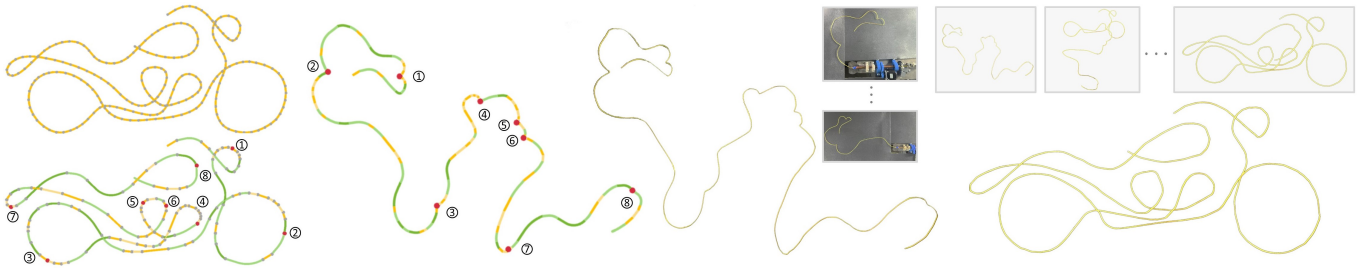
Tune-It: Optimizing Wire Reconfiguration for Sculpture Manufacturing

QIBING WU*, Shandong University, China
ZHIHAO ZHANG*, Shandong University, China
XIN YAN, Shandong University, China
FANCHAO ZHONG, Shandong University, China
YUEZE ZHU, University College London, United Kingdom
XURONG LU, Shandong University, China
RUNZE XUE, Shandong University, China
RUI LI, Shandong University, China
CHANGHE TU, Shandong University, China
HAISEN ZHAO†, Shandong University, China



(a) (b) (c) (d)

Fig. 1. We introduce a novel computational framework for wire sculpture fabrication that seamlessly integrates the strengths of wire-bending machines and human craftsmanship. The input wire (top of (a)) is fitted into a segment-bendable wire (bottom of (a)) with bending fabricable line segments and circular segments (indicated in different colors), considering the manufacturing constraints of wire-bending machine. For collision-free wire bending, we generate a tuned wire in which tuned points are indicated with red dots (b). The manufacturing result after the machine-bending stage with our DIY bending machine (c). The final result after the human-bending stage by a non-skilled participant (d). The embedded photos show the intermediate manufacturing process of machine-bending stage and human-bending stage.

Wire sculptures are important in both industrial applications and daily life. We introduce a novel fabrication strategy for wire sculptures with complex geometries by tuning the target shape to a collision-free shape for the wire-bending machine and then bending it back to the target by a human. The key challenge lies in tuning the least number of bending points, which is formulated as an "Optimizing Wire Reconfiguration" problem. We first fit the input target wire with consecutive line segments and circular segments to ensure the bending manufacturing constraints for each segment, then generate tuned wire through a bilevel optimization. This involves selecting the bending points at the upper level with a beam search strategy and determining the specifically tuned angles at the lower level. We perform a thorough physical evaluation using a DIY wire-bending machine. The results show the effectiveness of our proposed approach in realizing a wide range of intricate and complex wire sculptures.

CCS Concepts: • **Computing methodologies** → **Shape modeling**; *Graphics systems and interfaces.*

Additional Key Words and Phrases: Wire sculpture, wire fabrication, wire reconfiguration, computational fabrication

ACM Reference Format:
Qibing Wu, Zhihao Zhang, Xin Yan, Fanchao Zhong, Yueze Zhu, Xurong Lu, Runze Xue, Rui Li, Changhe Tu, and Haisen Zhao. 2024. Tune-It: Optimizing Wire Reconfiguration for Sculpture Manufacturing. *ACM Trans. Graph.* 1, 1 (September 2024), 11 pages. https://doi.org/10.1145/3680528.3687588

*Equal contribution
†Corresponding author

Authors' addresses: Qibing Wu, qibingwu0228@gmail.com, Shandong University, Qingdao, China; Zhihao Zhang, zhangsfsd1@gmail.com, Shandong University, Qingdao, China; Xin Yan, io.yanxin@gmail.com, Shandong University, Qingdao, China; Fanchao Zhong, fanchaoz98@gmail.com, Shandong University, Qingdao, China; Yueze Zhu, yueze.zhu@outlook.com, University College London, London, United Kingdom; Xurong Lu, lxr20030328@foxmail.com, Shandong University, Qingdao, China; Runze Xue, runzexue@outlook.com, Shandong University, Qingdao, China; Rui Li, ruizai6@hotmail.com, Shandong University, Qingdao, China; Changhe Tu, chtu@sdu.edu.cn, Shandong University, Qingdao, China; Haisen Zhao, haisenzhao@sdu.edu.cn, Shandong University, Qingdao, China.

