RoboCut: Hot-wire Cutting with Robot-controlled Flexible Rods

SIMON DUENSER, ETH Zurich
ROI PORANNE, University of Haifa and ETH Zurich
BERNHARD THOMASZEWSKI, Université de Montréal and ETH Zurich
STELIAN COROS, ETH Zurich

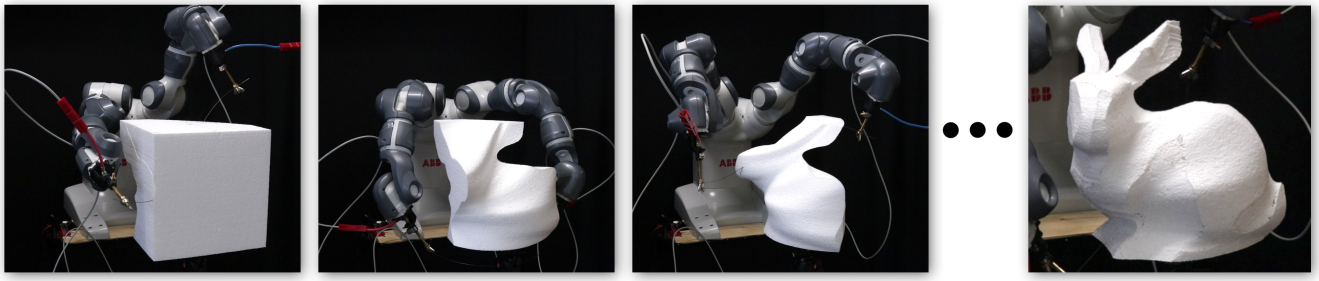


Fig. 1. Our method enables robotically-controlled hot wire cutting of complex shapes through tight integration of physical simulation, surface approximation, and path planning. Our algorithm anticipates and controls the deformations of the cutting rod to maximize the efficiency of every sweep, allowing this bunny shape to emerge after only 2 cuts, and be completed after 10.

Hot-wire cutting is a subtractive fabrication technique used to carve foam and similar materials. Conventional machines rely on straight wires and are thus limited to creating piecewise ruled surfaces. In this work, we propose a method that exploits a dual-arm robot setup to actively control the shape of a flexible, heated rod as it cuts through the material. While this setting offers great freedom of shape, using it effectively requires concurrent reasoning about three tightly coupled sub-problems: 1) modeling the way in which the shape of the rod and the surface it sweeps are governed by the robot's motions; 2) approximating a target shape through a sequence of surfaces swept by the equilibrium shape of an elastic rod; and 3) generating collision-free motion trajectories that lead the robot to create desired sweeps with the deformable tool. We present a computational framework for robotic hot wire cutting that addresses all three sub-problems in a unified manner. We evaluate our approach on a set of simulated results and physical artefacts generated with our robotic fabrication system.

CCS Concepts: • **Computer graphics → Computational geometry and object modeling**; *Physically based modeling*;

Additional Key Words and Phrases: Computer graphics, robotics, fabrication, sensitivity analysis

ACM Reference Format:
Simon Duenser, Roi Poranne, Bernhard Thomaszewski, and Stelian Coros. 2020. RoboCut: Hot-wire Cutting with Robot-controlled Flexible Rods. *ACM Trans. Graph.* 39, 4, Article 98 (July 2020), 15 pages. https://doi.org/10.1145/3386569.3392465

Authors' addresses: Simon Duenser, ETH Zurich, simon.duenser@inf.ethz.ch; Roi Poranne, University of Haifa , ETH Zurich; Bernhard Thomaszewski, Université de Montréal , ETH Zurich; Stelian Coros, ETH Zurich.

1 INTRODUCTION

The ability to use tools is a hallmark of intelligence and it has profoundly shaped the evolution of human culture. For example, carving—the process of sculpting away material from a workpiece in order to reveal a desired artefact—has been used to create art forms and functional objects since ancient times. Carving demands skillful manipulation of various tools and intuitive understanding of the physical interactions between the tools and the objects they are applied to. Fueled by technological advances, the craft of carving has grown into CNC milling and cutting processes that now enjoy widespread use. With the continued evolution of carving techniques as a long-term goal, in this paper we present a computational framework for robotic hot-wire cutting.

A hot-wire cutter is a tool used to carve polystyrene foam and other materials that melt or vaporize when subjected to a source of intense heat. The tool consists of a thin metal wire that is typically held under tension using a bow. When attached to an electrical power source, the wire heats up past the melting temperature of the material. The cutter can therefore create thin cuts through the workpiece without the need to engage in any physical contact.

Hot wire cutters can be easily mounted on robotic platforms, and together with milling tools, they are routinely employed in machine shops. As they are equipped with a *straight* wire, each cut generated by a typical hot wire cutter is a *ruled surface*. Generating efficient toolpaths for hot-wire cutting is thus a matter of approximating a desired shape using piecewise ruled surfaces—a topic of intense ongoing research. Nevertheless, finding a high quality approximation is only part of the challenge; another crucial problem in generating *feasible* toolpath trajectories is to ensure that the work-space of the fabrication machine and collision avoidance constraints are taken into account—a point we will return to shortly.

The straight-wire cutters that are typically used today prompted us to explore the following question: what if the wire could *bend* dynamically throughout each cut? To begin answering this question, we employ a YuMi® IRB 14000 robot as our hardware platform for hot-wire cutting. The robot has two arms, each with 7 degrees of freedom, and it holds the hot wire—an inextensible elastic metal rod—with its end effectors. The shape of the rod can therefore be actively controlled by adapting the position and orientation of the robot's grippers. This setup enables a much broader space of possible shapes to be cut. Nevertheless, the task of approximating a target shape using surfaces generated by sweeping a flexible rod is an open problem. In our setting, this problem is particularly interesting as the shape of the rod is governed by elasticity equations that capture its mechanical behavior.

In summary, hot wire cutting with robot-controlled flexible rods entails three tasks that are inseparably intertwined:

Physics-based modeling: determine how the shape of the rod and the surface it sweeps change with the robot's motions.

Surface approximation: approximate a target shape using a sequence of surfaces swept by an inextensible elastic rod.

Path planning: generate collision-free motion trajectories that enable the robot to create desired sweeps with the deformable tool.

In this paper, we present a toolpath generation technique for robotic hot wire cutting that accomplishes all three of the above tasks within a unified formulation. In particular, our method is able to anticipate and exploit the way in which the robot's pose affects the shape of the cutting rod at every moment in time. In combination with a dedicated distance measure, this ability enable us to maximize, directly as a function of the robot's motions, the degree to which the path swept by a deforming rod conforms to a given target shape. We generate a variety of simulated results and physical artefacts to evaluate the efficacy of our method.

2 RELATED WORK

Flexible hot-wire cutting involves concepts from different fields, ranging from the mathematics of surface approximation by swept curves to the simulation of elastic rods, and from motion planning and tool path generation to material science in general. To our best knowledge, the problem has not been addressed as a whole before. In particular, this is a first attempt to simultaneously solve surface approximation and tool path planning, considering multiple cuts and leveraging all degrees of freedom offered by a robot-controlled hot-wire cutter.

Surface Approximation. Path planning for conventional hot-wire cutting has close ties to computational geometry. Indeed, the surfaces generated by sweeping straight wires along spatial curves correspond exactly to the class of *ruled surfaces*, which have many applications in architecture [Pottmann and Wallner 2001]. The question of how to best approximate a given input shape by ruled surfaces is still the subject of ongoing research. For simple tensor-product patches, this problem can be solved using dynamic programming [Wang and Elber 2014]. For more general surfaces, [Flöry et al. 2013; Flöry and Pottmann 2010] decompose the target shape into strips based on asymptotic directions. These strips are then approximated by a set of ruled surfaces that minimize the quadratic distance from the input. While each of the strips could potentially be cut with a single sweep of a straight wire, the deviation from the initial shape can be significant. Our path planning algorithm controls the shape of the wire during cutting such as to best conform to a given surface region. In this way, our method is able to cut non-ruled patches with a single sweep, which leads to better surface approximation for a given number of cuts.

Apart from the particular class of ruled surfaces, approximating general shapes with simpler types of surfaces is a problem that has been studied extensively in computer graphics and computational geometry. For example, Cohen-Steiner et al. [2004] introduced a variational framework for approximating shapes with flat polygons. Chen et al. [2013] used a similar idea for shape fabrication using flat panels. Shape approximation by *developable* surfaces is another heavily investigated topic; see, e.g., the recent works by Stein et al. [2018] and Rabinovich et al. [2018]. While flat panels and developable surfaces have important applications in cost-efficient manufacturing, our method exploits the potential of flexible hot-wire cutting to efficiently approximate complex shapes with non-developable surfaces.

Surface Matching & Registration. As a core component of our method, we must quantify the efficiency of individual cuts, which requires a measure of distance between the surface swept by the cutting wire and the target shape. Since these surfaces do not exhibit an *a priori* correspondence, this problem is similar to partial surface registration. Standard registration is the process of aligning two data sets by finding a rigid transformation that takes one set, known as the *moving* set, as close as possible to the other, *fixed* set. Arguably the most widely used method to solve this problem is Iterative Closest Point (ICP) [Besl and McKay 1992; Chen and Medioni 1991], which splits the optimization into two alternating steps. The first step matches each point in one set to the point closest in the other set based on their Euclidean distance. With these correspondences, the second step then finds an optimal rigid transformation using, e.g. , the Procrustes method.

Many variations of ICP have been proposed. Some methods consider *robust* [Jian and Vemuri 2011] or sparse [Bouaziz et al. 2013] registration to deal with noisy or partial data for which not all points can be reliably matched. Non-rigid registration is another extension of ICP in which the moving data set is allowed to *deform.* This requires a deformation model, which can be either intrinsic [Li et al. 2008] or extrinsic [Myronenko and Song 2010] and must balance between matching and distortion. Our approach can also be seen as an instance of non-rigid registration, but the admissible deformations are restricted to the space of surfaces that can be swept by equilibrium configurations of a robot-controlled elastic rod.

Toolpath Generation. As a core component of our method, path planning for our dual-arm robot setup can be framed in the broader context of toolpath generation problems. In CNC applications, the tool is most commonly a drill bit and the objective is to find a path that creates the desired surface while respecting reach and speed constraints imposed by the machine. For example, Muntoni et al. [2018] decompose arbitrary shapes into axis-aligned pieces that can be milled with a 3-axis CNC machine in a single pass. Similarly, Zhao et al. [2018] decompose an input shape for a *3+2*-axis machine and use Fermat spirals for the toolpath. Similar concepts appear in the context of 3D-printing, where the last few years have seen increasing interest in using robot arms for, e.g., wireframe fabrication [Wu et al. 2017, 2016] and curved printing [Dai et al. 2018]. In contrast, in this paper we consider the challenges and opportunities that arise when using a deformable tool, whose shape is governed by the motions of the robotic system and the mechanics of flexible Kirchhoff rods.

Perhaps most closely related to our work is the effort by Søndergaard et al. [2016], who demonstrate the potential of flexible hot-wire cutting to produce doubly-curved surfaces with one single sweep. There are, however, two important differences between their method and the approach that we propose. First, Søndergaard et al. require the cutting blade to stay planar, which simplifies computations but restricts the space of admissible robot motions. Our approach imposes no such restrictions, allowing the robot to automatically discover complex motions that exploit non-planar wire configurations. Especially for complex models, this ability proves crucial for optimizing cut efficiency while avoiding collisions. As the second major difference, a sweep path normal to the cutting plane has to be prescribed by the user in [Søndergaard et al. 2016]. Furthermore, the input model has to be segmented into sufficiently simple patches that can be cut with a single sweep. Assuming this type of input simplifies surface approximation to a sequence of 2D curve approximation problems, but it critically relies on the user to solve nontrivial problems. While our approach supports manual guidance, it does not rely on any user input and can generate optimized cut sequences in a fully automatic way.

