	#Tiles	$\hat{A}$	$\bar{A}/\hat{A}$	Soft	Hard	Glob.	Loc.	Fab.
H.-Sph.	594	145	1.63	40	31	59	-	71
Hyperb.	571	128	1.64	51	27	33	-	77
Lilium	1844	449	1.46	318	286	173	-	130
Turtle	2212	535	1.67	665	426	1162	1096	144
B. Cap	2531	606	1.82	466	613	1876	2623	175
Mask	2428	638	1.55	530	547	1308	-	183
Spot	2545	479	1.87	1936	557	1362	6205	149

Table 1. For each model, we show the number of tiles, the area in the actuated configuration (cm^2), the ratio of the approximate area in the flat configuration to area in the actuated configuration, the coarse optimization times for only soft constraints, for hard constraints, the global verification time, the local refinement time (sec), and the required 3D printing time (min).

Accuracy. In order to validate the reproduction accuracy of the design and fabrication method, we have computed a quantitative evaluation for the *Lilium* model with the size of ~25 cm. We have 3D scanned the fabricated model using the *David* laser scanner and computed the Hausdorff distance between the target and designed models, which equals ~8 mm. We also visualize point-wise errors in Fig. 13, color-coded from blue for no error to red clamped at 4 mm. The average point-wise distance is below 1 mm. We also scanned our most challenging model, *Spot*, which is approximately 20 cm large. The average error is below 4.3 mm. The maximal error is 12.8 mm, observed at the cow's muzzle, since the head is slightly tilted down due to a highly curved and narrow neck connecting it to the body.

8 CONCLUSION

We have presented a new concept for designing structures we call *CurveUps*. CurveUps are stable, self-actuated, optionally doubly curved, smooth three-dimensional shells that form from an initially flat state. The core of our system is a computational approach for computing a physically valid layout, an optimal shape of tiles, and pins. Our method builds on an accurate physics-based model for evaluating and optimizing a design. To make this global optimization problem tractable, we have introduced an effective approximation with localized updates that allow optimizations at interactive rates. As demonstrated by our results, we successfully reproduced a variety

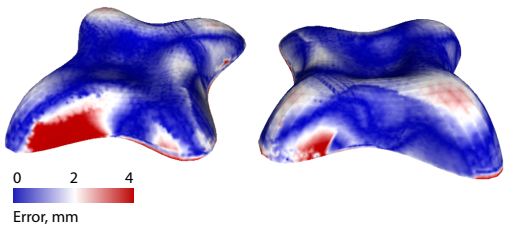


Fig. 13. A 3D scan of the *Lilium* model viewed from two opposite sides.