0730-0301/2024/9-ART $15.00
https://doi.org/10.1145/3680528.3687588

1 INTRODUCTION

Wire sculptures [Wikipedia 2024] serve as a remarkably simple yet effective embodiment of design concepts, exuding a distinctive aesthetic appeal as seen in various art forms, including wire sculpture art, wire-wrapped jewelry, furniture design, and so on. Wire sculptures also play a pivotal role in various industrial sectors, including

Fig. 2. Wire sculptures: (left) man-made wire artworks, as a 3D ballerina art and a metal artwork from artist Diego Cabezas; (right) custom wire forms fabricated by the wire-bending machines, including springs, wire netting, hooks and metal fences.

springs, wire netting, and metal fences; see Figure 2. Several singular bends can transform a 1D wire into a sophisticated 2D or 3D wire sculpture. This simplicity of fabrication highlights the unique beauty inherent in these creations.

When it comes to fabricating wire structures, there are two common methods, manual bending and computer numerical control (CNC) bending. The manual strategy involves the use of artists' craftsmanship, which allows for great flexibility in creating highly intricate shapes. However, this method is best suited for experienced experts and not beginners. CNC strategy involves automatically bending a wire through a series of extrusion and bending operations [Baraldo et al. 2022]. Many high-end *wire-bending machines* are available in industry to produce functional products; see Figure 2.

However, the intricacy and geometric details that can be produced by wire-bending machines are largely limited by the collision-free constraint, which does not allow any self-collision and global collision between the wire and the machine body during bending, as the collision disrupts the physical fabrication process; see a failure case in Figure 3. Consequently, wire-bending machine can handle only simple or regular geometries, such as planar rod structures [Miguel et al. 2016] and Eulerian wires [Lira et al. 2018] for the abstraction of 3D shapes. We are curious if it is possible to greatly expand the range of objects that can be manufactured by wire-bending machines, such as the artistically attractive wire structures.

This paper aims to address this curiosity by presenting a "**Machine-And-Then-Human-Bending**" strategy to fabricate a single continuous wire with intricate geometric features, which we call the strategy two-stages-bending for short. In machine-bending stage, wire-bending machine creates a deformed version of the input wire to ensure a collision-free CNC bending process. In human-bending stage, we bend tuned wire backwards to the target wire with the help of a human; see Figure 1 and Figure 3. Due to flexible craftsmanship, even non-skilled humans excel at bending a few specific angles using a roughly finished wire produced in machine-bending stage. It is also essential to reduce the labor involved by human helpers. Consequently, it is preferable to have the minimum number of tuned points.

Inspired by LineUp [Yu et al. 2019], which produces a collision-free physical transform motion, we generate a collision-free tuned wire by tuning a minimum number of bending angles. The generation of tuned wire is a specific variation of optimal reconfiguration planning (ORP), which has been proven to be complete with NP [Hou and Shen 2010]. ORP searches for the least number of reconfiguration steps to transform between configurations, for example, a square-shaped robot to a circular shape. In our case, the

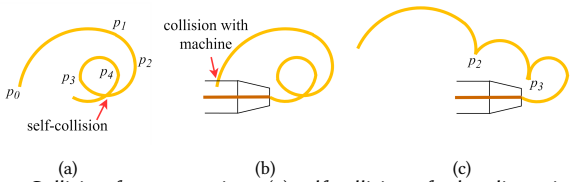


Fig. 3. Collision-free constraints: (a) self-collision of a bending wire, (b) global collision between the bending wire and wire-bending machine, (c) ensuring wire-bending machine is collision-free by tuning p_2 and p_3, which will be bent backwards in the human-bending stage.

input wire is taken as a chain-type modular reconfiguration robot; however, the shape of the target wire (the target robot configuration) is unknown; our objective is to generate a collision-free tuned wire from the input wire. The problem, referred to as "**Optimal-Wire-Reconfiguration, OWR**" in this paper, is computationally challenging due to its exponential search space, where each input bending angle has its tuning freedom. More importantly, the global constraint of being collision-free further increases its difficulty. To determine whether and how to tune a bending angle, we must consider the bent wire from all previous bending processes.

In this paper, we present a fully automatic computational framework to make problem OWR tractable which enables us to generate a collision-free tuned wire, with as few tuned points as possible. First, fit the input wire into a set of fabricable bending segments, such as 3D line segments and circular segments, so that the fitted line segments and circular segments meet the bending manufacturing constraints. Second, generate tuned wire through bilevel optimization, where the upper-level optimization focuses on selecting the bending points to be tuned, while the lower-level optimization aims at determining the tuned angles with the consideration of eliminating all collisions. Moreover, we explore a weighted beam search strategy to trade off search efficiency and solution quality. Finally, we validated our algorithm by conducting experiments on 3D models and fabricating a set of 2D models using a DIY bending machine.