Rust et. al [2016a; 2016b] propose another approach to flexible hot-wire cutting based on a forward design approach. In their method, the user manually specifies a robot trajectory and the system displays a simulation-based preview of the result. Their simulation model accounts for contact between a slack, soft wire and the carving material. While we do not aim to simulate the physics of the cutting process itself, our method leverages *inverse* simulation to anticipate and control the shape of the wire during cutting.

3 THE METHOD

Our method enables robotically-controlled hot wire cutting of complex shapes by tightly integrating physical simulation, surface approximation, and path planning. Our physical setup, illustrated in Fig. 2, consists of a dual-armed robot controlling the shape of a flexible metal rod through the position and orientation of its two end-effectors. Given a target shape, our method generates a sequence of trajectories for the robot that sweep the rod through the workpiece to create a cut. The trajectories for each cut are optimized such that the rod moves and deforms in order to maximally decrease the difference between the current workpiece and the target shape.

Having introduced some basic concepts and definitions (Sec. 3.1), we start by describing the map between robot configuration and equilibrium shape of the cutting rod (Sec. 3.2). To quantify the efficiency of a given cut, we propose a dedicated metric based on robust partial surface matching (Sec. 3.3). Our optimization method leverages these two components to generate cuts with maximal efficiency (Sec. 3.4) while ensuring robustness of the wire against perturbations and inaccuracies (Sec. 3.5). To avoid undesirable solutions

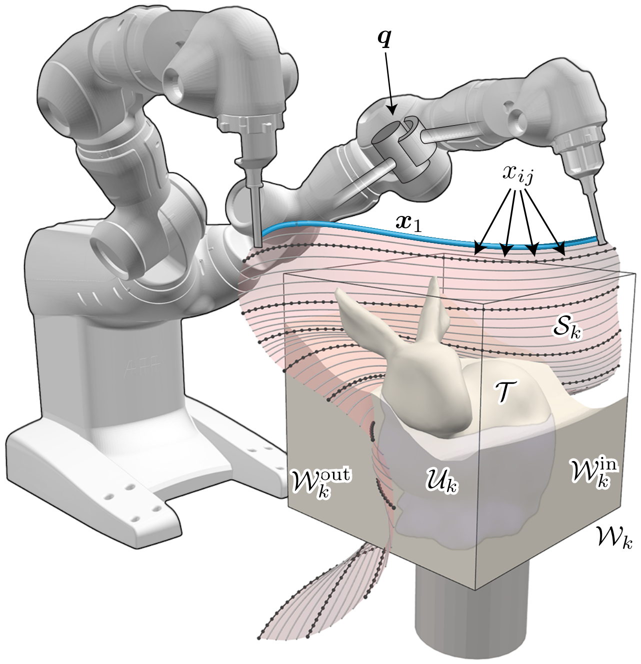


Fig. 2. An overview of the main components of our system, which takes as input a kinematic description of the robot and a target shape $\mathcal{T}$. The output are robot trajectories which generate toolsurfaces $\mathcal{S}_k$ that cut through the material. These cuts lead to a sequence of workpieces $\mathcal{W}_k$ that approximate the target shape with increasing accuracy.

corresponding to local minima, we additionally describe a continuation method that adaptively tightens convergence thresholds based on progress (Sec. 3.6). Finally, we integrate this cut optimization scheme into a sequential path planning tool that initializes cuts either in a fully-automated or user-controlled way (Sec. 3.7).

3.1 Toolsurfaces & Proper Cuts

Analogously to toolpaths, we refer to the surface generated by a given cut as the *toolsurface* $\mathcal{S}$. Each cut removes part of the workpiece $\mathcal{W}$, thereby reducing its distance to the target shape. Starting from an initial workpiece $\mathcal{W}_1$, the sequence of toolsurfaces $\mathcal{S}_k$ results in a sequence of workpieces $\mathcal{W}_k$, where $\mathcal{W}_{k-1} \supset \mathcal{W}_k \supset \mathcal{T}$. Each $\mathcal{S}_k$ divides the current $\mathcal{W}_k$ into two volumes: the one that remains, $\mathcal{W}_k^{\text{in}}$, and the one that is removed, $\mathcal{W}_k^{\text{out}}$ (see Fig. 2). Applying $\mathcal{S}_k$ to $\mathcal{W}_k$ thus results in a more refined workpiece $\mathcal{W}_{k+1} = \mathcal{W}_k^{\text{in}}$, which is then used as input for computing $\mathcal{S}_{k+1}$.

Discretization. In the discrete setting, a toolsurface is defined by a set of piecewise linear curves stacked in time, each corresponding to an equilibrium state of the cutting wire. Given robot joint angles $\boldsymbol{q}_i$, the corresponding equilibrium shape $\boldsymbol{x}_i$ of the rod is found by minimizing its elastic energy subject to the boundary conditions imposed by the robot (Sec. 3.2). We assume that the rod is quasi-inextensible, and that the robot moves slowly enough such that the motions are quasistatic. The joint trajectories are linearly discretized into timesteps $\boldsymbol{q}_i$ where $i = 1, \ldots, n_k$. Each $\boldsymbol{q}_i$ produces a different

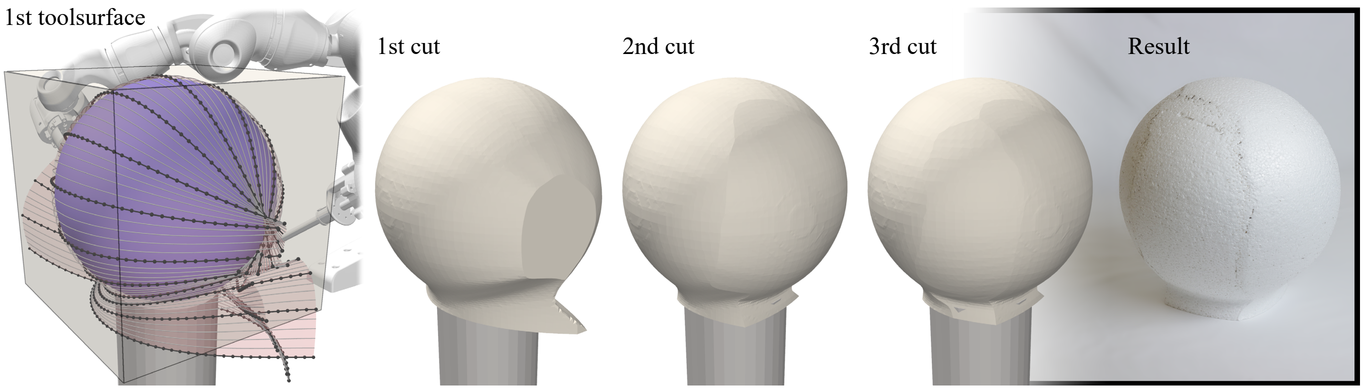


Fig. 3. Cutting a sphere. The first toolsurface shown to the left already covers most of the sphere. The second cut removes almost all of the remaining material, while the result of the third cut is barely distinguishable from the second one.

wire configuration $\boldsymbol{x}_i$ defined by nodal positions $x_{ij} \in \mathbb{R}^3$. We obtain $\mathcal{S}$ by linearly interpolating between subsequent $\boldsymbol{x}_i$.

Proper Cuts. We want each cut to maximize the newly uncovered surface area, i.e., the part of the toolsurface whose distance to the target shape is within a user-defined tolerance. However, the reward generated by a given cut is not a continuous function, it changes abruptly once a chunk of material can be removed, i.e., when the wire exits the workpiece. We therefore define a proper cut as a toolsurface that satisfies $(\boldsymbol{x}_1 \cup \boldsymbol{x}_n) \cap \mathcal{W} = \emptyset$ and $(\bigcup_{i=2}^{n-1} \boldsymbol{x}_i) \cap \mathcal{W} \neq \emptyset$, i.e., that starts and ends entirely outside of the workpiece.

3.2 Robot-Controlled Rod Deformations

Anticipating and controlling the deformations that the robot induces in the cutting rod is at the heart of our method. To this end, we use the elastic rod model of Bergou et al. [2010] and determine the shape of the cutting rod as a function of the robot configuration $\boldsymbol{q}$. This model represents twisting and bending of the wire by defining a *material frame* along its piecewise linear centerline $\boldsymbol{x}_i$, whose edges are equipped with additional angle variables $\boldsymbol{\alpha}_i = (\alpha_{ij})$ that represent rotations of the cross section. The joint angles $\boldsymbol{q}_i$ of the robot determine the position and orientation of its end-effectors which, in turn, impose boundary conditions on $\boldsymbol{x}_i$. In practice, we enforce these boundary conditions using soft constraints on x_{i1}, x_{i2}, and the material frame normal n_{i1} on one end point, and analogously for $x_{i,m-1}, x_{im}$ and $n_{i,m-1}$ on the other end point. The positions and orientations of the end-effectors are computed using forward kinematics. Specifically, let $x_{il}^{\text{local}}, l = 1, 2, m-1, m$, and $n_{il}^{\text{local}}, l = 1, m-1$ be the locations and the material frame normal of the attachment points in the end-effector's local coordinate system. The soft boundary constraint is then defined as

$$E_{\text{bound}}(\boldsymbol{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{q}_i) = \sum_{l=1,2,m-1,m} \|x_{il} - \mathcal{K}(x_{il}^{\text{local}}, \boldsymbol{q}_i)\|^2 + \sum_{l=1,m-1} \angle(n_{il}(\boldsymbol{x}_i, \boldsymbol{\alpha}_i), \mathcal{K}(n_{il}^{\text{local}}, \boldsymbol{q}_i))^2 , \tag{1}$$

where $\mathcal{K}(\cdot,\cdot)$ is the mapping from local to global coordinates, and $\angle(\cdot,\cdot)$ is the angle between two vectors.

We determine equilibrium shapes for the wire by minimizing

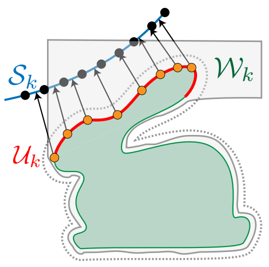
$$E_{\text{wire}}(\boldsymbol{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{q}_i) = E_{\text{stretch}}(\boldsymbol{x}_i) + E_{\text{bend}}(\boldsymbol{x}_i, \boldsymbol{\alpha}_i) + E_{\text{twist}}(\boldsymbol{x}_i, \boldsymbol{\alpha}_i) + w_{\text{bound}} E_{\text{bound}}(\boldsymbol{x}_i, \boldsymbol{q}_i) , \tag{2}$$

where $E_{\text{stretch}}, E_{\text{bend}}$ and E_{twist} denote energy terms corresponding to stretching, bending and twisting as defined in [Bergou et al. 2010], and w_{bound} is a scaling parameter for the penalty term E_{bound}. Note that, since the wire is sufficiently stiff and lightweight, we do not include gravity.

3.3 Surface Matching

Our approach aims to optimize the efficiency of each cut, which requires some measure of difference between a given toolsurface $\mathcal{S}_{k+1}$ and the target shape $\mathcal{T}$ that accounts for the current workpiece $\mathcal{W}_k$. As the toolsurface covers the target shape only partially and in an approximate way, this setting bears some similarity to the problem of non-rigid partial surface registration. However, as a crucial difference to standard metrics, parts of a cut that are not sufficiently close to the target shape should receive zero reward, since the corresponding surface region will have to be cut again.

In designing a metric that reflects this aspect, we draw inspiration from Bouaziz et al. [2013], who replace the standard Euclidean norm with a sparsity-inducing L_p-norm where $0 < p < 1$ to obtain improved robustness against outliers. As a starting point for such a metric, we first have to establish correspondence between the target shape $\mathcal{T}$ and its partial approximation $\mathcal{S}$.



