Computational Design of Actuated Deformable Characters

Mélina Skouras[1,2] Bernhard Thomaszewski[2] Stelian Coros[2] Bernd Bickel [2] Markus Gross[1,2]
[1]ETH Zurich [2]Disney Research Zurich

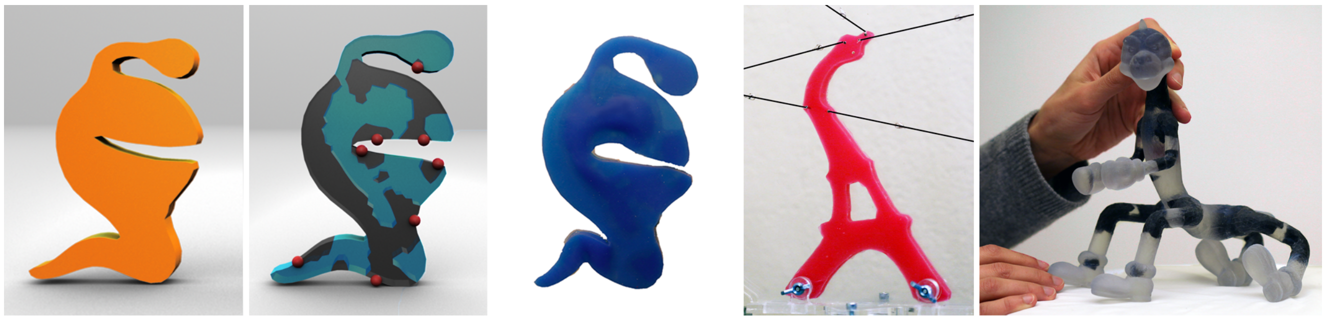

Figure 1: *We present a method to convert animated digital characters into physically fabricated prototypes. Our physical characters can be actuated using pins, strings, or posed by hand.*

Abstract

We present a method for fabrication-oriented design of actuated deformable characters that allows a user to automatically create physical replicas of digitally designed characters using rapid manufacturing technologies. Given a deformable character and a set of target poses as input, our method computes a small set of actuators along with their locations on the surface and optimizes the internal material distribution such that the resulting character exhibits the desired deformation behavior. We approach this problem with a dedicated algorithm that combines finite-element analysis, sparse regularization, and constrained optimization. We validate our pipeline on a set of two- and three-dimensional example characters and present results in simulation and physically-fabricated prototypes.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation; I.6.8 [Simulation and Modeling]: Types of Simulation—Animation

Keywords: computational materials, physically-based simulation, elastic solids, control

Links: DL PDF

1 Introduction

Character design is a vital part of animated movie production, game development and other applications of computer graphics. Many virtual characters are rigidly articulated, others are very deformable, and most of them show properties between these two extremes ranging from humanoid virtual actors with bulging muscles, to invertebrate figures like jelly monsters and stylized background characters such as plants, buildings and other man-made objects. Digital characters are typically created solely for the virtual worlds they live in. However, many other applications such as theme park attractions, exhibitions, artistic installations or next-generation games require real, physical embodiments of these figures. While there is an extensive set of tools for digital character design and animation, translating animated characters to the real world is an extremely difficult task. This problem is made even more evident by the quickly growing availability of rapid manufacturing devices and services that might soon lead to a switch from mass fabrication to personalized design of characters such as action figures.

Realizing this technological trend, recent work by Bächer et al. [2012] and Calì et al. [2012] proposed solutions for transforming articulated digital characters into 3D-printed figures that can be posed in various ways. While this is a significant advancement in fabrication-oriented character design, these methods are restricted to rigidly articulated characters and, more importantly, do not address the problem of how to animate the resulting figures.

Motivated by these observations, we propose a method for fabrication-oriented design of actuated deformable characters. Given a digital representation of an animated (or animatable) character as input, we seek to find a system of external actuators as well as an internal material distribution that allow us to fabricate a physical prototype whose range of deformation and movements closely approximate the input. Our solution to this problem is a dedicated algorithm that combines finite-element analysis, sparse regularization, and constrained optimization. We demonstrate our method on a set of two- and three-dimensional example characters. We present results in simulation as well as physically-fabricated prototypes with different types of actuators and materials.

2 Related Work

Designing and animating digitial characters is a central research area in computer graphics. We refer the interested reader to the survey by McLaughlin et al. [2011] for an overview of the many challenges related to this task. Here, we focus on three fields that are most closely related to our research.

ACM Reference Format
Skouras, M., Thomaszewski, B., Coros, S., Bickel, B., Gross, M. 2013. Computational Design of Actuated Deformable Characters. ACM Trans. Graph. 32, 4, Article 82 (July 2013), 9 pages.
DOI = 10.1145/2461912.2461979 http://doi.acm.org/10.1145/2461912.2461979.

Computational Materials are used in animated movies and live-action films to simulate the motion of, e.g., hair, cloth, and muscles. For an overview of existing techniques, we refer to Nealen et al. [2006]. A common challenge with such simulations is the selection of material parameters that yield a desired deformation behavior. Since hand-selecting the *right* parameters of a nonlinear material model and the *right* spatial distribution of multiple materials is a virtually impossible task, Bickel et al. [2009] presented a method to measure the parameters from a set of samples. However, such a data-driven approach is an option only if the material, the character in our case, is already physically available. Recent work by Martin and colleagues [2011] offers an elegant way to sidestep the tedious taks of parameter selection by describing material behavior through a set of example deformations. However, it is unclear how to translate example-based materials to physical materials since a real-world counterpart might not even exist. We follow the example-based modeling paradigm but solve a very different problem—the deformation space of our characters is described by a set of example poses, but we ask that the character be composed of a set of given base materials and that example poses should be reachable using only a limited number of external actuators.

Structural Design Optimization is used in engineering to design, e.g., elastic structures [Bendsoe and Sigmund 2004]. Sizing optimization adapts the thickness of components of a model to meet, e.g., structural stability thresholds while keeping the topology of the model fixed. Stava et al. [2012] improve the structural strength of 3D printable objects based on a stress analysis by hollowing and thickening parts or inserting supporting struts. In a shape optimization problem [Haslinger and Mäkinen 2003], the goal is to find an optimal shape defined by a prescribed domain. In graphics, realistic structural models that can be interacted with are of importance in physical simulations. Smith et al. [2002] optimize the geometry and the mass of truss structures for designing, e.g., bridges, towers, or roof supports. Procedural modeling in combination with structural optimization is also used for the design of buildings [Whiting et al. 2012] and plant modeling [Hart et al. 2003], where static analysis has been used to balance the weight of branches for creating realistic tree structures.

Topology optimization can be seen as a generalization of this approach, involving additional features such as the number and location of holes as well as the connectivity of the domain [Rozvany 2009]. We draw inspiration from topology optimization by formulating the discrete material distribution problem as a continuous optimization task. While material optimization is only one part of our pipeline, it is to our knowledge the first scheme to allow for multiple target poses, nonlinear elastic models, unknown external forces, and extreme deformations.