Contributions. Our main contribution is the development of a wire-bending method that combines human collaboration with wire-bending machines. This novel solution enables even non-skilled humans to create wire sculptures with intricate geometric details. Furthermore, our research has potential benefits for the industry. It allows wire-bending machines to achieve a higher complexity in their output. Another significant contribution is our computational framework, which efficiently identifies a small number of tuned points that result in fabricable and collision-free tuned wire. To validate our method, we performed evaluations using wires of varying geometric complexity.

2 RELATED WORK

Computational fabrication has been a popular topic in the field of computer graphics [Bickel et al. 2018; Matusik and Schulz 2019; Umetani et al. 2015]. Many of these works utilize additive or subtractive manufacturing to create 3D geometries [Martínez et al. 2016; Narumi et al. 2023; Wu et al. 2016; Yue et al. 2017; Zhong et al. 2023]. This section will primarily review related works on wire sculpture fabrication via machines or through manual wire bending.

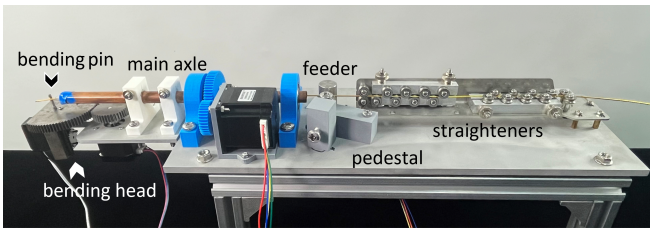


Fig. 4. Our DIY wire-bending machine, based on [Dejan 2018], consists of straighteners, a wire feeder, a bending head with bending pin, and pedestal. The rotation center of the bending pin is aligned with the wire outlet. The DuPont wires in the picture connect the motors and drivers.

Machine-based Sculpture Bending. [Xu et al. 2018] and [Liu et al. 2017] apply wire-bending machines to fabricate elastically deforming wire structures that serve as the skeleton of the kinetic wire characters produced. However, their wire structures have relatively simple geometries. To increase the complexity of fabricating wire sculpture, a common method is the "decompose-then-assemble" strategy: [Miguel et al. 2016] abstracts the target shape using a set of planar-rod wire contours, then computes the stability-aware assembly sequence; [Lira et al. 2018] and [Bhundiya and Cordero 2023] decompose the wire abstraction into a series of fabricable Eulerian wires and then assemble them with connectors.

These works commonly involve the fabrication of decomposed subwires using wire-bending machines, followed by a manual assembly process. In contrast, this paper processes a single wire with complex geometric features without breaking down the target sculpture. We tune it to create a fabricable wire structure, bending both line segments and circular segments for improved aesthetic appeal. These works only produced straight polyline wire structures. In addition, neither the above methods nor our method takes into account the planning of the bending sequence itself. [Baraldo et al. 2022] focuses on optimizing the bending orders for wire-bending machines, while [Liu et al. 2023] presents a motion planner algorithm for the robot arm to collaborate with a wire-bending machine to perform 3D metal wire curving.

Human-based Sculpture Bending. Bending complex wire sculptures entirely by hand is a labor-intensive task, especially for inexperienced users. As a result, researchers have developed various methods to assist humans during the manual bending process. [Yang et al. 2021] provides detailed instructions for bending the wires. [Iarussi et al. 2015], [Torres et al. 2016] and [Tojo et al. 2024] compute printable physical support structures that enables users to wrap the wire around the protruding structure. [Wang et al. 2019] generates grooves on the 3D surface of the given model and prints them to serve as wire-mold structures. [Garg et al. 2014] creates a 3D scaffold by laser cutting and labels the plannar wire mesh material to assist the bending process of the wire mesh.

These tasks significantly enhance the efficiency of manual bending. However, the manual bending process is still time-consuming and labor-intensive. In contrast, we minimize human involvement by allowing them to bend the semi-finished wire articles produced by machine bending.

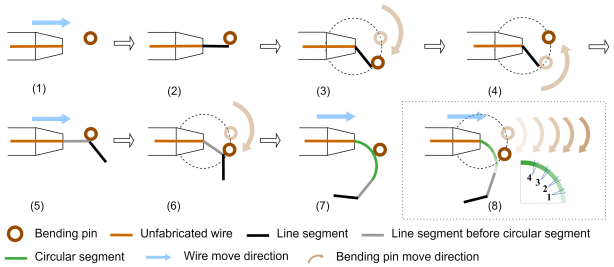


Fig. 5. Three bending strategies for line-circular freeform fabrication. (1-4): Flexion bending strategy for line segment. (5-7): Interpolated bending strategy for circular segment. (8): Strike bending strategy for circular segment.