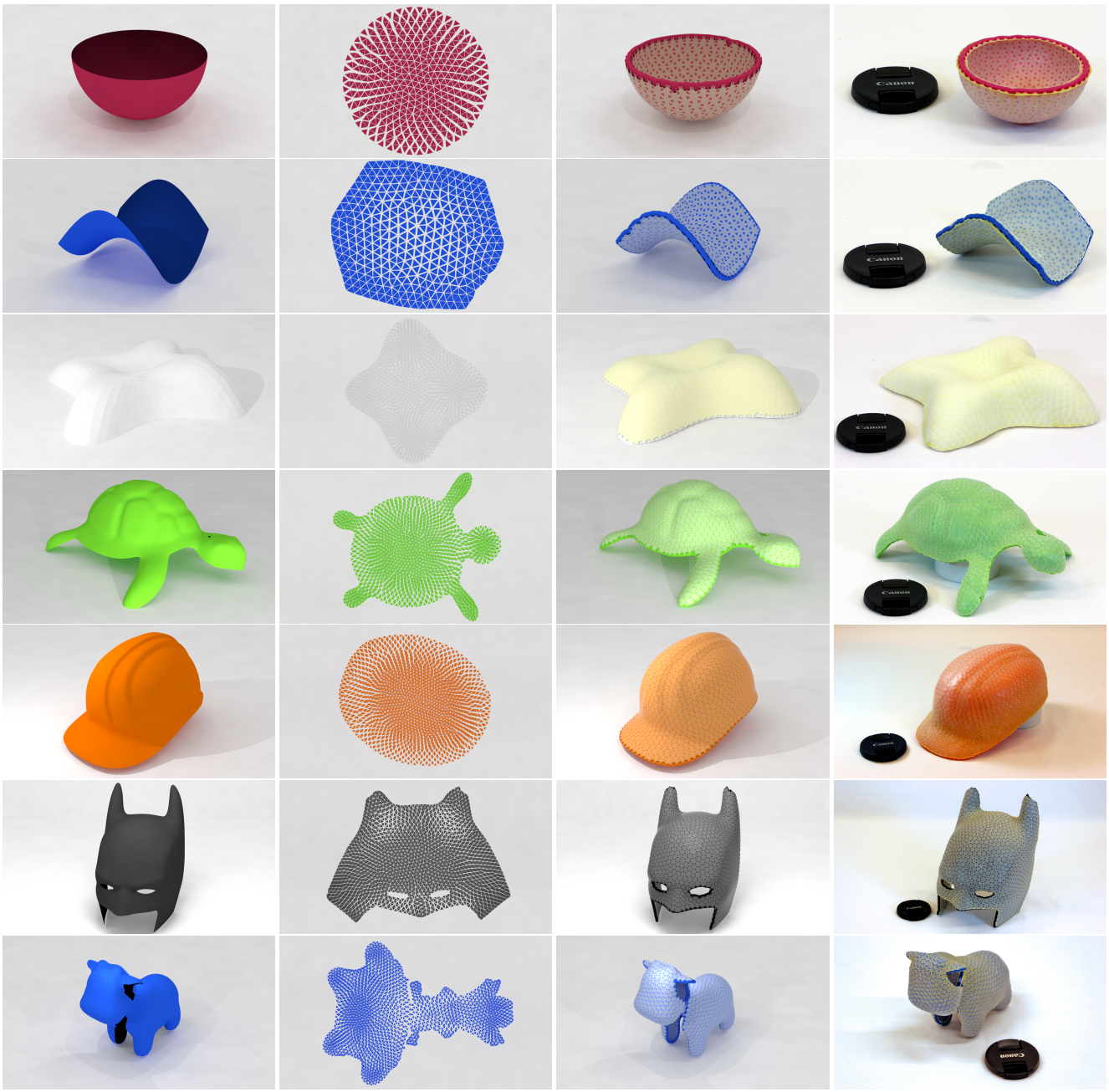


Fig. 12. Input mesh, flat layout, actuated configuration, and fabricated result for the test models. From top to bottom: *Half-Sphere*, *Hyperboloid*, *Lilium*, *Turtle*, *Bump Cap*, *Mask*, and *Spot*.

a compelling shapes and also evaluated the accuracy of several models by 3D scanning their surfaces and comparing them to the predicted surface geometry.

Limitations and Future Work. While our structures are guaranteed to be in equilibrium in the computed final configuration, we cannot guarantee that the structures self-actuate from the flat to the final shape. Along the deformation path, they might have to overcome states with a minimum in deformation energy. In practice, we address this issue by slightly manipulating the object by hand. Taking the actuation path into account during the design stage would be an exciting avenue for future work.

Due to physical limitations with regard to the maximum stretchability of the elastic sheet, both the local compressibility and dihedral angle between neighboring elements are constrained. Therefore, not all shapes are realizable as a single piece. We provide a user interface that visualizes problematic regions, allows users to place cuts, and interactively updates the configuration. A potential extension would be a system that automatically suggests cuts or strategies as to how to split challenging models into multiple fabricable pieces. For future work, another interesting route would be to indicate limits on achievable shape approximations and provide this information during shape modeling.

Currently, our model assumes infinite friction between tiles, which turned out to be a reasonable approximation in the presence of friction bumps. For a more accurate treatment, one could measure material properties and employ a friction model. We also currently neglect gravity, although this change is potentially easy to implement. In practice, these effects become more relevant as the size of the shell increases.