Fabrication-Oriented Design of physically reproducible objects recently gained increased attention in the computer graphics community. Several approaches were presented for reproducing properties such as reflectance [Weyrich et al. 2009; Malzbender et al. 2012], subsurface scattering [Hasan et al. 2010; Dong et al. 2010], and shape in the context of computing burr puzzles from 3D models [Xin et al. 2011], designing plush toys [Mori and Igarashi 2007], or designing furniture [Lau et al. 2011; Umetani et al. 2012]. Bächer et al. [2012] and Cali et al. [2012] presented systems for creating 3D-printable posable characters, whereas Zhu et al. [2012] proposed a method to synthesize mechanical toys given the motion of their features as input. Common to these approaches is that they generate models consisting of static geometry or piecewise *rigid* parts. In contrast, our work addresses the problem of computing a *deformable* character. The fabrication of deformable models has been investigated in a number of recent works, including materials with desired deformation behavior [Bickel et al. 2010], custom-shaped rubber balloons [Skouras et al. 2012], and synthetic skin for animatronic figures [Bickel et al. 2012]. Our work shares some of these goals but takes a significantly different approach. Instead of designing the rest shape of an object made of a single material, we optimize for the spatial distribution of multiple materials, providing a much larger and more expressive design space. Furthermore, we automatically compute the number of actuators and their respective locations and applied forces that allow us to accurately capture the deformation space of the input model.

3 Overview

Our method accepts as input a mesh describing a deformable character in its neutral state as well as a set of target shapes that represent desired deformations. We then optimize for the actuation parameters (number, placement, and forces) and an internal material distribution that allow us to fabricate a physical character whose range of deformation closely approximates the target shapes. Our formulation admits arbitrary types of actuators, but we focus on three variants in this work: pin-type actuators that can apply arbitrary forces at a given location, string-type actuators that can apply forces only in certain directions, and clamp-type actuators that prescribe the positions for sets of vertices, thus emulating the process of posing characters by hand. As summarized in Fig. 2, our pipeline for deformable character design consists of three main stages.

Initial Actuation The first stage determines an estimate of how many actuators should be used and in which regions they should be placed. We consider two variants: by default, we let the user select actuation points on the model. This is convenient when the user wants to have a particular number of actuators and/or knows roughly where they should be placed. In other cases, in particular for characters without apparent articulation structure, it can be difficult for the user to make a suggestion. In this case, we automatically suggest a number of actuators and their locations using sparse regularization.

Actuator Locations Given the number of actuators and an initial estimate for their locations, we next adjust their placement such as to minimize the overall distance of the model to the individual target poses. At this point, we take into account the type of actuation, i.e., strings or pins, and solve the problem using constrained optimization.

Material Optimization With the actuator positions fixed on the model, the third stage computes an internal material distribution that further optimizes the matching for the individual target poses. We assume that there are two materials available and initially allow each element of the model to assume an arbitrary convex combination of the base materials. As the optimization progresses, we drive the elements' materials toward one of the base materials, which eventually results in a discrete material layout that is ready for fabrication.

Finally, we use the optimized actuator locations and material distribution to fabricate a physical prototype of the deformable character using rapid prototyping technology.

4 Theory

The input to our model consists of a deformable character in its neutral pose as well as a set of target deformed poses. We represent the models using triangle meshes for 2D characters and tetrahedron meshes in the 3D case. We let n_v denote the number of vertices in the mesh and use $\mathbf{X} \in \mathbb{R}^{\dim \cdot n_v}$ to refer to the vector of undeformed positions, where dim denotes the number of dimensions. Likewise,

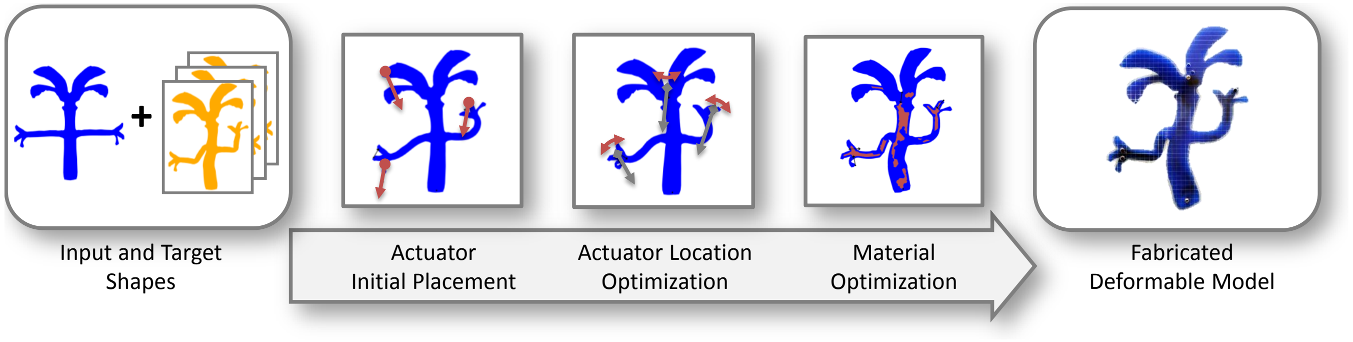
Input and Target Shapes
Actuator Initial Placement
Actuator Location Optimization
Material Optimization
Fabricated Deformable Model

Figure 2: *An overview of our optimization scheme: an input character and target shapes are provided by the user, the number of actuators and their initial locations are determined, the positions of the actuators are optimized, and an internal material distribution is computed. Finally, the physical prototype is fabricated.*

we let $\mathbf{t}^i \in \mathbb{R}^{\dim \cdot n_v}$ for $i \in \{1, \ldots, n_p\}$ denote the position vectors of the n_p target poses.

4.1 Elastic Model

We start by constructing an elastic model of the input shape using finite elements. Since the characters that we consider exhibit large deformations, we use a nonlinear deformation measure but retain linear elements for the sake of efficiency. For simplicity, we first focus on the three-dimensional case. Let $\mathbf{X}^e \in \mathbb{R}^{12}$, resp. $\mathbf{x}^e \in \mathbb{R}^{12}$, denote the vectors of concatenated undeformed, resp. deformed, nodal positions $\mathbf{X}_i^e$, $0 \leq i \leq 3$, resp. $\mathbf{x}_i^e$, $0 \leq i \leq 3$ of a single tetrahedral element e. We first compute the deformation gradient $\mathbf{F}^e(\mathbf{X}^e, \mathbf{x}^e) = \mathbf{d}\mathbf{D}^{-1}$ where $\mathbf{d}$ is the $(3{\times}3)$ matrix whose columns hold the edge vectors $\mathbf{d}_i = \mathbf{x}_i^e - \mathbf{x}_0^e$. Analogously, the edge vector matrix $\mathbf{D}$ is defined through $\mathbf{D}_i = \mathbf{X}_i^e - \mathbf{X}_0^e$. We define the elastic energy density of the element using a Neo Hookean material model,

$$\Psi^e(\mathbf{F}^e) = \mu(\mathrm{tr}(\mathbf{C}^e) - 3) + \frac{\kappa}{2}(\det(\mathbf{F}^e) - 1)^2 , \tag{1}$$

where μ and κ are material parameters and $\mathbf{C} = (\mathbf{F}^e)^T\mathbf{F}^e$ is the right Cauchy-Green tensor. The elastic energy of the deformed element is then obtained by integrating (1) over its domain. Since we use linear finite elements, the deformation gradient is constant across the element such that we have $W^e = \Psi^e V^e$, where V^e denotes the volume of the tetrahedron. The global deformation energy $W(\mathbf{X}, \mathbf{x})$ between the rest state $\mathbf{X}$ and an arbitrary deformed state $\mathbf{x}$ is obtained by summing up elemental contributions as $W(\mathbf{X}, \mathbf{x}) = \sum_e W_e(\mathbf{X}_e, \mathbf{x}_e)$. Finally, the elastic energy gives rise to internal forces $\mathbf{f}_{\text{int}}^i \in \mathbb{R}^{\dim \cdot n_v}$ as

$$\mathbf{f}_{\text{int}} = -\frac{\partial W(\mathbf{X}, \mathbf{x})}{\partial \mathbf{x}} . \tag{2}$$