Correspondence. Let $\mathcal{U}_k$ be the uncut surface part of $\mathcal{T}$ after $k-1$ cuts. The first step is to find corresponding points between the wire nodes $\boldsymbol{x} = \{x_{ij}\}$ that define the toolsurface $\mathcal{S}_k$ and $\mathcal{U}_k$. A straightforward approach would be to match each vertex x_{ij} with its closest counterpart on $\mathcal{U}_k$. However, since nothing prevents several wire configurations $\boldsymbol{x}_i$ from taking on the same shape, this strategy runs the risk of making the toolsurface collapse to a single curve. We therefore switch roles and have points on the target surface attract their closest counterparts on the toolsurface.

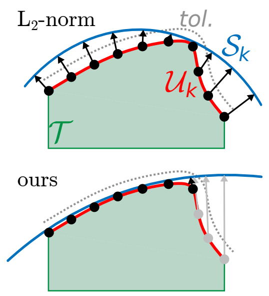
To this end, we start by triangulating $\mathcal{S}_k$ and uniformly sample $\mathcal{U}_k$. Denoting the samples on $\mathcal{U}_k$ by $\{u_l\}_{l=1}^{n_s}$, we find for each u_l its closest point on $\mathcal{S}_k$, which we denote by $e_l(\boldsymbol{x})$. This point can be either a vertex, or it can lie on an edge or within a triangle.

Distance Measure. Having established correspondence, we define the distance between $\mathcal{S}_k$ and $\mathcal{U}_k$ as

$$E_{\text{dist}}(\boldsymbol{x}) = \sum_{l=1}^{n_s} f_{\text{p}}(\text{d}(e_l(\mathbf{x}), u_l))\, A_l \,, \tag{3}$$

where $f_{\text{dist}}(\cdot)$ models a generalized L_p-norm, $\text{d}(\cdot,\cdot)$ is the Euclidean distance, and A_l is the surface area pertaining to sample u_l.

The choice for f_{p} has a large impact on the quality of the cut. As illustrated in the inset figure, the standard L_2-norm is inadequate in our setting since minimizing the average distance can lead to large parts of the cut lying outside the tolerance. An alternative approach would be to use an L_p-norm with $0 < p \leq 1$ that penalizes larger distances less severely. However, while somewhat mitigated, we observed in our experiments that large sample distances still have a significant impact and ultimately deteriorate the efficiency of the cut. Ideally, we would like to have as many samples u_l as possible within the tolerance. The exact distance is not important as long as it is less than a given tolerance a. Likewise, a sample whose distance is larger than a should be assigned the same cost, since it has to be uncovered by another cut, irrespective of its actual distance. These observations motivate an L_0-like metric that counts the number of samples in $\mathcal{U}_k$ whose distance is larger than a. Directly using the L_0-norm would, however, lead to a combinatorial problem that is prohibitively expensive to solve.



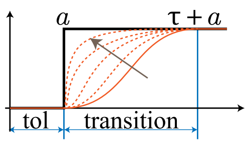
To remain within the continuous setting, we define a sequence of smoothed step-functions

$$H_{a\tau}(t) = \begin{cases} 0 & t \leq a \\ 3\left(\frac{t-a}{\tau}\right)^2 - 2\left(\frac{t-a}{\tau}\right)^3 & a < t < a+\tau \\ 1 & t \geq a+\tau \,. \end{cases} \tag{4}$$

As $\tau \to 0$, this function yields the desired L_0-like behavior when applied to the Euclidean distance, i.e.,

$$E_{\text{dist}}(\boldsymbol{x}) = \sum_l H_a(\text{d}(e_l(\mathbf{x}), u_l)), \quad \text{with } H_a(t) = \begin{cases} 0 & t < a \\ 1 & t \geq a \,. \end{cases} \tag{5}$$

It should be noted that every function in the one-parameter family (4) is a cubic spline that smoothly transitions from 0 to 1 over a span of length τ (see inset). During optimization, we use a continuation method [Allgower and Georg 2003; Poranne et al. 2017] that gradually decrease τ, thus tightening the transition region such that $H_{a\tau} \to H_a$. We discuss our strategy for decreasing τ in Sec 3.6.



Penetration of the Target Shape. The above distance measure is symmetric as it does not discriminate between negative and positive distance. The real-world setting is, however, asymmetric: while cuts too far from the target shape can always be improved, cuts into the target shape cannot be compensated. Consequently, we must prevent cutting into the target shape by more than the user-provided tolerance. To this end, we complement our symmetric distance measure with a unilateral barrier function that strongly penalizes cuts below the admissible depth. The exact definition is given by (28) in Appx. B.

3.4 The Optimization Problem

The goal of our optimization method is to find *feasible* wire shapes $\boldsymbol{x}_i$ that minimize the surface matching objective. In order for a given $\boldsymbol{x}_i$ to be feasible, it needs to be a stable equilibrium configuration with respect to the boundary conditions imposed by the robot configuration $\boldsymbol{q}_i$, i.e.,

$$G(\boldsymbol{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{q}_i) := \frac{\partial}{\partial \boldsymbol{y}_i} E_{\text{wire}}(\boldsymbol{y}_i, \boldsymbol{q}_i) = \mathbf{0} \quad \text{and}$$

$$\mathbf{H}_{\text{wire}}(\boldsymbol{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{q}_i) := \frac{\partial^2}{\partial \boldsymbol{y}_i^2} E_{\text{wire}}(\boldsymbol{y}_i, \boldsymbol{q}_i) \succ 0 \,,$$

where we summarized position and angle variables as $\boldsymbol{y}_i = [\boldsymbol{x}_i^T, \boldsymbol{\alpha}_i^T]^T$. With these constraints introduced, the optimization problem can now be formulated as

$$\min_{\boldsymbol{q},\boldsymbol{y}} \quad O(\boldsymbol{q}, \boldsymbol{y}) = E_{\text{dist}}(\boldsymbol{y}) + E_{\text{reg}}(\boldsymbol{y}, \boldsymbol{q}), \tag{6a}$$

$$\text{s.t.} \quad G(\boldsymbol{y}_i, \boldsymbol{q}_i) = 0, \forall \boldsymbol{y}_i, \boldsymbol{q}_i \tag{6b}$$

$$\mathbf{H}_{\text{wire}}(\boldsymbol{y}_i, \boldsymbol{q}_i) \succ 0, \forall \boldsymbol{y}_i, \boldsymbol{q}_i \tag{6c}$$

where $E_{\text{reg}}(\boldsymbol{y}, \boldsymbol{q})$ is a regularization term that we discuss in Sec. 3.5, and $\boldsymbol{y} = (\boldsymbol{y}_i)$ Note that the first of the above expressions asks for stationarity, which is a necessary but not sufficient condition for stability. The second condition requires the Hessian to be positive definite, thus restricting feasible $\boldsymbol{y}_i$ to stable local minima.

The challenge of solving (6) stems mainly from (6b), which captures the highly non-linear relationship between the robot pose and the equilibrium shape of the cutting rod. We therefore use the implicit function theorem to eliminate the constraints and instead solve the unconstrained problem

$$\min_{\boldsymbol{q}} \quad O(\boldsymbol{y}(\boldsymbol{q}), \boldsymbol{q}) \,. \tag{7}$$

Note that the only variables of this problem are the robot configurations $\boldsymbol{q}$, which implicitly determine the wire configurations $\boldsymbol{y}$. The derivative of the objective O is therefore

$$\frac{\text{d}O}{\text{d}\boldsymbol{q}} = \frac{\partial O}{\partial \boldsymbol{y}} \frac{\text{d}\boldsymbol{y}}{\text{d}\boldsymbol{q}} + \frac{\partial O}{\partial \boldsymbol{q}} \,. \tag{8}$$

As derived in full detail in Appx. A, we use sensitivity analysis to analytically compute $\frac{\text{d}\boldsymbol{y}}{\text{d}\boldsymbol{q}}$ as

$$\text{S} := \frac{\text{d}\boldsymbol{y}}{\text{d}\boldsymbol{q}} = -\left(\frac{\partial G}{\partial \boldsymbol{y}}\right)^{-1} \frac{\partial G}{\partial \boldsymbol{q}} \tag{9}$$

and a generalized Gauss-Newton Hessian approximation of O,

$$\mathbf{H} = \mathbf{S}^T \frac{\partial^2 O}{\partial \boldsymbol{y}^2} \mathbf{S} + 2\mathbf{S}^T \frac{\partial^2 O}{\partial \boldsymbol{y} \partial \boldsymbol{q}} + \frac{\partial^2 O}{\partial \boldsymbol{q}^2} \, . \tag{10}$$

To minimize the objective, we use these derivatives within a standard Quasi-Newton method augmented with line search. For each evaluation of $O(\boldsymbol{y}(\boldsymbol{q}), \boldsymbol{q})$ and its derivatives, we first compute $\boldsymbol{y}(\boldsymbol{q})$ by solving a separate minimization problem. While this yields an equilibrium configuration for every admissible robot state, not all equilibria are equally desirable as explained next.

3.5 Stability and Robustness of Wire States

Stability. We compute equilibrium states for the wire using Newton's Method with Levenberg-Marquardt-type regularization. However, while (6b) is a necessary condition for stability, it is not sufficient since stationary configurations could stem from a local energy maximum or a saddle point. To ensure that the wire is indeed stable, (6c) must also be satisfied. We guarantee this by adapting our minimization strategy as follows: whenever the algorithm has converged to an equilibrium point, we test whether the Hessian is indefinite. If so, we perturb the solution in the direction of the eigenvector associated with the most negative eigenvalue and proceed with the minimization. It should be stressed that the algorithm cannot ascend back to the pre-perturbation state—thanks to its line search and regularization strategies, it is forced to descend toward a stable local minimum.

Robustness. To achieve reliable and accurate cutting, two more considerations regarding the equilibrium state of the wire play an important role: (1) its stability with respect to external disturbances, and (2) its sensitivity with respect to inaccuracies in the boundary conditions induced by the robot. The first aspect can be quantified by considering the smallest eigenvalue of the Hessian, $\lambda_{\min}(\mathbf{H}_{\text{wire}})$, which represents the resistance of the system to external forces in its weakest direction. Similarly, the sensitivity matrix S in (9) expresses how deviations of the robot pose translate into changes in wire shape, and its largest singular value $\sigma_{\max}(\mathbf{S})$ quantifies the worst possible amplification of such inaccuracies.

A natural strategy for avoiding these problems would be to impose bounds of the form $\lambda_{\min} > a_\lambda$ and $\sigma_{\max} < a_\sigma$ that are enforced as part of the optimization problem. However, directly applying these bounds would require tracking eigenvalues and their derivatives which is computationally expensive and numerically unstable. We therefore seek alternative characteristics of the wire state that allow us to identify non-robust and too-sensitive configurations in an efficient way. We find such an attribute in the angle spanned by the tangents of the wire ends,

$$\gamma_i = \angle(x_{i,2} - x_{i,1}, x_{i,m-1} - x_{i,m}) \, . \tag{11}$$

In Appx. C we empirically show that, considering the constraints of our particular problem, a bound a_γ can be found such that

$$\lambda_{\min}(\boldsymbol{y}) > a_\lambda, \; \sigma_{\max}(\boldsymbol{y}) < a_\sigma \quad \forall \boldsymbol{y} \in \{\boldsymbol{y} \mid \gamma_i > a_\gamma\} \, , \tag{12}$$

for reasonable choices of bounds a_λ and a_σ. We therefore consider wire configurations as robust if the tangents at the wire's ends are sufficiently non-parallel. While there may exist sufficiently robust configurations that do not satisfy this condition, our extensive numerical experiments suggest that the reverse is not true, i.e., (12) is a sufficient, though not necessary, condition for robustness. We use this finding to construct an objective which enforces wire states that are robust with respect to external forces and inaccuracies in the robot's pose. While we have not observed a negative impact on shape approximation quality, this objective can make the difference between success and failure, as we demonstrate on a dedicated example in Sec. 4.2.