For future work, we plan to combine our approach with other types of actuating mechanisms, such as heat-shrink foils, or active materials, such as Stratasys' water absorptive 3D printer material, which changes its properties when exposed to different environmental conditions. We also plan to investigate alternative fabrication techniques, such as milling, and extend the functionality of tiles such that they automatically interlock when they come into contact with one another, resulting in a single rigid shape.

ACKNOWLEDGMENTS

We would like to thank Helmut Pottmann for valuable discussions, Fabrizio Amoroso for creating the artistic models, and all proofreaders and anonymous reviewers.

REFERENCES

Byoungkwon An, Shuhei Miyashita, Michael T Tolley, Daniel M Aukes, Laura Meeker, Erik D Demaine, Martin L Demaine, Robert J Wood, and Daniela Rus. 2014. An end-to-end approach to making self-folded 3D surface shapes by uniform heating. In *International Conference on Robotics and Automation (ICRA)*. 1466–1473.

Paolo Cignoni, Nico Pietroni, Luigi Malomo, and Roberto Scopigno. 2014. Field-aligned mesh joinery. *ACM Trans. Graph.* 33, 1 (2014), 11.

Erik D Demaine and Joseph O'Rourke. 2007. *Geometric folding algorithms*. Cambridge university press Cambridge.

Mario Deuss, Daniele Panozzo, Emily Whiting, Yang Liu, Philippe Block, Olga Sorkine-Hornung, and Mark Pauly. 2014. Assembling self-supporting structures. *ACM Trans. Graph.* 33, 6 (2014), 1.

Levi H Dudte, Etienne Vouga, Tomohiro Tachi, and L Mahadevan. 2016. Programming curvature using origami tessellations. *Nature materials* (Jan. 2016).

Michael Eigensatz, Martin Kilian, Alexander Schiftner, Niloy J Mitra, Helmut Pottmann, and Mark Pauly. 2010. Paneling architectural freeform surfaces. *ACM Trans. Graph.* 29, 4 (2010), 45.

Samuel M Felton, Michael T Tolley, ByungHyun Shin, Cagdas D Onal, Erik D Demaine, Daniela Rus, and Robert J Wood. 2013. Self-folding with shape memory composites. *Soft Matter* 9, 32 (July 2013), 7688–7694.

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.

A Sydney Gladman, Elisabetta A Matsumoto, Ralph G Nuzzo, L Mahadevan, and Jennifer A Lewis. 2016. Biomimetic 4D printing. *Nature materials* 15, 4 (April 2016), 413–418.

Elliot Hawkes, B An, NM Benbernou, H Tanaka, S Kim, ED Demaine, D Rus, and RJ Wood. 2010. Programmable matter by folding. *Proc. National Academy of Sciences* 107, 28 (2010), 12441–12445.

Kristian Hildebrand, Bernd Bickel, and Marc Alexa. 2012. crdbrd: Shape fabrication by sliding planar slices. In *Computer Graphics Forum*, Vol. 31. 583–592.

Yuki Igarashi, Takeo Igarashi, and Jun Mitani. 2012. Beady: interactive beadwork design and construction. *ACM Trans. Graph.* 31, 4 (2012), 49.

Martin Kilian, Simon Flöry, Zhonggui Chen, Niloy J Mitra, Alla Sheffer, and Helmut Pottmann. 2008. Curved folding. *ACM Trans. Graph.* 27, 3 (2008), 75.

Jungwook Kim, James A Hanna, Myunghwan Byun, Christian D Santangelo, and Ryan C Hayward. 2012. Designing responsive buckled surfaces by halftone gel lithography. *Science* 335, 6073 (2012), 1201–1205.

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 (2016), 89.

Tsz-Ho Kwok, Charlie C. L. Wang, Dongping Deng, Yunbo Zhang, and Yong Chen. 2015. Four-Dimensional Printing for Freeform Surfaces: Design Optimization of Origami and Kirigami Structures. *Journal of Mechanical Design* 137 (2015).

Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. 2002. Least Squares Conformal Maps for Automatic Texture Atlas Generation. *ACM Trans. Graph.* 21, 3 (July 2002), 362–371.