For the two-dimensional case, we use linear triangle elements, and the derivation of the elastic energy is largely similar. As an important difference, however, we follow Skouras et al. [2012] and expand the two-dimensional Cauchy-Green tensor to three dimensions by inferring the thickness stretch from the assumption of volume-preserving deformations. This allows us to also use (1) for the two-dimensional case.

4.2 Optimization Scheme

Basic Formulation Our goal is to compute the internal material structure of the deformable model and the sets of actuation forces that, when applied to the model, lead to deformed states that are as close as possible to their corresponding target shapes. As it is generally not possible to exactly match the target poses, we ask that the distance of each deformed pose $\mathbf{x}^i$ to its corresponding target pose $\mathbf{t}^i$ be minimized. We quantify *closeness* using a distance energy function that measures differences in positions on the model's boundary between deformed $\mathbf{x}^i$ and target poses $\mathbf{t}^i$ as

$$E_d = \sum_i E_d^i(\mathbf{t}^i, \mathbf{x}^i) = \tfrac{1}{2} \sum_i \sum_{j \in \mathcal{B}} ||\mathbf{x}_j^i - \mathbf{t}_j^i||^2 , \tag{3}$$

where $\mathcal{B}$ denotes the set of boundary vertices. Letting $\mathbf{p}$ denote the vector of generic parameters that we want to optimize for, we seek to find optimal values for $\mathbf{p}$ that minimize the distance energy E_d when the physical system is at equilibrium. This can be formulated as a constrained optimization problem:

$$\begin{aligned} &\min_{\mathbf{x}^i, \mathbf{p}} \sum_i^{n_p} E_d^i(\mathbf{t}^i, \mathbf{x}^i) \\ &\text{s.t} \quad \mathbf{f}_{\text{int}}^i(\mathbf{x}^i, \mathbf{p}) = -\mathbf{f}_{\text{ext}}^i(\mathbf{x}^i, \mathbf{p}) \quad \forall i \in 1 \ldots n_p , \end{aligned} \tag{4}$$

where $\mathbf{f}_{\text{ext}}^i$ are external forces including actuation. Note that the objective function prefers deformed poses $\mathbf{x}^i$ that are close to their target counterparts, whereas the constraints require that each of the $\mathbf{x}^i$ is a physically-feasible solution, i.e., represents an equilibrium state in which the internal forces $\mathbf{f}_{\text{int}}^i$ are in balance with the externally applied forces $\mathbf{f}_{\text{ext}}^i$.

Numerical Optimization The minimization problem (4) has a large number of degrees of freedom as well as numerous nonlinear constraints which have to be satisfied exactly. Following recent work in graphics [Skouras et al. 2012; Narain et al. 2012], we use an augmented Lagrangian method (ALM) to optimize for the system's variables. In ALM, the usual Lagrangian function is augmented by an additional term penalizing constraint violations. The resulting function is minimized iteratively by alternating between unconstrained minimization of the modified objective and Lagrange multiplier updates. For our specific optimization problem, the augmented objective function has the following form,

$$\Lambda(\mathbf{x}, \mathbf{p}) = E_d(\mathbf{x}) - \lambda^t \mathbf{f}(\mathbf{x}, \mathbf{p}) + \frac{\mu}{2}||\mathbf{f}(\mathbf{x}, \mathbf{p})||^2 , \tag{5}$$

where $\mathbf{x} \in \mathbb{R}^{\dim \cdot n_v \cdot n_p}$ is the concatenation of the n_p deformed position vectors $\mathbf{x}^i$ and $\mathbf{f}(\mathbf{x}, \mathbf{p}) = \mathbf{f}_{\text{int}}(\mathbf{x}, \mathbf{p}) + \mathbf{f}_{\text{ext}}(\mathbf{x}, \mathbf{p})$.

In every step of the optimization, the multipliers λ are fixed, and the minimization of Λ is performed using a Newton-Raphson procedure including line search. Bound constraints on the variables are handled using a gradient projection approach as described in Nocedal and Wright [2000]. After each minimization step, we update the Lagrange multipliers using the scheme $\lambda_i = \lambda_i - \mu \mathbf{f}_i$, provided that the decrease in the constraints is sufficient. Otherwise, the weight of the penalty term μ is increased while keeping the multipliers unchanged. The procedure stops when both the gradients of Λ and of the constraints $\mathbf{f}$ are smaller than given thresholds.

4.3 Initial Actuation

Given a deformable character and its desired range of deformation, we first have to determine a number of actuators and an initial estimate for their locations. We will require that the actuators be placed on the boundary of the characters and denote by $\mathcal{Q} = (\mathbf{l}, \mathbf{q}^1, \ldots, \mathbf{q}^{n_p})$ the actuation system where $\mathbf{l} \in \mathbb{R}^{\dim \cdot n_a}$ holds the locations of the n_a actuators in the rest state and each vector $\mathbf{q}^i \in \mathbb{R}^{\dim \cdot n_a}$ holds the actuation forces for a target pose $\mathbf{t}^i$. In some cases, the structure of a character will largely imply the number of actuators. In other cases, the user can have a specific idea of how many actuators should be used and where they should be placed. Yet, for many characters, in particular those without apparent skeletal structure, the answer to this question is far from obvious.

In order to treat both of these cases, we support two variants of actuator layout in our system: the user can hand-select a desired number of points on the undeformed model, in which case we directly proceed to the next stage of our pipeline. Otherwise, we ask the user to specify an admissible range for the number of actuators (e.g. 5-10) and automatically compute a suggestion as described subsequently.

The underlying reasoning of our approach is that it is generally desirable to have the smallest number of actuators that yield a sufficiently good approximation of the target poses. Indeed, the mechanical complexity of real-world actuation systems typically imposes strict bounds on the number of actuators, as is the case for our string-based setup (see Fig. 5) and animatronic figures in general (see, e.g., [Bickel et al. 2012]). Finding the optimal number of actuators is an inherently discrete problem that does not directly lend itself to continuous optimization. Instead of turning toward specialized optimization methods such as mixed-integer programming, we draw inspiration from sparse regularization techniques used in, e.g., machine learning and image processing. Starting with a dense set of actuators on the boundary of the model, we introduce a regularizer that prefers sparse solutions, i.e., a small number of actuators.