Additional Objectives. In addition to surface matching and wire stability, we include several other objectives in our formulation that promote smooth solutions, prevent collisions between robot and workpiece, enforce workspace constraints, and safeguard against plastic deformations of the wire. All of these objectives are added to $E_{\text{reg}}(\boldsymbol{x}, \boldsymbol{q})$ in Eq. (6a). Detailed definitions are provided in Appx. B.

3.6 Continuation Schedule

We cast optimal path planning as an unconstrained minimization problem with judiciously designed objectives. Nevertheless, several of these objectives are strongly nonlinear and non-convex, which can lead to ill-conditioning and other numerical problems that make robust optimization challenging. In particular, we observed that it is often difficult to select parameters such as the coefficients for distance objective and constraint penalties: too aggressive values can attract the solver to undesirable local minima, while too conservative values lead to less accurate surface approximation and constraint satisfaction. Drawing inspiration from recent work in geometric optimization [Poranne et al. 2017; Stein et al. 2018], we instead turn to a continuation strategy that adjusts these coefficients—primarily the transition length τ of the distance function (4)—based on progress. Starting with conservative choices, we gradually increase tightness whenever progress—as measured by the difference $\rho_j = O_j - O_{j+1}$ in objective values between subsequent iterations—falls below a given threshold $\rho_{\min}$, which is the only remaining parameter in this continuation method. We provide an experimental analysis of the impact of this threshold on the solution of the optimization problem in Sec. 4.4. In practice, this strategy allows us to reliably converge to good final results even for far off initializations.

3.7 Sequential Cut Planning

The optimization algorithm described above is integrated in a sequential planning approach that, given a rough path as initialization, maximizes the efficiency of each cut. This initialization is generated from a set of simple rules and heuristics, allowing for a fully automatic computation of the entire cutting sequence without any need for user input. If desired, initializations can also be defined interactively by the user, offering a means of guiding the length of a cut and its general placement. We briefly describe these two approaches in the following.

Automatic Cut Initialization. Each cut is initialized based on an automatically generated offset curve on the target shape. This offset curve is created by intersecting the target shape with a cutting plane whose orientation is chosen at random. After offsetting and resampling, each point on the curve is assigned a score reflecting

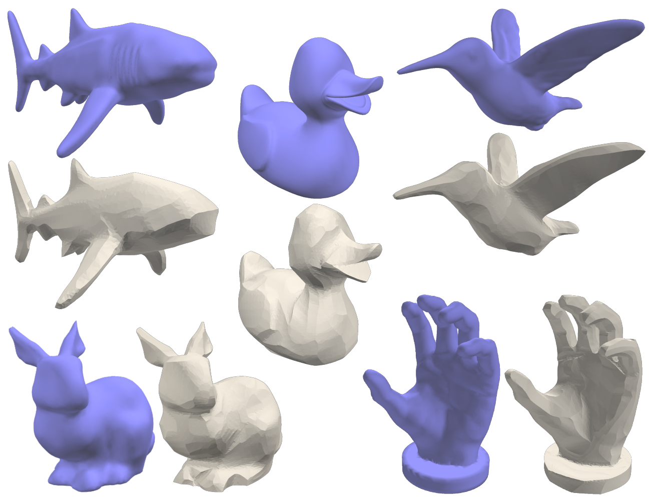


Fig. 4. Simulated results obtained through optimization with automatic initialization (*gray*) and their corresponding target shapes (*blue*).

their position with respect to the uncut regions of the target surface. We create a set of candidate curves by sampling cutting planes with different orientations and select the one that maximize the sum of per-point scores. Several examples produced with this fully automatic procedure are shown in Fig. 4, demonstrating the versatility and reliability of this approach even for fairly complex shapes.

User Guidance. In addition to automatic cut initialization, our system also offers two ways for manually guiding the aesthetics of the cut surface. First, the user can provide the initialization path for each cut and, if desired, prescribe the orientation of the workpiece. In this way, the user can effectively control the general region in which a cut will be made as well as its approximate length. Second, the user can restrict the optimization to consider only sub-regions of the target surface for matching, thus narrowing down the placement of the cut. We provide several examples generated with this type of user guidance in Sec. 4.4.

4 RESULTS

We evaluate our method on a diverse set of simulation examples as well as several physical prototypes. Before we present and analyze these example, we briefly describe our compute and manufacturing setup.

4.1 Hardware Setup

Our physical setup consists of a YuMi® IRB 14000, a robot with two 7-DoF arms. Its working range is about 660mm for each arm. In front of it we place a cubic block of polysterene foam with an edge length of 290mm. For all examples shown in this section, the workpiece was mounted on a tripod and manually rotated between cuts. Using an actuated mount is a straightforward extension that, on the software side, only requires adding another degree of freedom to the robot's joint angles. The wire with a diameter of 1mm is connected to a power source of 6.6 Amperes and 4.6 Volts, heating it to a temperature of around 250℃. At this temperature, the optimal velocity of a point on the wire was found to be around 2mm/s, which ensures melting of the material just before contact. For a simple, one-cut example like the *variable curvature* (Fig. 7) this results in a pure cutting time of about 3min, while the *bunny* requires 46min. Higher cutting speeds may be achieved with an increased temperature, though potentially at the cost of a reduced cut quality and a lowered yield stress of the wire. However, we did not investigate the fabrication process in more detail.

Cutting. Once the trajectories for all cuts have been computed, they are transferred to the robot for execution. Before a trajectory is transmitted, the duration between two subsequent robot poses is set such that the maximum cutting velocity of the wire equals a target value of 2mm/s. In between cuts, the user may need to reorient the workpiece to match the orientation specified in the application. Although not strictly necessary, we remove excess material after each cut to provide more room for the robot arms to maneuver. We refer the reader to the accompanying video which shows several actual cutting sequences.

4.2 Non-ruled Surfaces

Regular hot-wire cutting is limited to ruled surfaces, which have non-positive Gaussian curvature and thus cannot produce convex regions with a single cut. These limitations do not apply to our method, which we validate experimentally on a set of three examples.

Cutting a Sphere. As our first example to demonstrate non-zero Gaussian curvature, we cut the spherical shape shown in Fig. 3. After a single cut, 51% of the surface are already within the required tolerance of ±2mm, whereas the third cut barely removes any material. For comparison, the ideal strategy for a standard hot-wire cutter would be to perform three circular cuts that sweep out the intersection of three cylinders at an angle of 120°—a rather crude approximation of a sphere.

Cutting a Bowl. Another limitation of regular hot-wire cutting is that elliptical concave shapes such as a bowl cannot even be approximated. In contrast, our method generates trajectories for the robot arms that sweep the wire in a constant-curvature configuration, carving out the target shape with a single cut (see Fig. 6).

Variable Curvature. Besides constant curvature cuts, our method can also create surfaces with variable curvature in a single sweep. An example of such a case is shown in Fig. 7 and the top row of Fig. 8, where our method gracefully handles transition between positive and negative curvature. This is no simple feat, as zero curvature suggests a straight wire, which corresponds to an inherently problematic configuration: slight inaccuracies in the distance between the end-effectors can induce buckling. As described in Sec. 3.5, our criterion for robust equilibrium states causes the wire to bend on the target surface, allowing it to accurately navigate through this difficult terrain.

In Fig. 8 we demonstrate how wire shapes with sensitivity can deteriorate manufacturing accuracy. In the top row, a conservative upper bound on the maximum sensitivity was enforced during optimization, resulting in a maximum singular value of $\sigma_{\max} = 3.1$. The bottom row shows the results obtained when optimizing without

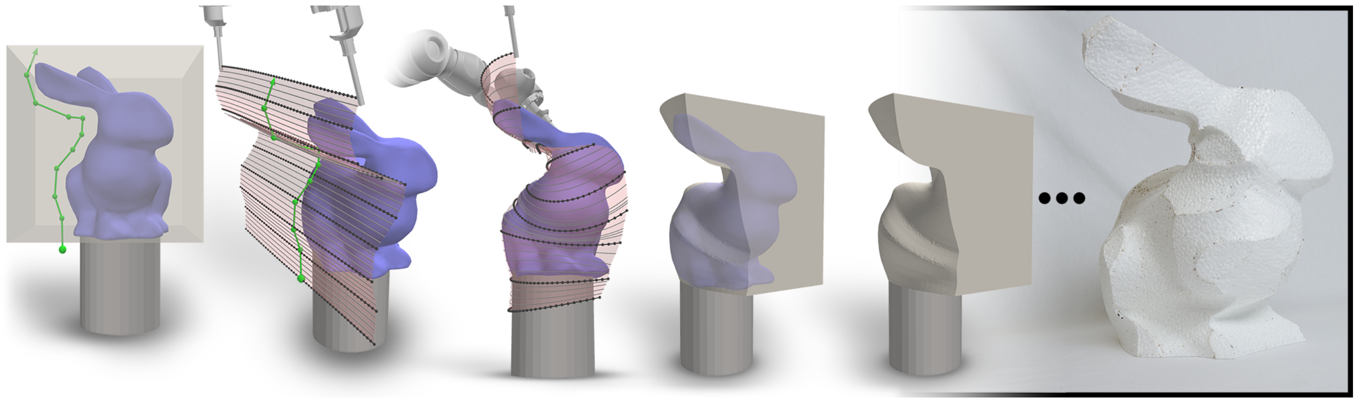


Fig. 5. Manual cut initialization. The user provides an initial path indicating where the cut should be made. The system then finds a valid trajectory and begins optimizing it. Once converged, the cut part is removed and the process repeats.

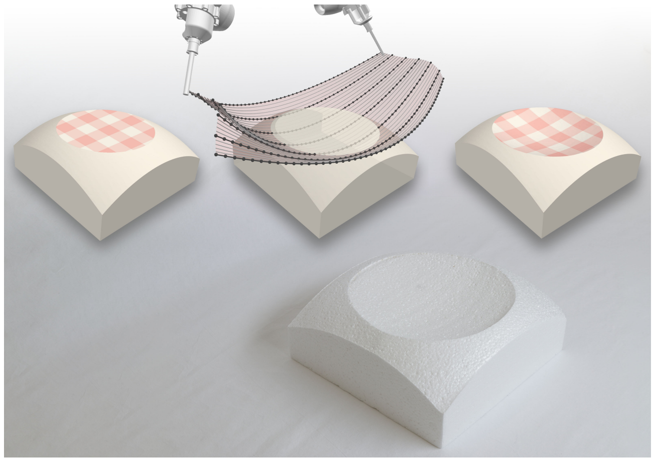


Fig. 6. Cutting a bowl. The flat region of the input shape (*top left*) is transformed into a concave spherical cap (*top right*), impossible to create with conventional hot-wire cutters. Our robotic fabrication system executes the optimized trajectories (*top middle*) to reveal the desired shape after a single cut (*bottom*).

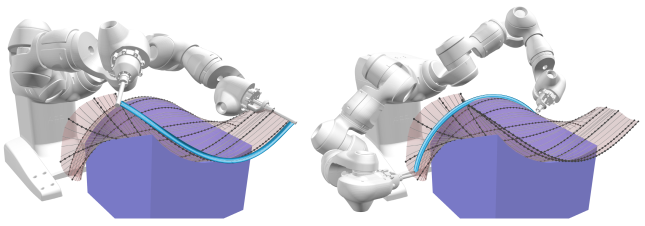


Fig. 7. Cutting a surface with sign-changing curvature.