F Massarwi, C Gotsman, and G Elber. 2007. Papercraft Models using Generalized Cylinders. *15th Pacific Conference on Computer Graphics and Applications (PG'07)* (2007), 148–157.

Jun Mitani and Takeo Igarashi. 2011. Interactive design of planar curved folding by reflection. *Pacific Graphics (short paper)* (2011).

Jifei Ou, Mélina Skouras, Nikolaos Vlavianos, Felix Heibeck, Chin-Yi Cheng, Jannik Peters, and Hiroshi Ishii. 2016. aeroMorph-Heat-sealing Inflatable Shape-change Materials for Interaction Design. In *Proc. 29th Annual Symposium on User Interface Software and Technology*. ACM, 121–132.

A. C. Pipkin. 1986. The relaxed energy density for isotropic elastic membranes. *IMA Journal of Applied Mathematics* 36 (1986).

Helmut Pottmann, Alexander Schiftner, Pengbo Bo, Heinz Schmiedhofer, Wenping Wang, Niccolo Baldassini, and Johannes Wallner. 2008. Freeform surfaces from single curved panels. *ACM Trans. Graph.* 27, 3 (2008), 76.

Dan Raviv, Wei Zhao, Carrie McKnelly, Athina Papadopoulou, Achuta Kadambi, Boxin Shi, Shai Hirsch, Daniel Dikovsky, Michael Zyracki, Carlos Olguin, and others. 2014. Active printed materials for complex self-evolving deformations. *Scientific reports* 4 (2014).

Jennie Ryu, Matteo D'Amato, Xiaodong Cui, Kevin N Long, H Jerry Qi, and Martin L Dunn. 2012. Photo-origami—Bending and folding polymers with light. *Applied Physics Letters* 100, 16 (2012), 161908.

Yuliy Schwartzburg and Mark Pauly. 2013. Fabrication-aware Design with Intersecting Planar Pieces. *Computer Graphics Forum* 32, 2pt3 (2013), 317–326.

Robert F Shepherd, Filip Ilievski, Wonjae Choi, Stephen A Morin, Adam A Stokes, Aaron D Mazzeo, Xin Chen, Michael Wang, and George M Whitesides. 2011. Multigait soft robot. *Proc. National Academy of Sciences* 108, 51 (2011), 20400–20403.

Mélina Skouras, Stelian Coros, Eitan Grinspun, and Bernhard Thomaszewski. 2015. Interactive surface design with interlocking elements. *ACM Trans. Graph.* 34, 6 (2015), 224.

Mélina Skouras, Bernhard Thomaszewski, Bernd Bickel, and Markus Gross. 2012. Computational Design of Rubber Balloons. *Computer Graphics Forum* 31, 2pt4 (May 2012), 835–844.

Mélina Skouras, Bernhard Thomaszewski, Peter Kaufmann, Akash Garg, Bernd Bickel, Eitan Grinspun, and Markus Gross. 2014. Designing inflatable structures. *ACM Trans. Graph.* 33, 4 (July 2014), 1–10.

D. J. Steigmann. 1990. Tension-field theory. *Proc. Royal Society of London. Series A: Mathematical and Physical Sciences* 429 (1990).

Chengcheng Tang, Pengbo Bo, Johannes Wallner, and Helmut Pottmann. 2016. Interactive design of developable surfaces. *ACM Trans. Graph.* 35, 2 (2016), 12.

Michael T Tolley, Samuel M Felton, Shuhei Miyashita, Daniel Aukes, Daniela Rus, and Robert J Wood. 2014. Self-folding origami: shape memory composites activated by uniform heating. *Smart Materials and Structures* 23, 9 (2014), 094006.

Etienne Vouga, Mathias Höbinger, Johannes Wallner, and Helmut Pottmann. 2012. Design of self-supporting surfaces. *ACM Trans. Graph.* 31, 4 (2012), 87.