A widely used approach for sparse regularization is to penalize the L_1-norm of the design variables, i.e., the sum of absolute values.

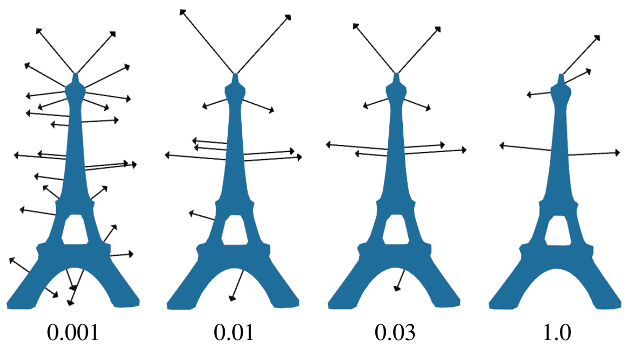

Figure 3: *Sparse regularization demonstrated on the* TourEiffel *example. The number of actuators (black arrows) is effectively controlled by the coefficient k_{sparse} (indicated below the figures).*

In our case, the design variables are force vectors comprising two or three components. Since all of these components have to be zero simultaneously in order to obtain a zero net force, we introduce a regularizer that penalizes the sum of force magnitudes. Considering only a single target pose i for the moment, we define a sparse regularizer as

$$R_{\text{sparse}}(\mathbf{q}^i) = k_{\text{sparse}} \sum_j^{n_v} \left(\sum_k^{\dim} (\mathbf{q}_j^i)_k^2 \right)^{1/\alpha} , \tag{6}$$

where k_{sparse} is a scaling parameter and $\alpha > 2$ generalizes the L_1-norm to more strongly penalize small values. In order to incorporate multiple target poses, we ask that a given actuator should have a zero value only if its force vectors vanish for all target poses. This requirement can be modeled as

$$R_{\text{sparse}}(\mathbf{q}^1, \ldots, \mathbf{q}^{n_p}) = k_{\text{sparse}} \sum_j^{n_v} \left(\sum_i^{n_p} \sum_k^{\dim} (\mathbf{q}_j^i)_k^2 \right)^{1/\alpha} . \tag{7}$$

We add this reqularizer to the basic optimization problem (4) with $\mathbf{p} = (\mathbf{q}^1, \ldots, \mathbf{q}^{n_p})$ and, depending on the value of k_{sparse}, obtain solutions with different numbers of actuators (see Fig. 3). Since it is not possible to choose k_{sparse} *a priori* in order to obtain a desired number of actuators n_a, we solve a sequence of problems with different values of k_{sparse} until we find a solution within the admissible range specified by the user.

4.4 Actuator Locations

The initial actuation step provides us with a number of actuators and their initial locations. Next, we improve the matching quality, i.e., the correspondence between deformed poses $\mathbf{x}^i$ and target poses $\mathbf{t}^i$, by allowing the actuators to slide along the boundary of the character. In order to ensure that actuators can move freely along the surface but not in their normal directions, we introduce a penalty term that attracts the actuators to the zero-levelset of a local distance field Φ around the surface. Using the implicit moving least squares (IMLS) method of Öztireli et al. [2009], the distance field is defined as

$$\Phi(\mathbf{x}) = \frac{\sum_k \mathbf{n}_k \cdot (\mathbf{x} - \mathbf{X}_k)\phi_k(\mathbf{x})}{\sum_k \phi_k(\mathbf{x})} , \tag{8}$$

where $\phi_k(\mathbf{x}) = \left(1 - \frac{||\mathbf{x}-\mathbf{X}_k||_2^2}{h^2}\right)^4$ are locally-supported kernel functions that vanish beyond their support radius h. Using this formulation, we define a penalty energy

$$E_{\text{IMLS}} = \sum_i \Phi(\mathbf{l}_i)^2 \tag{9}$$

that attracts the actuator locations $\mathbf{l}_i$ to the surface. In order to ensure that the actuation forces vary smoothly as the actuators move along the surface, we distribute them to a local neighborhood of vertices using the IMLS-kernels

$$\mathbf{f}_k^i = \frac{\mathbf{q}_j^i \phi_k(\mathbf{l}_j)}{\sum_{l \in \mathcal{S}_j} \phi_l(\mathbf{l}_j)} . \tag{10}$$

Here, $k, l \in \mathcal{S}_j$ where $\mathcal{S}_j$ denotes the set of vertices whose kernel functions are nonzero at $\mathbf{l}_j$ (see also Fig. 4, *left*). In addition to this penalty term, we also want to prevent actuators from moving too close together as coinciding actuators would lead to a null-space

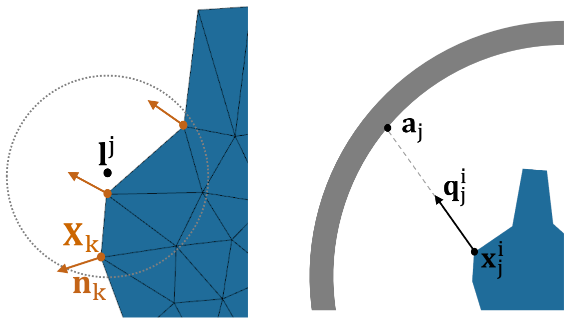

Figure 4: *Illustrations for the IMLS distance field (left) and the string-based actuator (right).*

during optimization. To this end, we define a C^2 continuous repulsion term

$$E_{\text{rep}}(\mathbf{l}_j, \mathbf{l}_k) = k_{\text{rep}}(\varepsilon_{\text{rep}} - ||\mathbf{l}_j - \mathbf{l}_k||)^3 \,, \tag{11}$$

which is activated whenever the distance between actuators is smaller than a threshold value ε_{rep}.

Adding (9) and (11) to (3), we compute the optimal positions $\mathbf{x}$ and parameters $\mathbf{p} = (\mathbf{l}, \mathbf{q}^1, \dots, \mathbf{q}^{n_p})$ which solve the problem (4).

String-Based Actuation The formulation so far assumed that the actuators can apply forces $\mathbf{q}_i$ in arbitrary directions. Since this is not the case for strings, a few adaptations have to be made. A string actuator is defined through an attachment point $\mathbf{a}_j$ located on an external support structure as well as another attachment point on the surface of the character. The force that can be exerted by such an actuator depends on the deformed configuration,

$$\mathbf{q}_j^i = k_j^i \frac{(\mathbf{a}_j - \mathbf{x}_j^i)}{||\mathbf{a}_j - \mathbf{x}_j^i||} \,, \tag{12}$$

where $k_j^i > 0$ corresponds to the tension of the string and $\mathbf{x}_j^i$ is the actuator's location on the deformed surface (see Fig. 4, *right*). Note that we require $k_j^i > 0$ since the string can only pull, not push. Using the formulation described above, we optimize for the actuator's location $\mathbf{l}_j$ on the rest state but add k_j^i and $\mathbf{a}_j$ as further degrees of freedom.