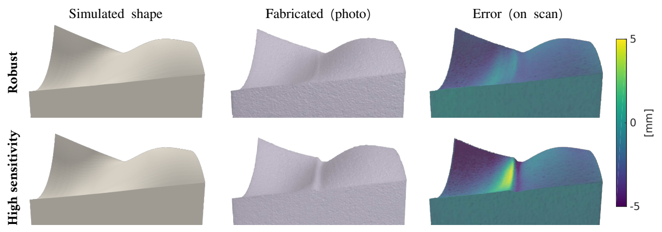


Fig. 8. Comparison between optimization results obtained with (*top*) and without (*bottom*) constraining the maximum singular value of the sensitivity matrix. Without this constraint, inaccuracies in the boundary conditions are strongly amplified, leading to severe artefacts in the manufactured model.

this constraint. In this case, the maximum sensitivity, which occurs at the transition between positive and negative curvature, reaches $\sigma_{\max} = 9.3$. It is interesting to note that the simulated results are virtually identical to the target shape in both cases. However, when cutting using the trajectories with high sensitivity, the wire exhibits significant, unpredicted buckling due to slight inaccuracies in the robot movement. This buckling, in turn, leads to large deviations of the resulting surface (Fig. 8, bottom right). In contrast, the robust solution leads to a significantly more accurate cut (8, top right).

Non-zero Genus. By curving the tool appropriately it is possible to produce objects with genus larger than zero, as we demonstrate with two examples of tori. However, accessibility to interior regions of such shapes is often limited, which may render parts of the surface unreachable. While e.g. the slender torus on the left of Fig. 9, which features a comparably large inner diameter, can be cut to a precision of $\pm 1mm$, this is not possible for the example on the right. Here the inner diameter is smaller, and the tool can not be bent tightly enough to reach the innermost portions of the target surface.

4.3 Complex Models

The above examples demonstrate the ability of our method to produce doubly-curved surfaces that are difficult, or even impossible, to achieve with conventional straight-wire cutting. The following examples show the potential of our approach to handle complex shapes in a fully-automated way.

The process of reproducing the Stanford bunny is depicted in Fig. 5. The relatively thin, but far protruding ears are particularly challenging as they limit the accessibility of the neck region. Our method nevertheless produces a visually pleasing rendition of this

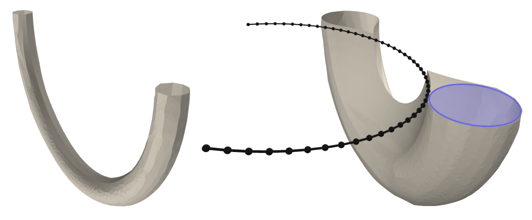


Fig. 9. Tori of different slenderness and inner diameter (view clipped to bottom half). Non-zero genus objects can be produced (left), though accessibility is often limited and it may not be possible to bend the tool tightly enough to reach all regions adequately (right).

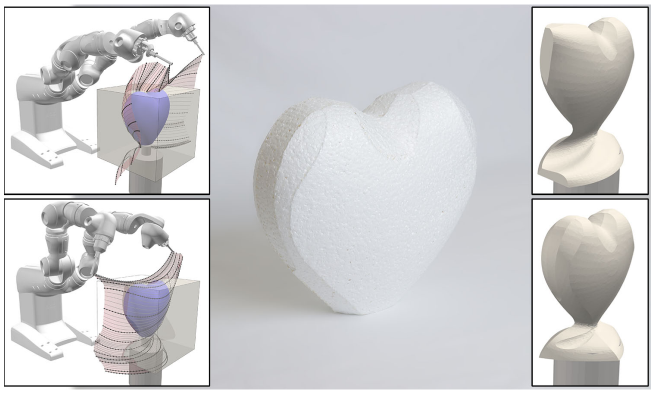


Fig. 10. Heart model. *Left*: toolsurfaces for the first and second cut. *Right*: simulated result after 6 cuts (*top*) and final result after 12 cuts (*bottom*). *Middle*: robotically manufactured result.

model with good approximation accuracy. See also Sec. 4.4 for a quantitative analysis.

In Fig. 10 we show the manufactured result of a heart model. The insets on the left show the first and second toolsurfaces, while the right insets show the result after 6 cuts, and the final outcome after 12 cuts.

Fig. 11 shows a prototype of an abstract arch made of four unique, individually-cut parts. This design exploits the ability of flexible hot-wire cutting to carve doubly-curved joining surfaces such as to create a stable interlocking configuration. Each part required four cuts, two for the top and bottom faces and two for the joining surfaces.

4.4 Analysis

Performance. We implemented our method in C++, using the Eigen library [Guennebaud et al. 2010] for matrix algebra and for solving the linear system in (8). For closest point searches we use spatial acceleration structures implemented in the libigl library [Jacobson et al. 2018]. Similarly, we rely on the fast winding number approximation [Barill et al. 2018] from libigl for inside-outside tests.

The most computation-intensive tasks—rod simulation, closest point search, and winding number computations—are parallelized with OpenMP. We found that on a typical cut each of these account for roughly 25% of the computational costs of the optimization.

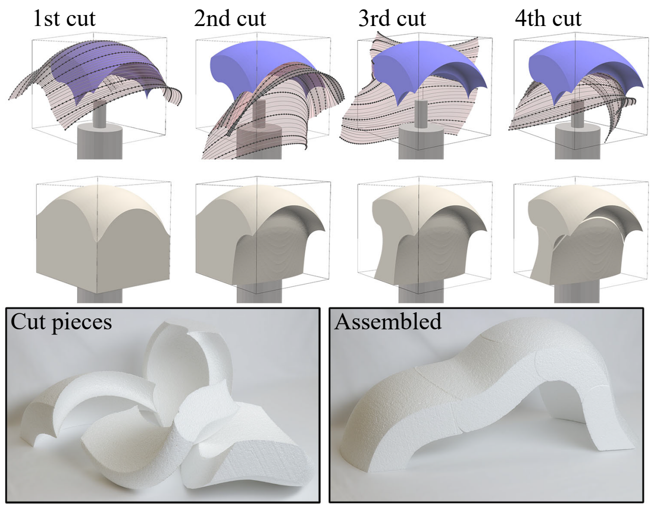


Fig. 11. Interlocking arch. *Top*: cutting sequence for one out of four unique pieces. *Bottom*: thanks to the doubly curved joining surfaces, the structure stands stably once assembled.

The computation time generally depends on the complexity of the object and the required accuracy of the result. A single, simple cut as, e.g., in the example *variable curvature* shown in Fig. 7 takes about 4*min* on a mobile workstation with an Intel® Xeon® E3-1505M processor. On the other end of the spectrum, the *hand* shown in Fig. 4, which is by far our most complex example and features several very long cuts, takes 4*h* to optimize. The *bunny*, when using automatic initialization, requires 2*h* 20*min*.

Continuation Schedule. The continuation method described in Sec. 3.6 increases the tightness of specific parameters of the optimization method (such as the transition length τ in (4)) whenever the decrease in objective falls below a given threshold. The value of this threshold influences the quality of each cut as well as the required computation time. Fig. 12 illustrates this effect on the first cut of the bunny model, plotting the final objective value—upon full convergence after the last step of the continuation schedule—as a function of the threshold value. While the objective functions are identical at this point, depending on the continuation threshold we obtain several distinctive local minima. We observe that for lower, more conservative values we generally obtain better results with lower objective values. Moreover, for a sufficiently low threshold, the optimization consistently converges to the same local minimum.

While this example alone does not allow for conclusions on the general case, it nevertheless allowed us to identify a threshold value that gave satisfying quality and performance for all examples presented in this work.

Accuracy of Optimization. The goal of the optimization procedure is to achieve a final shape that approximates the target up to a user-defined tolerance. This tolerance induces a symmetric band around the target shape, and the relevant objective functions are adjusted such that they force the cut surface to lie within this band. During optimization, the lower bound of the tolerance is of

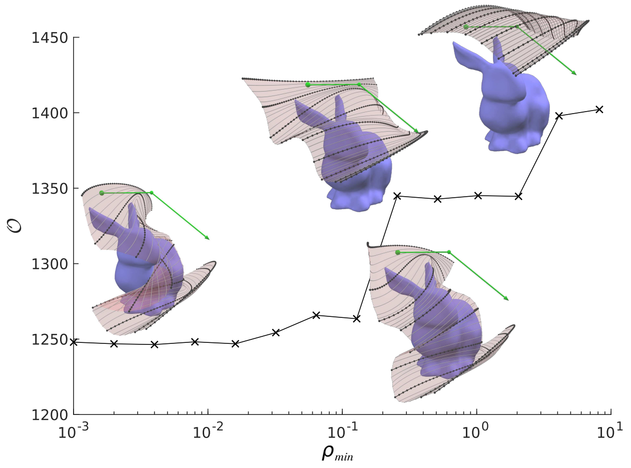


Fig. 12. Impact of the continuation schedule threshold on the final objective after full convergence. A large, aggressive value (*far right*) may cause the optimization to converge to an undesirable local minimum. Choosing a lower, more conservative value generally leads to better solutions. Below some minimum value, the optimization will converge to the same solution irrespective of the threshold value.

particular importance and must not be violated since too deep cuts cannot be undone. For this reason, we enforce the lower bound with a unilateral penalty function as described in Appx. B. The upper bound, on the other hand, does not need to be enforced, since any excess material can be removed by subsequent cuts.

The accuracy with respect to positive errors—towards the outside of the target shape—is mainly determined by the termination threshold in case of the fully-automated approach, or the number of cuts and their lengths, if chosen by the user. Nevertheless, some parts of the target shape may not be reachable without violating the lower bound on distance or the upper bound on admissible wire deformation—these regions will inevitably exhibit larger error.

These aspects can be observed on the example of the bunny in Fig. 13 (upper left). Close to 80% of the bunny are cut to within an error of $\pm 1mm$, and the deviation is strictly larger than $-2mm$, which is the lower bound on the admissible distance for this example. However, there are two distinct regions of larger deviation between the legs, which cannot be reached by the tool due to its limited deformability. Another smaller region right above the back leg is reachable in principle, but was left as is by the user in favour of a lower number of cuts. In contrast, the heart example depicted on the upper right of Fig. 13 was fully cut to a precision of $\pm 2mm$.

A summary of the accuracy for all of the produced examples can be found in Tab. 1.

Fabrication Accuracy. In addition to the above analysis on simulated examples, we also evaluated the accuracy of our method, and hardware setup, for the manufactured results. To this end, 3D models of the styrofoam objects were reconstructed using photogrammetry and registered to their corresponding simulated shape. In Fig. 13 (bottom) we show the resulting error for the bunny and the heart model. We find mean average errors of $1.97mm$ and $0.76mm$ respectively. Though we did not investigate the sources of these errors in further detail, we conjecture that the following factors are likely the major contributors: (1) The kerf width, i.e., the thickness of the cut is hard to predict and varies significantly with cutting speed and wire temperature, neither of which are constant during manufacturing. Our optimization assumed a uniform kerf width of $3mm$. (2) The robot, while boasting excellent repeatability, offers no guarantees in terms of absolute positioning accuracy. It further has a relatively low structural stiffness and was routinely operated above its nominal payload. (3) Although using a camera tripod to mount the workpiece offers great flexibility, it decidedly limits the accuracy to which the workpiece can be positioned, both with respect to the robot and to the tripod's rotation axis. Considering this fairly unoptimized hardware setup, we consider the measured accuracy to be well within expectations.

5 DISCUSSION

We presented a computational framework for robotic hot-wire cutting. In contrast to traditional approaches based on straight-wire cutting, we employed a dual-arm robot that has the ability to actively control the shape of an inextensible elastic rod during the cutting process. This setup enables efficient cutting of a broad space of shapes; however, using it effectively demands three tightly coupled sub-problems to be considered concurrently: 1) modeling the way in which the shape of the wire and the surface it sweeps are governed by the robot's motions; 2) approximating a target shape through a sequence of surfaces swept by an inextensible elastic rod; and 3) generating collision-free motion trajectories that enable the robot to create desired sweeps with the deformable tool. The optimization-based toolpath generation framework we presented addresses these sub-problems in a unified manner, and it has enabled us to generate a diverse array of simulated results and physical prototypes.