A APPENDIX

A.1 Thin Sheet Simulation

In order to compute the elastic forces acting on the tiles, we use the incompressible Neo-Hookean material model since it captures the behavior precisely enough, even for large strains. The Right Cauchy Green tensor for a shell can be written as

$$\bar{\mathbf{C}} = \mathbf{F}^T\mathbf{F} = \lambda_1\mathbf{N}_1\mathbf{N}_1^T + \lambda_2\mathbf{N}_2\mathbf{N}_2^T, \tag{17}$$

where λ_1, λ_2, $\mathbf{N}_1$, and $\mathbf{N}_2$ are the principal stretches squared and their corresponding eigenvectors.

Due to incompressibility and the absence of transverse shearing, we can write down the Cauchy Green tensor as:

$$\mathbf{C} = \begin{bmatrix} \bar{\mathbf{C}} & 0 \\ 0 & (\lambda_1\lambda_2)^{-1} \end{bmatrix} \tag{18}$$

and the strain energy density function as:

$$\Psi = \kappa(\operatorname{tr}\mathbf{C} - 3) = \kappa(\lambda_1 + \lambda_2 + \frac{1}{\lambda_1\lambda_2} - 3), \tag{19}$$

where κ is the stiffness coefficient.

Using Constant Strain Triangles, we calculate the elastic forces as:

$$\mathbf{f}_i^e = -\sum_{e\in\mathcal{F}_i} \frac{\partial\Psi^e}{\partial\mathbf{x}_i} V_e = -\sum_{e\in\mathcal{F}_i} \left(\frac{\partial\Psi^e}{\partial\lambda_1^e}\frac{\partial\lambda_1^e}{\partial\mathbf{x}_i} + \frac{\partial\Psi^e}{\partial\lambda_2^e}\frac{\partial\lambda_2^e}{\partial\mathbf{x}_i} \right) hA_e,$$

where $\mathcal{F}_i$ is the set of incident faces to vertex i, h is the thickness of the latex sheet in the rest configuration, A_e is the area of element e in the rest configuration, $V_e = hA_e$ is the volume of element e in the rest configuration and $\mathbf{x}_i$ is here redefined to be elastic sheet vertices.

While each pin may potentially be connected to many elastic sheet domain vertices along the edges, we redistribute the forces over the three pin vertices weighted by their barycentric coordinates.

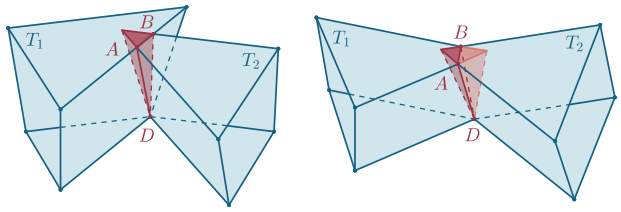
A.2 Computation of Contact Points

We only take into account interaction between tiles sharing a front face edge. These pairs of tiles always have a flat interface with four contact points shared by both tiles. These points define a new "truncated" shape of each tile such that they do not intersect anywhere in the actuated structure.

In order to resolve self-intersections of tiles and compute correct contact points, we cut each tile by three planes, one per front face vertex. In Fig. 14, we denote D as the front face vertex shared by two tiles, and A and B are the intersection points for tile edges and contact planes. We cut each pair of tiles with planes ABD.

The points A and B can be located on one edge of the tile or on two different edges. We call an intersection Type 1 if one tile has both intersection points on one edge (see Fig. 14a), and Type 2 if for both tiles the intersection points lie on different edges (see Fig. 14b). In the first case, only one tile is effectively cut, whereas in the second case both tiles are cut. It is easy to see that there is no other type of intersection.

Originally, each contact face has contact points set to the four vertices of the contact face, but each time the tile is cut, we update the contact points to the new intersection points. Note that a pair of tiles in contact never has an intersection to resolve, since they share the entire front face edge, not just one vertex.



(a) Type 1: only T_2 is affected. (b) Type 2: both tiles are affected.

Fig. 14. Two types of intersections of tiles, cut by plane ABD, where D is a front face vertex and A and B are the intersection points for tile edges and contact planes.

Received January 2017; revised February 2017; final version April 2017; accepted April 2017