We enforce the condition that $\mathbf{a}_j$ be on the support structure using a formulation similar to (9). Assuming a circular support, we formulate the penalty energy as

$$E^j_{\text{string}}(\mathbf{a}_j) = (||\mathbf{a}_j - \mathbf{c}|| - r)^2 \,, \tag{13}$$

where r is the radius of the support and $\mathbf{c}$ denotes its center.

In order to limit the likelihood of strings intersecting with the model or tangling up during animation, we prefer string directions that are close to the boundary normals at the attachment point. We express this preference with a further penalty term

$$E_{\text{dir}} = k_{\text{dir}} \sum_i^{n_p} \sum_j^{n_a} \left(1 - \frac{\mathbf{q}_j^i}{||\mathbf{q}_j^i||} \cdot \mathbf{n}_i(\mathbf{l}_j)\right)^2 \,, \tag{14}$$

where $\mathbf{n}_i(\mathbf{l}_j) = \frac{\sum_k \mathbf{n}_k^i \phi_k(\mathbf{l}_j)}{\sum_k \phi_k(\mathbf{l}_j)}$ denotes the interpolated normal at the string location $\mathbf{l}_j$ in pose i.

4.5 Material Optimization

Even with the location optimization described in the previous section, characters can have poses that are difficult to achieve. Such difficult poses arise, e.g., from conventional articulation such as the sharp bending of an arm. Approximating such poses with a homogeneous material would require a large number of actuators. We address this problem by allowing the material properties to vary spatially, thus building preferences for deformation directly into the model.

As a basis for material optimization, we will assume that there is a library of non-miscible base materials (such as silicone and printable plastics) described by energy density functions W^i. For simplicity, we restrict considerations to two material types per character, typically a soft and a stiff one. We allow the material properties to vary among the elements but assume that each element consists of a homogeneous material. If we directly constrain each element to take on only material properties from the library, we arrive at a discrete optimization problem and its associated difficulties. In order to avoid the need for more complex optimization methods, we convert the discrete problem into a continuous one by allowing the per-element materials to be interpolations of the base materials. We start the optimization by allowing arbitrary (convex) combinations of the base materials and then progressively drive the interpolation weights to the boundary of the intervals, thus enforcing a discrete material distribution. On a technical level, we do not interpolate material descriptions but the elastic energies that would result from the different base materials. The effective deformation energy of a given element e is defined as

$$W(\mathbf{F}^e, \rho^e) = \rho^e \, W^1(\mathbf{F}^e) + (1 - \rho^e) \, W^2(\mathbf{F}^e) \,, \tag{15}$$

where ρ^e are interpolation weights. Adopting this interpolated material model and adding the interpolation weights per element as free variables to the optimization problem, we can solve for a material distribution that leads to an optimal approximation of the target poses by optimizing for the parameters $\mathbf{p} = (\mathbf{q}^1, \dots, \mathbf{q}^{n_p}, \rho^{e_1}, \dots, \rho^{e_n})$. However, in order to obtain a physically meaningful solution, we have to drive the interpolation weights to 0 or 1. We achieve this with a penalty energy of the form

$$R_{\text{mat}} = k_{\text{mat}} \sum_e \rho^e (1 - \rho^e) \,, \tag{16}$$

where k_{mat} is a scaling parameter that is progressively increased until a solution is found that satisfies $\rho^e(1 - \rho^e) < \varepsilon$, where ε denotes a small threshold value. While the penalty term R_{mat} will eventually ensure that all interpolation weights are either 0 or 1, they can potentially assume values out of this range in earlier iterations. We therefore have to explicitly enforce the bounds on the interpolation variables since we could otherwise encounter nonphysical material combinations that would hinder convergence.

The material optimization scheme computes material distributions that are optimal in the sense of approximating the target poses. However, multiple solutions can lead to equivalent approximations of the target poses. In our approach, we aim to find macroscopic material distributions rather than micro-level structures. We therefore favor larger material clusters rather than small, isolated islands by adding the regularizer,

$$R_{\text{smooth}} = k_{\text{smooth}} \sum_j (\rho^j - \frac{1}{n_j} \sum_{k \in \mathcal{T}_j} \rho^k)^2 \,, \tag{17}$$

where $\mathcal{T}_j$ denotes the set of n_j elements adjacent to element j. Finally, since our focus is on deformable characters, we also prefer

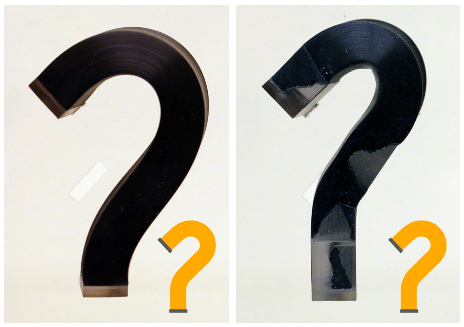

Figure 6: *3D printed example of a straight bar with uniform (left) and optimized (right) material distribution that is bent into a question mark shape by applying force only on its end caps. The small orange icon shows the target shape.*

soft materials over stiff ones in regions where the difference in approximation quality is small. We model this preference with a second regularization term,

$$R_{\text{soft}} = k_{\text{soft}} \sum_{i}^{n_e} (\rho^e)^2 \,, \qquad (18)$$

assuming that the soft material corresponds to $\rho = 0$.

5 Results

We evaluated our method by designing and fabricating six characters with different types of actuators and materials. For fabrication, we used an Objet Connex350 multi-material printer with two base materials of significantly different stiffnesses. We chose VeroClear, a rigid and transparent material, and TangoBlack+, a black material with properties similar to soft rubber. Three of the characters were printed directly ("Palmy3D", "Questionmark", "Grampolo"), whereas the remaining three were fabricated using silicone injection molding with 3D-printed rigid parts ("TourEiffel", "Palmy2D", "WormEye"). We simulated the material behavior during the design process using measured data for silicone [Bickel et al. 2012] and data provided by Objet for the 3D-printed materials. In the following paragraphs, we discuss our results and highlight the roles played by the individual stages of our design pipeline.

Actuator Optimization For all examples shown in this paper, we employed the sparse regularization approach for computing an appropriate number of actuators. We observed that, especially for characters that do not have an obvious internal structure, manually selecting the number of actuators and their placement proved to be difficult. Our automatic approach significantly simplified this design task.

Fig. 5 shows an example of string-based actuation. We extracted five key frames from an artist-generated input animation and used the sparse regularization method to automatically determine the number of actuators as well as their initial locations. We fabricated the model with silicone and attached strings driven by servo motors to playback the animation. The strings are routed through pulleys at a ring around the model, whose locations are optimized as well. We refer to the accompanying video for the full animation.

The "QuestionMark" example (Fig. 6) is posed using clamp-type actuators that fix both the position and orientation of the end caps of the model. The remaining examples were designed using pin-type actuators. We pose the planar characters by attaching small pins at the actuators locations. These pins are then plugged into precision-drilled holes in an acrylic plate to reproduce the target poses. The 3D characters, "Palmy3D" and "Grampolo", were designed using the same framework as for the 2D examples, but for simplicity, we pose and animate these models by hand.

The first stage of our pipeline provides an initial guess for the actuator locations and already leads to fair approximation quality in some cases. As can be seen from Table 1, however, the subsequent actuator location optimization is able to significantly reduce the error for all examples.