The optimization problem we formulate contains various constraints that need to be satisfied, in order, for example, to avoid collisions or penetration of the target shape. Our experiments show that one-sided quadratic penalty functions are sufficiently effective for this purpose. Penetration of the target shape by the tool was strictly smaller than the targeted tolerance of $2mm$ for all of the reported examples, and no robot collisions were observed. If an application demands strict constraint satisfaction, it is easy to apply logarithmic or inverse barrier functions instead. However, we find quadratic barriers to be quite beneficial, as they allow to initialize the problem in infeasible states.

Our initial investigations into this challenging problem domain highlight exciting avenues for future work. First, the problem of approximating a target shape with a sequence of surfaces swept by a deformable rod raises interesting theoretical questions in terms of convergence guarantees and optimal number of cuts. Such theoretical considerations need to be complemented by further investigations into the constraints imposed by the physical setup. For example, cutting away parts of the initial workpiece without attempting to match the input surface could prove to be a valuable strategy in reducing possible collisions between the robot and the block of material during subsequent cuts.

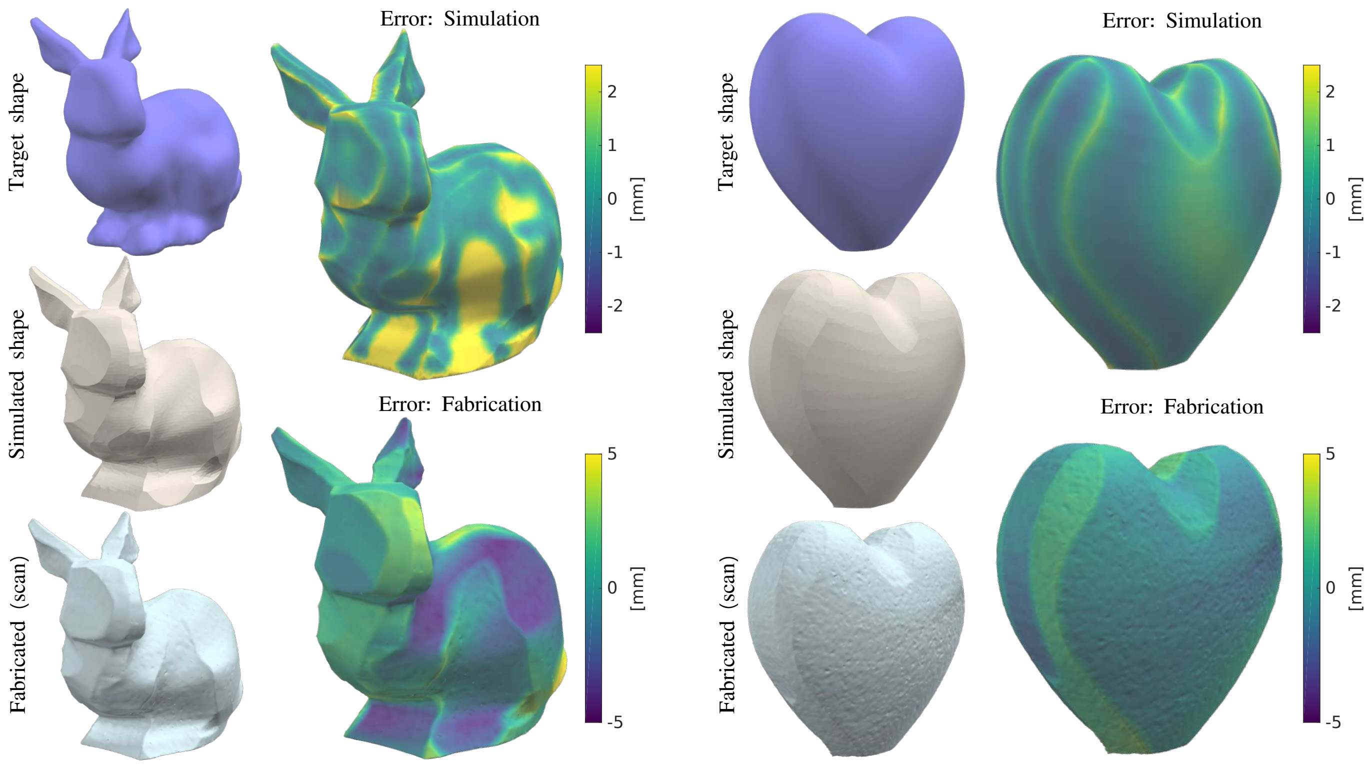


Fig. 13. Accuracy of the simulated final shape (*top*) and the fabricated shape (*bottom*) for the bunny and the heart model. The error of the simulated shape is measured as the distance to the target shape, whereas the error of the fabricated object is measured with respect to the simulated shape. A negative sign indicates a point lying within the target or simulated shape, respectively. The scan of the manufactured model has been rigidly registered to the simulated shape by minimizing the distance in a least squares sense. Both models occupy a bounding box with an edge length of $290mm$.

Table 1. Overview of the simulated objects. Objects that have also been fabricated are marked with *. The *total cut length* is defined as the sum of the maximum distances that the wire travels within the current workpiece $\mathcal{W}$ in each step. This measure defines the duration of the cutting process. *Cut surface* specifies the fraction of the target shape surface that lies within a distance of *tol* to the resulting simulated surface. *Mean absolute error* refers to the distance of the target shape to the resulting simulated surface. All objects were scaled to fit into a bounding box with a square base of $290mm$ edge length.

	Number of cuts	Total cut length	Cut surface tol $\pm 1mm$	Cut surface tol $\pm 2mm$	Mean absolute error	Initialization	Fig.
Sphere*	3	$2.44m$	87.9%	94.6%	$0.92mm$	manual	3
Bowl*	2	$0.49m$	99.9%	100%	$0.21mm$	manual	6
Variable curvature*	1	$0.37m$	99.4%	100%	$0.25mm$	manual	7
Torus slender	10	$5.20m$	100%	100%	$0.22mm$	automatic	9
Torus thick	7	$4.64m$	84.9%	89.5%	$1.10mm$	manual	9
Bunny*	10	$5.53m$	78.4%	87.1%	$1.10mm$	manual	5
Heart*	12	$3.17m$	94.2%	100%	$0.52mm$	manual	10
Arch part 1*	4	$0.97m$	99.9%	100%	$0.10mm$	manual	11
Arch part 2*	4	$1.33m$	99.9%	100%	$0.10mm$	manual	11
Arch part 3*	4	$1.45m$	99.8%	100%	$0.12mm$	manual	11
Arch part 4*	4	$1.17m$	100%	100%	$0.11mm$	manual	11
Sphere (auto init.)	3	$3.22m$	86.9%	98.1%	$0.50mm$	automatic	—
Bunny (auto init.)	8	$7.95m$	77.4%	89.4%	$1.03mm$	automatic	4
Rubber duck	7	$7.62m$	89.0%	98.5%	$0.53mm$	automatic	4
Shark	9	$6.35m$	84.2%	98.2%	$0.56mm$	automatic	4
Hummingbird	7	$6.87m$	81.0%	97.3%	$0.58mm$	automatic	4
Hand	8	$13.21m$	82.4%	96.9%	$0.61mm$	automatic	4

Another challenge that remains is the concise characterization of the space of cuttable shapes. We observe that the most limiting factor are collisions between tool and target shape, in combination with the limited shape space of the elastic wire. Locally, an arbitrarily small portion of a smooth surface can be matched well as long as at least one principal curvature is of lower magnitude than the bending limit of the wire. Regarding the second direction, there is no inherent limit. In the global context, however, the limited shape space of the elastic rod needs to be considered on top of the deformation constraint, and collisions of the tool may render some parts of an object entirely unreachable. This could be, for example, cavities like they are present between the legs of the bunny model (Fig. 13, left), but also the inside of a torus, when the inner diameter is too tight (Fig. 9, right). Though provided that all regions of the surface are reachable, even a torus or other non-zero genus shapes can be achieved (Fig. 9, left). As of now, our optimization procedure can provide a very strong indication as to whether a model can be cut adequately — though ultimately the call lies with the user. Finding formal, simplified criteria to anticipate the cuttability of a shape could not only streamline the fabrication planning, but also assist an informed design process.

Finally, the degree to which the wire can bend throughout the carving process is currently limited, as we explicitly aim to avoid plastic deformations. These could either be modeled and exploited during cutting, or wires that are pre-bent could be used instead: the robot could in principle choose the best tool from an input set for each cut. Moreover, mounting the workpiece on a robotic arm would eliminate the need to manually re-adjust the placement and orientation of the material before each cut, and it would drastically increase the effective workspace of the robotic setup. Last but not least, it will be interesting to develop computational techniques that will enable robots to use specialized carving tools such as hot knives and chisels to add high-frequency surface details to the shapes generated through hot-wire cutting.

ACKNOWLEDGMENTS

This work was supported in part by the Swiss National Science Foundation through the National Centre of Competence in Digital Fabrication (NCCR dfab).

REFERENCES

E. L. Allgower and Kurt Georg. 2003. *Introduction to Numerical Continuation Methods.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Gavin Barill, Neil G. Dickson, Ryan Schmidt, David I. W. Levin, and Alec Jacobson. 2018. Fast Winding Numbers for Soups and Clouds. *ACM Trans. Graph.* 37, 4, Article Article 43 (July 2018), 12 pages. https://doi.org/10.1145/3197517.3201337

Miklós Bergou, Basile Audoly, Etienne Vouga, Max Wardetzky, and Eitan Grinspun. 2010. Discrete Viscous Threads. In *ACM SIGGRAPH 2010 Papers (SIGGRAPH '10).* ACM, New York, NY, USA, Article 116, 10 pages. https://doi.org/10.1145/1833349.1778853

P. J. Besl and N. D. McKay. 1992. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14, 2 (Feb 1992), 239–256. https://doi.org/10.1109/34.121791

Sofien Bouaziz, Andrea Tagliasacchi, and Mark Pauly. 2013. Sparse Iterative Closest Point. *Computer Graphics Forum* 32, 5 (2013), 113–123. https://doi.org/10.1111/cgf.12178 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12178

Desai Chen, Pitchaya Sitthi-amorn, Justin T. Lan, and Wojciech Matusik. 2013. Computing and Fabricating Multiplanar Models. *Computer Graphics Forum* 32, 2pt3 (2013), 305–315. https://doi.org/10.1111/cgf.12050 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12050

Y. Chen and G. Medioni. 1991. Object modeling by registration of multiple range images. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation.* 2724–2729 vol.3. https://doi.org/10.1109/ROBOT.1991.132043

David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. 2004. Variational Shape Approximation. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 905–914. https://doi.org/10.1145/1015706.1015817

Chengkai Dai, Charlie C. L. Wang, Chenming Wu, Sylvain Lefebvre, Guoxin Fang, and Yong-Jin Liu. 2018. Support-free Volume Printing by Multi-axis Motion. *ACM Trans. Graph.* 37, 4, Article 134 (July 2018), 14 pages. https://doi.org/10.1145/3197517.3201342

Simon Flöry, Yukie Nagai, Florin Isvoranu, Helmut Pottmann, and Johannes Wallner. 2013. Ruled Free Forms. In *Advances in Architectural Geometry 2012*, Lars Hesselgren, Shrikant Sharma, Johannes Wallner, Niccolo Baldassini, Philippe Bompas, and Jacques Raynaud (Eds.). Springer Vienna, Vienna, 57–66.

Simon Flöry and Helmut Pottmann. 2010. Ruled Surfaces for Rationalization and Design in Architecture. In *LIFE in:formation. On Responsive Information and Variations in Architecture*, Aaron Sprecher, Shai Yeshayahu, and Pablo Lorenzo-Eiroa (Eds.). Association for Computer Aided Design in Architecture (ACADIA), 103–109. Proc. ACADIA 2010.

Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org.

Alec Jacobson, Daniele Panozzo, et al. 2018. libigl: A simple C++ geometry processing library. http://libigl.github.io/libigl/.

B. Jian and B. C. Vemuri. 2011. Robust Point Set Registration Using Gaussian Mixture Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 8 (Aug 2011), 1633–1645. https://doi.org/10.1109/TPAMI.2010.223

Hao Li, Robert W. Sumner, and Mark Pauly. 2008. Global Correspondence Optimization for Non-Rigid Registration of Depth Scans. *Computer Graphics Forum* 27, 5 (2008), 1421–1430. https://doi.org/10.1111/j.1467-8659.2008.01282.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2008.01282.x

Vittorio Megaro, Jonas Zehnder, Moritz Bächer, Stelian Coros, Markus Gross, and Bernhard Thomaszewski. 2017. A Computational Design Tool for Compliant Mechanisms. *ACM Trans. Graph.* 36, 4, Article Article 82 (July 2017), 12 pages. https://doi.org/10.1145/3072959.3073636

Alessandro Muntoni, Marco Livesu, Riccardo Scateni, Alla Sheffer, and Daniele Panozzo. 2018. Axis-Aligned Height-Field Block Decomposition of 3D Shapes. *ACM Trans. Graph.* 37, 5, Article 169 (Oct. 2018), 15 pages. https://doi.org/10.1145/3204458

A. Myronenko and X. Song. 2010. Point Set Registration: Coherent Point Drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 12 (Dec 2010), 2262–2275. https://doi.org/10.1109/TPAMI.2010.46

Roi Poranne, Marco Tarini, Sandro Huber, Daniele Panozzo, and Olga Sorkine-Hornung. 2017. Autocuts: Simultaneous Distortion and Cut Optimization for UV Mapping. *ACM Trans. Graph.* 36, 6, Article 215 (Nov. 2017), 11 pages. https://doi.org/10.1145/3130800.3130845

Helmut Pottmann and Johannes Wallner. 2001. *Computational Line Geometry.* Springer-Verlag, Berlin, Heidelberg.

Michael Rabinovich, Tim Hoffmann, and Olga Sorkine-Hornung. 2018. The Shape Space of Discrete Orthogonal Geodesic Nets. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)* 37, 6 (2018).

Romana Rust, Fabio Gramazio, and Matthias Kohler. 2016a. Force Adaptive Hot-Wire Cutting: Integrated Design, Simulation, and Fabrication of Double-Curved Surface Geometries. In *Advances in Architectural Geometry 2016*, Sigrid Adriaenssens, Fabio Gramazio, Matthias Kohler, Achim Menges, and Mark Pauly (Eds.). Hochschulverlag an der ETH Zürich, 288–305. https://doi.org/10.3218/3778-4_20

Romana Rust, David Jenny, Fabio Gramazio, and Matthias Kohler. 2016b. Spatial Wire Cutting - Cooperative robotic cutting of non-ruled surface geometries for bespoke building components. In *Living Systems and Micro-Utopias: Towards Continuous Designing, Proceedings of the 21st International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA 2016)*, S. Chien, S. Choo, M. A. Schnabel, W. Nakapan, M. J. Kim, and S. Roudavski (Eds.), Vol. 21. The Association for Computer-Aided Architectural Design Research in Asia (CAADRIA), Hong Kong, 529–538. http://papers.cumincad.org/data/works/att/caadria2016_529.pdf

Asbjørn Søndergaard, Jelle Feringa, Toke Nørbjerg, Kasper Steenstrup, David Brander, Jens Graversen, Steen Markvorsen, Andreas Bærentzen, Kiril Petkov, Jesper Hattel, Kenn Clausen, Kasper Jensen, Lars Knudsen, and Jacob Kortbek. 2016. *Robotic Hot-Blade Cutting.* Springer International Publishing, Cham, 150–164. https://doi.org/10.1007/978-3-319-26378-6_11

Oded Stein, Eitan Grinspun, and Keenan Crane. 2018. Developability of Triangle Meshes. *ACM Trans. Graph.* 37, 4, Article 77 (July 2018), 14 pages. https://doi.org/10.1145/3197517.3201303

Charlie C.L. Wang and Gershon Elber. 2014. Multi-dimensional dynamic programming in ruled surface fitting. *Computer-Aided Design* 51 (2014), 39 – 49. https://doi.org/10.1016/j.cad.2014.02.004

C. Wu, C. Dai, G. Fang, Y. Liu, and C. C. L. Wang. 2017. RoboFDM: A robotic system for support-free fabrication using FDM. In *2017 IEEE International Conference on Robotics and Automation (ICRA).* 1175–1180. https://doi.org/10.1109/ICRA.2017.7989140

Rundong Wu, Huaishu Peng, François Guimbretière, and Steve Marschner. 2016. Printing Arbitrary Meshes with a 5DOF Wireframe Printer. *ACM Trans. Graph.* 35, 4, Article 101 (July 2016), 9 pages. https://doi.org/10.1145/2897824.2925966

Haishen Zhao, Hao Zhang, Shiqin Xin, Yuanmin Duan, Changhe Tu, Wenping Wang, Daniel Cohen-Or, and Baoquan Chen. 2018. DSCarver: Decompose-and-Spiral-Carve for Subtractive Manufacturing. *ACM Transactions on Graphics* 37, 4 (2018), Article 137.

A SENSITIVITY ANALYSIS

We show how to solve Eq. (6) using sensitivity analysis. We begin by applying the chain rule on $O(\boldsymbol{y}(\boldsymbol{q}), \boldsymbol{q})$:

$$\frac{\mathrm{d}O}{\mathrm{d}\boldsymbol{q}} = \frac{\partial O}{\partial \boldsymbol{y}}\mathrm{S} + \frac{\partial O}{\partial \boldsymbol{q}}, \tag{13}$$

The term $\mathrm{S} := \frac{\mathrm{d}\boldsymbol{y}}{\mathrm{d}\boldsymbol{q}}$ is known as the *sensitivity* matrix. The analytic expression for it can be found using the fact that $G(\boldsymbol{y}, \boldsymbol{q})$ is always zero, i.e., we assume that for *any* $\boldsymbol{q}$ we can compute $\boldsymbol{y}(\boldsymbol{q})$ such that Eq. 6b is satisfied. This, in turn, implies

$$\frac{\mathrm{d}G}{\mathrm{d}\boldsymbol{q}} = \frac{\partial G}{\partial \boldsymbol{y}}\mathrm{S} + \frac{\partial G}{\partial \boldsymbol{q}} = 0\ . \tag{14}$$

By rearranging terms, we obtain

$$\mathrm{S} = -\left(\frac{\partial G}{\partial \boldsymbol{y}}\right)^{-1} \frac{\partial G}{\partial \boldsymbol{q}}\ , \tag{15}$$

and plugging into (13), we arrive at

$$\frac{\mathrm{d}O}{\mathrm{d}\boldsymbol{q}} = -\frac{\partial O}{\partial \boldsymbol{y}}\left(\frac{\partial G}{\partial \boldsymbol{y}}\right)^{-1} \frac{\partial G}{\partial \boldsymbol{q}} + \frac{\partial O}{\partial \boldsymbol{q}}\ . \tag{16}$$

We note that through a reordering of matrix multiplications, the well-known adjoint method avoids computing S directly as it evaluates $\frac{\mathrm{d}O}{\mathrm{d}\boldsymbol{q}}$. This is oftentimes more computationally efficient. However, we can leverage S to derive a second-order solver that exhibits much better convergence properties than first order alternatives.

To this end, we start by differentiating (13) to obtain

$$\frac{\mathrm{d}^2O}{\mathrm{d}\boldsymbol{q}^2} = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\frac{\mathrm{d}O}{\mathrm{d}\boldsymbol{q}} = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\left(\frac{\partial O}{\partial \boldsymbol{y}}\mathrm{S}\right) + \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\frac{\partial O}{\partial \boldsymbol{q}}\ . \tag{17}$$

The expression above involves third-order tensors, leading to somewhat cumbersome notation. For conciseness, we treat tensors as matrices and assume that contractions are clear from context. The second term in (17) simply follows as

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\frac{\partial O}{\partial \boldsymbol{q}} = \mathrm{S}^T\frac{\partial^2 O}{\partial \boldsymbol{y}\partial \boldsymbol{q}} + \frac{\partial^2 O}{\partial \boldsymbol{q}^2}\ , \tag{18}$$

while the first term evaluates to

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\left(\frac{\partial O}{\partial \boldsymbol{y}}\mathrm{S}\right) = \left(\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\frac{\partial O}{\partial \boldsymbol{y}}\right)\mathrm{S} + \frac{\partial O}{\partial \boldsymbol{y}}\left(\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\mathrm{S}\right)\ , \tag{19}$$

with

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\frac{\partial O}{\partial \boldsymbol{y}} = \mathrm{S}^T\frac{\partial^2 O}{\partial \boldsymbol{y}^2} + \frac{\partial^2 O}{\partial \boldsymbol{y}\partial \boldsymbol{q}}\ . \tag{20}$$

Here, $\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\mathrm{S}$ is a third-order tensor and

$$\frac{\partial O}{\partial \boldsymbol{y}}\left(\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\mathrm{S}\right) = \sum_i \frac{\partial O}{\partial y_i}\left(\frac{\mathrm{d}^2 y_i}{\mathrm{d}\boldsymbol{q}^2}\right).$$

The second-order sensitivity term $\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\mathrm{S}$ can be further expanded as

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{q}}\mathrm{S} = \left(\mathrm{S}^T\frac{\partial}{\partial \boldsymbol{y}}\mathrm{S} + \frac{\partial}{\partial \boldsymbol{q}}\mathrm{S}\right). \tag{21}$$

The partial derivatives of S are found by taking the second derivatives in (14) and rearranging terms. This results in

$$\frac{\partial}{\partial \boldsymbol{y}}\mathrm{S} = -\left(\frac{\partial G}{\partial \boldsymbol{y}}\right)^{-1}\left(\frac{\partial^2 G}{\partial \boldsymbol{y}^2}\mathrm{S} + \frac{\partial^2 G}{\partial \boldsymbol{q}\partial \boldsymbol{y}}\right), \tag{22}$$

$$\frac{\partial}{\partial \boldsymbol{q}}\mathrm{S} = -\left(\frac{\partial G}{\partial \boldsymbol{y}}\right)^{-1}\left(\frac{\partial^2 G}{\partial \boldsymbol{y}\partial \boldsymbol{q}}\mathrm{S} + \frac{\partial^2 G}{\partial \boldsymbol{q}^2}\right), \tag{23}$$

where once again we assume that the tensor expressions are self-evident. Combining all of the terms above leads to the following formula for the Hessian,

$$\frac{\mathrm{d}^2O}{\mathrm{d}\boldsymbol{q}^2} = \frac{\partial O}{\partial \boldsymbol{y}}\left(\mathrm{S}^T\frac{\partial}{\partial \boldsymbol{y}}\mathrm{S} + \frac{\partial}{\partial \boldsymbol{q}}\mathrm{S}\right) + \mathrm{S}^T\left(\frac{\partial^2 O}{\partial \boldsymbol{y}^2}\mathrm{S} + 2\frac{\partial^2 O}{\partial \boldsymbol{y}\partial \boldsymbol{q}}\right) + \frac{\partial^2 O}{\partial \boldsymbol{q}^2}. \tag{24}$$

Generalized Gauss-Newton. Although Newton's method generally converges much faster than L-BFGS or gradient descent, there are two issues with it. First, evaluating the second-order sensitivity term takes a non-negligible amount of time. Second, the Hessian is often indefinite and needs to be regularized. Both problems can be dealt with by simply excluding the tensor terms in (24). The result is a generalized Gauss-Newton approximation for the Hessian,

$$\mathbf{H} = \mathrm{S}^T\frac{\partial^2 O}{\partial \boldsymbol{y}^2}\mathrm{S} + 2\mathrm{S}^T\frac{\partial^2 O}{\partial \boldsymbol{y}\partial \boldsymbol{q}} + \frac{\partial^2 O}{\partial \boldsymbol{q}^2}\ . \tag{25}$$

Although $\mathbf{H}$ is not guaranteed to be positive-definite, we note that in many cases, O is a convex function of $\boldsymbol{y}$ and $\boldsymbol{q}$ such that the first and last terms are guaranteed to be positive definite. Additionally, O commonly does not explicitly couple $\boldsymbol{y}$ and $\boldsymbol{q}$, and therefore the mixed derivative (i.e. the second) term vanishes, which overall leads to a positive definite $\mathbf{H}$.