Material Optimization Allowing material properties to vary spatially further improves the visual and quantitative error of all characters. A particularly striking example can be seen in Fig. 9, where material optimization allowed us to create very different deformation styles with only two actuators. Note that our scheme leads to intuitive solutions if the character exhibits mostly rigid articulation, as is the case, e.g., for "Palmy2D" (Fig. 7) and "Grampolo" (Fig. 1, *right*). Although we used only one example pose for the "Grampolo" character, the optimization scheme was able to infer a meaningful material distribution, putting soft material at joint locations and rigid material at limbs. For characters with more complex deformations, however, the material distribution can be significantly more complex as shown in Fig. 8 and Fig. 9.

Weight Selection Our method uses a number of different penalty terms to enforce soft constraints or drive the solution toward a preferred subspace. The weights that we used for the different examples are listed in Table 2. For most of the weights, determining an appropriate value posed no difficulty, since the corresponding penalty terms were not directly competing with other objectives. However, the weights of the regularization terms (17) and (18) have a significant impact on the final material distribution. We set these weights by first selecting a value for (18) which, together with the target poses, determines the overall material structure. We use a default value for (17) and, if necessary, adjust it with 1-2 iterations to suppress small material islands without changing the boundary of larger structures too much.

Accuracy, Robustness and Performance Our fabricated prototypes show good agreement with the simulation. Fig. 7 illustrates the progressive improvement during each step of our pipeline including the final fabricated character.

A variation in the position of the actuators affects the resulting material distributions. However, in practice we observed that a small variation leads to a solution with similar quality. We also investigated whether multiple iterations of our pipeline would lead to improved results but did not observe any significant improvement after the first cycle in our experiments.

Statistics for each example including the number of elements, number of actuation points and computation times can be found in Table 2. The largest fraction of the computation time is spent on material optimization. This is mostly due to the fact that the optimization scheme uses several outer iterations to increase the parameter k_{mat} that eventually enforces each element to assume one of the base materials.

6 Conclusion

We presented a method for creating physically fabricated prototypes of animated digital characters. Our approach automatically finds a sparse set of actuation locations on the surface and optimizes the internal material distribution such that the resulting character

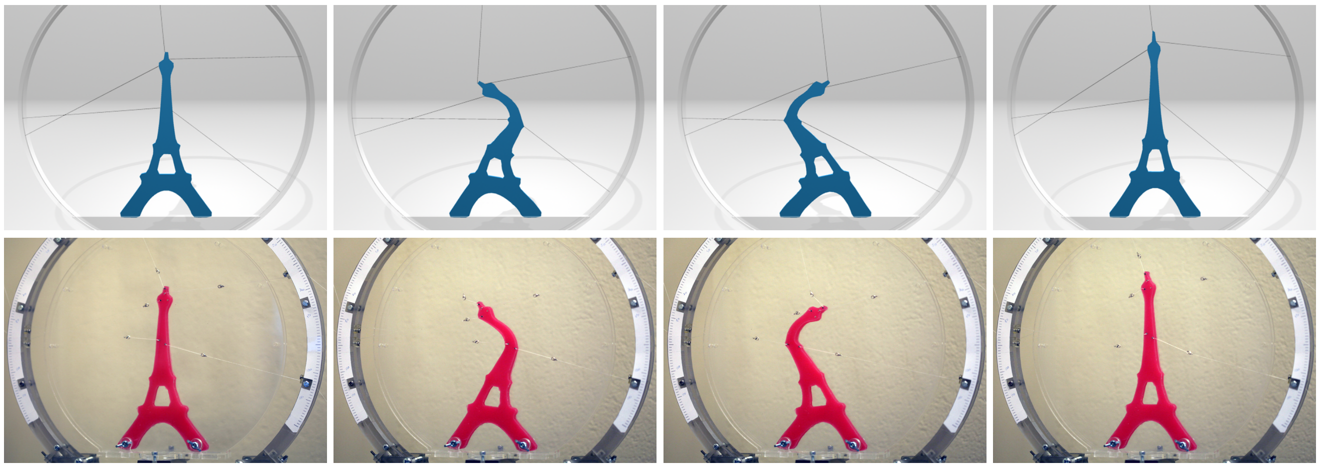

Figure 5: *"TourEiffel" model actuated with 5 strings. From a given animation, we extracted several key frames and automatically optimized the actuator locations. The top row shows the resulting deformations in simulation. We also fabricated the model with silicone and attached strings driven by servo motors to replay the animation (bottom row).*

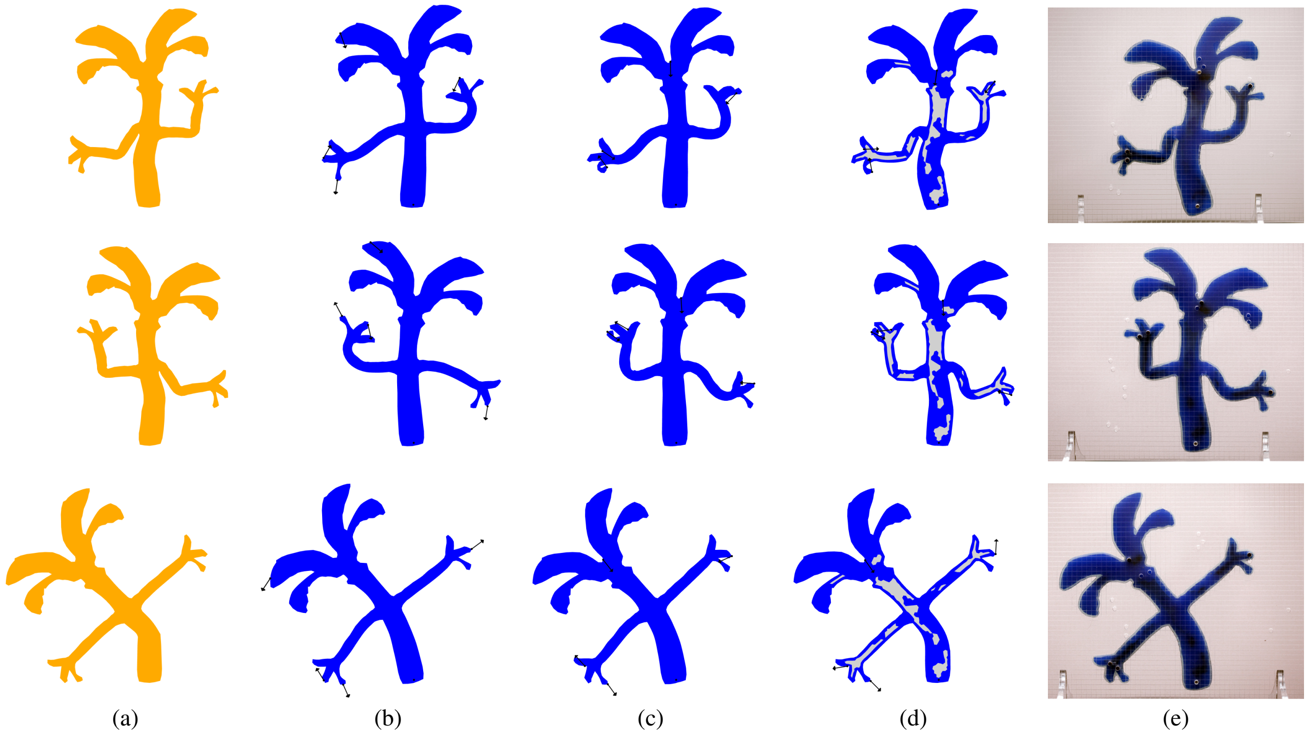

Figure 7: *"Palmy2D." Given 3 target poses (a), we use sparse regularization to find an initial location of 4 actuation points (b). We then refine the location of the actuation points by allowing them to slide on the surface (c) and optimize for an internal material distribution that allows us to better approximate the target poses (d). Columns (b-d) show the resulting simulated deformations of each optimization stage. We validated the results by fabricating the character with silicone and rigid insets. The fabricated character is then posed using pins (e).*

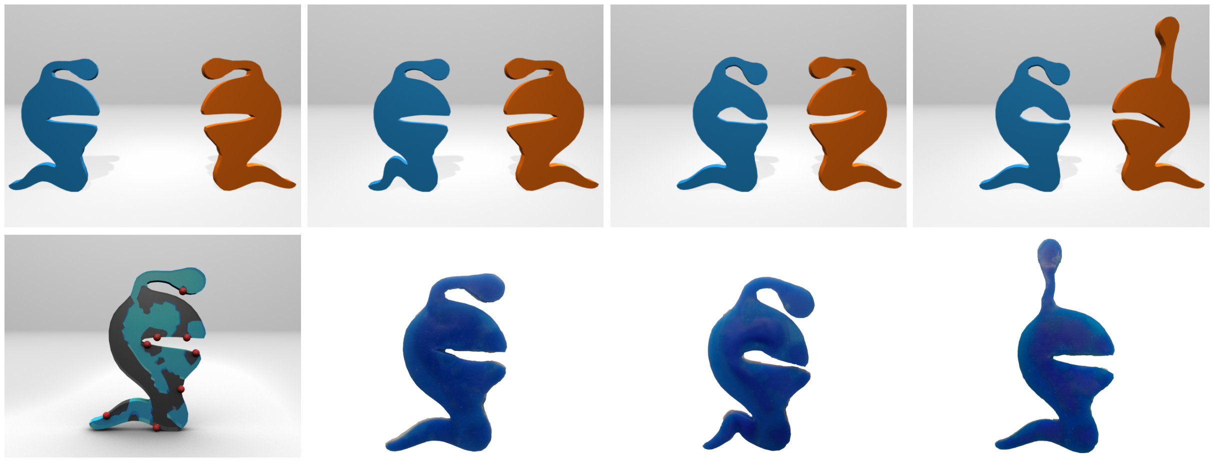

Figure 8: *"WormEye." Top row: Frames from an animation of two WormEye characters. Bottom row: Optimized material distribution and actuator locations for reproducing poses present in the animation (left) and real fabricated deformed character (right).*

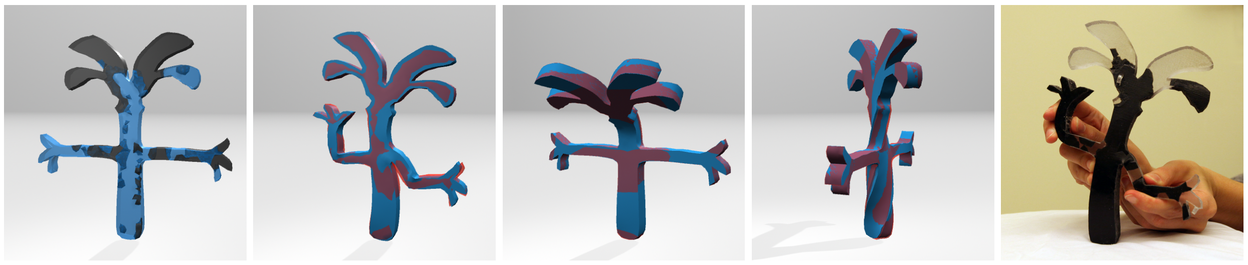

Figure 10: *"Palmy3D." From left to right: Rest pose with optimized material distribution (black/stiff, blue/soft), simulated poses (blue) overlaid with input target poses (red), 3D printed character.*

Model	Pose1	Pose2	Pose3	Pose4	Pose 5	Size
TourEiffel (1)	4.8/0.5	3.0/0.4	2.8/0.5	6.4/0.8	4.4/0.6	202
TourEiffel (2)	3.6/0.5	2.0/0.4	3.6/0.5	5.2/0.6	3.8/0.6	202
Palmy2D (1)	7.1/0.7	11.0/0.9	5.7/0.4	-	-	100
Palmy2D (2)	7.6/0.4	7.4/0.5	4.8/0.2	-	-	100
Palmy2D (3)	5.1/0.5	5.1/0.3	2.6/0.1	-	-	100
Questionmark (2)	8.3/0.4	-	-	-	-	80
Questionmark (3)	3.1/0.2	-	-	-	-	80
Grampolo (1)	17.1/3.7	-	-	-	-	102
Grampolo (2)	13.6/2.6	-	-	-	-	102
Grampolo (3)	2.7/0.4	-	-	-	-	102

Table 1: *Error statistics. Max/mean Euclidean distance (in mm) between vertices of the simulated poses and vertices of the input poses after initial actuation (1), optimization of actuation locations (2), and optimization of material distribution (3). The size corresponds to the maximum length of the character's bounding box.*

Model	#Elem.	#Actuation Points	Computation Time	Weights
TourEiffel	525	5	20min02s	1/-/-
Palmy2D	2745	4	5h26min	$0.01/10^{-5}/0.01$
Palmy3D	8362	4	7h17min	$0.1/10^{-7}/0.01$
WormEye	808	8	3h10min	$1.5\ 10^{-4}/10^{-6}/0.1$
Questionmark	4536	caps fixed	1h09min	$-/10^{-7}/1$
Grampolo	17709	7	4h44min	$500/10^{-3}/1$

Table 2: *Example statistics. From left to right: Number of elements, number of actuation points, total computation time, weights used for the regularization terms R_{sparse}, R_{soft} and R_{smooth}.*

exhibits the desired deformation behavior. We demonstrated our method on a set of simulated as well as physically-fabricated characters with different types of actuators and materials. We believe that our method is an important step toward physics-based design of real-world characters.

Limitations and Future Work We optimize for sets of actuation forces corresponding to equilibrium states that are as close as possible to the target poses. However, knowing the forces at the deformed state does not imply that there is a unique way of getting to that state from the undeformed configuration as there can be bifurcation points (buckling) along the way. While we did not encounter this sort of problems in our examples, further treatment might be necessary in order to ensure robust tracking of the input animation in between target poses.

As one possible direction for future work, we would like to explore the possibility of using a larger number or range of materials, e.g., by printing micro-level structures. An interesting related problem is also the design of more elaborate actuation systems that would enable us to animate more complex characters in an automated way.