B OBJECTIVES

We list the additional objectives that constitute $E_{\mathrm{reg}}(\mathbf{x}, \mathbf{q})$. We use the already defined smooth step function $H_{a\tau}$ and the point-to-element distance function in defining some of the objectives. Some objectives are modeled as soft barrier functions, which we define using the unilateral quadratic function

$$B_c^+(t) = \begin{cases} 0 & t \le c \\ (t-c)^2 & t > c \end{cases} \quad B_c^-(t) = \begin{cases} (t-c)^2 & t \le c \\ 0 & t > c \end{cases}. \tag{26}$$

Smooth Toolsurface. While the registration objective $E_{\mathrm{dist}}(\boldsymbol{x}, \boldsymbol{q})$ described in Sec 3.3 ensures that the overall shape of the final workpiece matches the target shape to within a certain tolerance, this measure is not adequate to also achieve visually pleasing results, since even minor surface irregularities can greatly affect the final artifact's perceived appearance. We therefore wish to explicitly encourage a smooth toolsurface. Let β_{ij} be the angle between the vectors $x_{i-1,j} - x_{ij}$ and $x_{ij} - x_{i+1,j}$. We penalize large angles only for x_{ij} that are close to $\mathcal{T}$, using

$$E_1(\mathbf{x}) = \sum_{ij} H_{2a,2\tau}(\mathrm{d}(\boldsymbol{x}_{i,j}, \mathcal{T}))\beta_{ij}^2, \tag{27}$$

where $\mathrm{d}(\boldsymbol{x}_{i,j}, \mathcal{T})$ is the point-to-element distance between $\boldsymbol{x}_{ij}$ and $\mathcal{T}$.

Regular and Well Resolved Toolpath. We would like to maintain sufficient spatial resolution of our toolpath, as well as some regularity of the step size in regions that are not otherwise guided by the remaining objectives. To this end we employ both a relatively weak quadratic regularization *and* a barrier function in the form of two regularizers,

$$E_2(\mathbf{x}) = \sum_{ij} \|x_{i,j+1} - x_{i,j}\|^2,$$

and

$$E_3(\mathbf{x}) = \sum_{ij} B^+_{a_u}(\|x_{i,j+1} - x_{i,j}\|),$$

where a_u is the desired maximum step size. In addition we set a lower barrier for the tool distance to avoid discontinuity in (27),

$$E_4(\mathbf{x}) = \sum_{ij} B^-_{a_l}(\|x_{i,j+1} - x_{i,j}\|),$$

where the limit a_l is chosen small compared to the average step size.

Joint Angle Smoothness. Further, we require the motion of the robot to be smooth, such to avoid interpolation issues between consecutive poses. We do this by regularizing the step size of joint angles as

$$E_{12}(\boldsymbol{q}) = \sum_{i} \|\boldsymbol{q}_i - \boldsymbol{q}_{i-1}\|^2 .$$

No Penetration. As mentioned, the wire should avoid penetrating the $\mathcal{T}$ by more than the specified tolerance, as the damage would be irreparable. We penalize penetration using a barrier,

$$E_5(\mathbf{x}) = \sum_{ij} B^-_{a_p}(\mathrm{d}(\boldsymbol{x}_{ij}, \mathcal{T})), \tag{28}$$

where a_p determines the desired distance from the surface.

Further, we require the first and last wire shapes in $\mathcal{S}_k$ to be outside $\mathcal{W}_k$, which can be treated using,

$$E_6(\mathbf{x}) = \sum_{\substack{i=1,k \\ j}} B^-_{a_p}(\mathrm{d}(\boldsymbol{x}_{ij}, \mathcal{W}_k))$$

For shapes of $\mathcal{T}$ and $\mathcal{W}$ with sharp features, as they can frequently appear, we consider additional samples on the toolsurface respectively the wire, which we treat analogously.

No Collisions. Similarly, the robot should not collide with $\mathcal{W}_k$. To avoid collisions, we placed a number of spherical collision primitives on the robot's links, and ensure that the distance from these spheres to the $\mathcal{W}_k$ is large enough. Let $c_s, s = 1, \ldots$ be the centers of those spheres in local coordinates and r_s the spheres radii. Further, let $\mathcal{K}(c_s, \boldsymbol{q})$ be the mapping from local to global coordinates. The collision barrier is defined by

$$E_7(\boldsymbol{q}) = \sum_{s} B^-_{\varepsilon_c}\left(\mathrm{d}(\mathcal{K}(c_s, \boldsymbol{q}), \mathcal{W}_k) - r_s\right),$$

where ε_c represents a safety margin.

In addition, we require the robot not to collide with itself. This is achieved by,

$$E_8(\boldsymbol{q}) = \sum_{s_1, s_2} B^-_{\varepsilon_c}(\|\mathcal{K}(c_{s_1}, \boldsymbol{q}) - \mathcal{K}(c_{s_2}, \boldsymbol{q})\| - r_{s_1} - r_{s_2}).$$

No Plastic Deformation. Our model does not allow for cases of plastic deformation in the wire. In order to prevent these, we use *maximum shear stress theory*, which posits that failure occurs due to shear stress, to determine an equivalent tensile stress from the superposition of the tensile and shear stress. The maximum equivalent tensile stress on the perimeter of the wire is

$$\sigma = \sqrt{\sigma^2_{\text{bend}} + 4\tau^2_{\text{twist}}}, \tag{29}$$

where σ_{bend} is the tensile stress from bending, and τ_{twist} is the shear stress from twisting; see also Megaro et al. [2017] for conversion from rod deformations to volumetric stresses. Using σ we define

$$E_9(\mathbf{x}) = B^-_{a_s}(\sigma),$$

where a_s is determined experimentally.

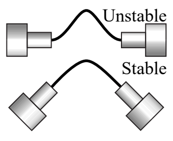
No Stretching. While we assume that the wire is virtually inextensible, the physical model we use does treat the wire as extensible, though with a very high stretching stiffness. To prevent large longitudinal deformation, which would translate to rupture of the wire, we also penalize stretching by using the objective

$$E_{10}(\mathbf{x}) = E_{\text{stretch}}(\mathbf{x}).$$

Robust Wire States. As described in Sec. 3.5 and further detailed in Appx. C, we can achieve states with high resistance to external disturbances and imprecise robot movements by introducing a lower bound for the angle spanned by the tangents of the two rod ends. We formulate this through the objective

$$E_{11}(\mathbf{x}) = \sum_{i} B^+_{a_\gamma} \angle(x_{i,2} - x_{i,1}, x_{i,m-1} - x_{i,m}). \tag{30}$$

C ROBUSTNESS OF WIRE STATES



In Sec 3.5 we address the requirement to allow only wire states that are sufficiently robust to external perturbations and changes in boundary conditions. As stated, measurements for these properties are given by the smallest eigenvalue $\lambda_{\min}$ of the Hessian $\mathbf{H}_{\text{wire}}$ and the largest singular value $\sigma_{\max}$ of the sensitivity matrix $\mathbf{S}$, respectively. Note that the two are inversely related through Eq. (9).

To provide an example for the problem, one very notable case is depicted in the inset figure on the top. When the ends of the rod are collinear, it follows alone from considerations of symmetry that, besides the depicted state, any rotation around the axis given by the two rod ends is an equilibrium state as well. In a physical setup, pushing the bent rod around this axis indeed requires virtually no force. The smallest eigenvalue $\lambda_{\min}$ is zero, and it follows from (9) that S is not defined. However, as a configuration approaches this state, the sensitivity will become arbitrarily large. The practical implication is the following: While for perfectly collinear ends the rod can buckle in any direction, even an arbitrarily small rotation of one of the boundaries will uniquely dictate the direction of buckling.

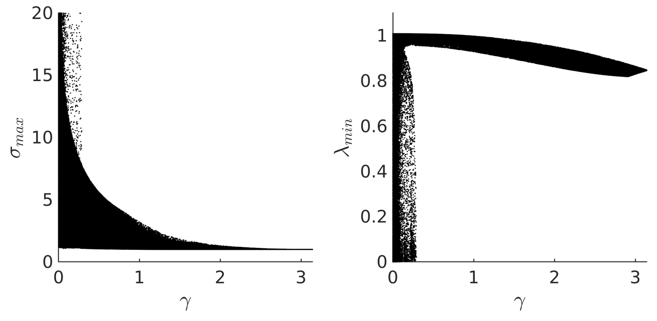


Fig. 14. Sensitivity $\sigma_{\max}$ (left) and stiffness $\lambda_{\min}$ (right) versus the relative angle γ between tangents at the rod's end points. Sensitivity is normalized w.r.t. rod length. Stiffness is scaled w.r.t. the highest achievable stiffness of the system.

Consequently, an equally small rotation in opposite direction will cause the rod to buckle to the opposite direction as well, resulting in large positional changes on the center of the rod.

We have motivated our need for a simplified criterion for robustness in Sec 3.5, and based on an extensive numerical study we find one in the angle γ that is spanned by the tangents of the wire ends (see Eq (11) and (12)).

Empiric Verification. In Fig. 14 we summarize the sensitivity and stiffness for about 1.7 million samples of admissible rod configurations, which were obtained from a combination of randomized sampling and worst-case optimization. Admissible configurations for this experiment are those for which the deformation is sufficiently small to be reversible in the physical wire. In our case that is a stainless-steel rod of $1mm$ diameter and a length of $0.53m$, at a temperature of 250°C. The maximum deformation, as determined empirically, is the equivalent to a bending radius of $0.154m$. The total deformation of any point on the wire is based on bending and twist, as defined in (29). With this constraint we find that for a sufficiently large γ, there appear to be bounds on the possible stiffness and sensitivity of the rod state. By introducing an objective that reliably enforces a lower bound of, e.g., $\gamma > 0.6$, we can ensure a stiffness $\lambda_{\min}$ of at least 81% of the maximal possible stiffness of the rod, as well as a sensitivity $\sigma_{\max}$ of not more than 5.0. Note that for this specific experiment the boundaries of the rod are not defined through a robot arm, but directly parameterized through position and Euler-angles of one rod end. Further, we only consider the partial rod state $\boldsymbol{x}_i$, such that the sensitivity here relates the node positions to the position and rotation of the boundary.

The result may seem surprising at first. But the reader shall be reminded that these findings do not in any way apply to general elastic rods. Rather, it is the yield strength, i.e. the elastic limit of the metal rod, which reduces the physically feasible configuration space so severely that such a simplification becomes possible.

The sample configurations for this study were obtained as follows. Initially, a set of 60 samples was crated from random boundary conditions, uniformly distributed across the domain of admissible parameters. Then, about 1 million samples were created by adding a small, random deviation to the boundary of an existing wire state. These existing samples in turn were randomly selected from existing samples, but with a strong bias towards the most recently computed samples, and states with poor robustness. This approach essentially results in a set of Wiener processes with occasional bifurcations.

Finally, the remaining 0.7 million samples stem from a worst-case optimization strategy. Starting from a selection of existing samples, we employed gradient descent to optimize for (1) low stiffness, (2) high sensitivity, (3) low stiffness and large γ (4) high sensitivity and large γ.

It should be pointed out that these data do not support the conclusion that no problematic wire states exist with large γ. However, given that none appeared in this experiment, we consider it reasonably unlikely that such a configuration will be encountered when applying our optimization framework.