Acknowledgments

We would like to thank the reviewers for their insightful comments. We also greatly appreciate the help of Christian Schumacher, Maurizio Nitti, and Ronnie Gänsli. This work was partly funded by the NCCR Co-Me of the Swiss NSF.

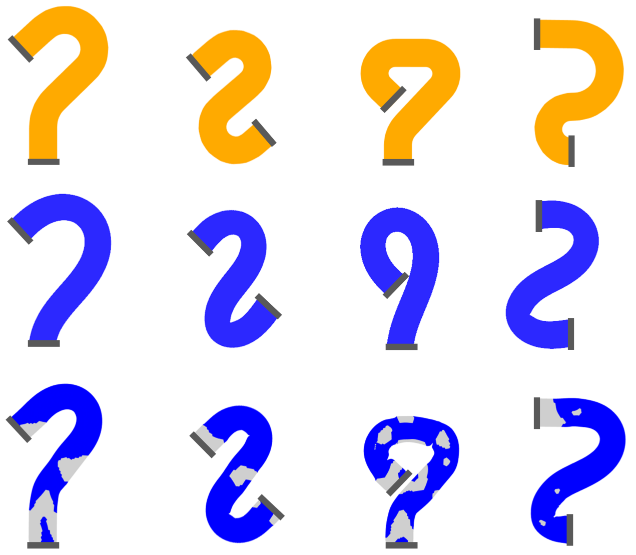

Figure 9: *A straight bar is deformed into four different question mark shapes by imposing position constraints on its end caps. The first row shows the target shapes, the second row the result without material optimization and the third our results after material optimization (stiff/soft material shown in grey/blue color).*

References

BÄCHER, M., BICKEL, B., JAMES, D. L., AND PFISTER, H. 2012. Fabricating articulated characters from skinned meshes. *ACM Trans. Graph. (Proc. SIGGRAPH) 31*, 4.

BENDSOE, M. P., AND SIGMUND, O. 2004. *Topology Optimization*. Springer.

BICKEL, B., BÄCHER, M., OTADUY, M. A., MATUSIK, W., PFISTER, H., AND GROSS, M. 2009. Capture and modeling of non-linear heterogeneous soft tissue. *ACM Trans. Graph. (Proc. SIGGRAPH) 28*, 3.

BICKEL, B., BÄCHER, M., OTADUY, M. A., LEE, H. R., PFISTER, H., GROSS, M., AND MATUSIK, W. 2010. Design and fabrication of materials with desired deformation behavior. *ACM Trans. Graph. (Proc. SIGGRAPH) 29*, 4.

BICKEL, B., KAUFMANN, P., SKOURAS, M., THOMASZEWSKI, B., BRADLEY, D., BEELER, T., JACKSON, P., MARSCHNER, S., MATUSIK, W., AND GROSS, M. 2012. Physical face cloning. *ACM Trans. Graph. (Proc. SIGGRAPH) 31*, 4.

CALÌ, J., CALIAN, D. A., AMATI, C., KLEINBERGER, R., STEED, A., KAUTZ, J., AND WEYRICH, T. 2012. 3d-printing of non-assembly, articulated models. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 31*, 6.

DONG, Y., WANG, J., PELLACINI, F., TONG, X., AND GUO, B. 2010. Fabricating spatially-varying subsurface scattering. *ACM Trans. Graph. (Proc. SIGGRAPH) 29*, 4.

HART, J. C., BAKER, B., AND MICHAELRAJ, J. 2003. Structural simulation of tree growth and response. *The Visual Computer*.

HASAN, M., FUCHS, M., MATUSIK, W., PFISTER, H., AND RUSINKIEWICZ, S. 2010. Physical reproduction of materials with specified subsurface scattering. *ACM Trans. Graph. (Proc. SIGGRAPH) 29*, 4.

HASLINGER, J., AND MÄKINEN, R. A. E. 2003. *Introduction to Shape Optimization*. SIAM.

LAU, M., OHGAWARA, A., MITANI, J., AND IGARASHI, T. 2011. Converting 3d furniture models to fabricatable parts and connectors. *ACM Trans. Graph. (Proc. SIGGRAPH) 30*, 4.

MALZBENDER, T., SAMADANI, R., SCHER, S., CRUME, A., DUNN, D., AND DAVIS, J. 2012. Printing reflectance functions. *ACM Trans. Graph. 31*, 3.

MARTIN, S., THOMASZEWSKI, B., GRINSPUN, E., AND GROSS, M. 2011. Example-based elastic materials. *ACM Trans. Graph. (Proc. SIGGRAPH) 30*, 4.

MCLAUGHLIN, T., CUTLER, L., AND COLEMAN, D. 2011. Character rigging, deformations, and simulations in film and game production. In *ACM SIGGRAPH 2011 Courses*.

MORI, Y., AND IGARASHI, T. 2007. Plushie: An interactive design system for plush toys. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3.

NARAIN, R., SAMII, A., AND O'BRIEN, J. F. 2012. Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 31*, 6.

NEALEN, A., MÜLLER, M., KEISER, R., BOXERMAN, E., AND CARLSON, M. 2006. Physically based deformable models in computer graphics. *Computer Graphics Forum 25*, 4.

NOCEDAL, J., AND WRIGHT, S. J. 2000. *Numerical Optimization*. Springer.

ÖZTIRELI, C., GUENNEBAUD, G., AND GROSS, M. 2009. Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum (Proc. Eurographics) 28*, 2.

ROZVANY, G. 2009. A critical review of established methods of structural topology optimization. *Structural and Multidisciplinary Optimization 37*, 3.

SKOURAS, M., THOMASZEWSKI, B., BICKEL, B., AND GROSS, M. 2012. Computational design of rubber balloons. *Computer Graphics Forum (Proc. Eurographics) 31*, 2.

SMITH, J., HODGINS, J. K., OPPENHEIM, I., AND WITKIN, A. 2002. Creating models of truss structures with optimization. *ACM Trans. Graph. (Proc. SIGGRAPH) 21*, 3.

STAVA, O., VANEK, J., BENES, B., CARR, N., AND MĚCH, R. 2012. Stress relief: improving structural strength of 3d printable objects. *ACM Trans. Graph. (Proc. SIGGRAPH) 31*, 4.

UMETANI, N., IGARASHI, T., AND MITRA, N. J. 2012. Guided exploration of physically valid shapes for furniture design. *ACM Trans. Graph. (Proc. SIGGRAPH) 31*, 4.

WEYRICH, T., PEERS, P., MATUSIK, W., AND RUSINKIEWICZ, S. 2009. Fabricating microgeometry for custom surface reflectance. *ACM Trans. Graph. (Proc. SIGGRAPH) 28*, 3.

WHITING, E., SHIN, H., WANG, R., OCHSENDORF, J., AND DURAND, F. 2012. Structural optimization of 3d masonry buildings. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 31*, 6.

XIN, S., LAI, C.-F., FU, C.-W., WONG, T.-T., HE, Y., AND COHEN-OR, D. 2011. Making burr puzzles from 3d models. *ACM Trans. Graph. (Proc. SIGGRAPH) 30*, 4.

ZHU, L., XU, W., SNYDER, J., LIU, Y., WANG, G., AND GUO, B. 2012. Motion-guided mechanical toy modeling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 31*, 6.