Optimal Reconfiguration Planning. As pointed out in the previous section, our work is also related to optimal reconfiguration planning (ORP) problems in the robotics domain. It is worth clarifying the similarities and differences between our work and ORP. [Hou and Shen 2010] proves the NP-completeness of the ORP problem for chain-type modular robots. ORP involves searching for the minimum number of reconfiguration steps required to transform between different robot configurations, which has brought many applications, including the transformation of Rubik's Snake [Zhang et al. 2022], foldable chain-based transformation [Zhang et al. 2023], furniture reconfiguration [Song et al. 2017], chain-based physical transformation of a single-line structure [Yu et al. 2019].

Our work shares an objective similar to the typical ORP problem, where the minimum number of tuned bending points corresponds to the least number of reconfiguration steps. Furthermore, the angle-tuning operators used in our paper are commonly seen in the reconfiguration of chain-type modular robots. For the differences, the most obvious difference is that the bending wire is not a robot capable of moving by itself. Instead, the movement of the wire is passive and relies on external forces from wire-bending machine or humans. Second, the target robot configuration in ORP is given. However, in our case, it's not, where we need to generate a target tuned wire from the input wire, by searching for the fewest number of tuned bending points. Third, the robot module can disconnect from one module and connect to another, which is not allowed in our case. Finally, the collision constraint is not only limited to the self-collision avoidance considered in ORP. We should also consider avoiding collisions with the wire-bending machine.

3 OVERVIEW

This section first clarifies the wire bending setting and manufacturing constraints in the proposed two-stages-bending process. Then we outline the optimization formulation and our algorithm's overview.

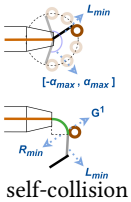
3.1 Wire Bending Setting

Figure 4 illustrates the components of a typical wire-bending machine, including wire straighteners, feeder, and bending pin, which are used to straighten, feed, and bend the input wire, respectively. Two types of bending segments can be produced by the wire-bending machine [Baraldo et al. 2022]: line segments using the *flexion bending* strategy and circular segments using the *interpolated bending* strategy and the *strike bending* strategy, as demonstrated in Figure 5:

- *Flexion bending* strategy: (i) Feed the wire through the main axle until its preset length (1-2). (ii) Bend the wire by the bending pin until its preset bending angle to fabricate a line segment (3). (iii) Return the bending pin to its home position for the subsequent bending operations (4).
- *Interpolated bending* strategy: (i) Feed the wire through the main axle until it reaches the bending pin (5). (ii) Bend the wire using the bending pin until its preset bending angle (6). (iii) Fix the bending pin and continuously feed the wire to fabricate a circular segment (7). The circular segments produced by this strategy have a limited minimum radius R_{min}, further detailed in Appendix A.
- *Strike bending* strategy: To create circular segments with a radius smaller than R_{min}, multiple but incremental flexion bending operations are applied in the strike bending strategy, where each bending operation is followed by an incremental feeding distance L_{in} of the wire (8).

Manufacturing constraints. Apart from the collision-free constraint illustrated in Figure 3, there are three manufacturing constraints to guarantee the fabricability of each bending segment during machine-bending stage. The detailed interpretation of the three constraints can be found in Appendix B.

- *Minimal length constraint*: each line segment has to be not shorter than L_{min}, which is defined as the distance between the wire outlet and the bending pin.
- *Bending angle range constraint*: the bending angles between adjacent segments are limited by the rotation range of the bending pin around the wire outlet, which is $[-\alpha_{max}, \alpha_{max}]$.
- *G1 line segment constraint*: the front of each circular segment has to be followed by a line segment, which is G1 continuous to the circular segment.



The above three constraints are illustrated in the inset figure. In summary, a line segment is *bending fabricable* as long as its length is greater than L_{min}, a circular segment is *bending fabricable* as long as it has a G1 continuous line segment in preceding. A wire is considered bending fabricable if it meets the following conditions: 1) it's collision-free (with no self-collision or global collision), 2) its included bending segments are bending fabricable, and 3) the bending angle between consecutive segments is within the range of $[-\alpha_{max}, \alpha_{max}]$.

3.2 Algorithm overview

Let $\mathcal{W}$ denote the input wire, which cannot be fabricated by the wire-bending machine due to its failure to meet the collision-free constraint (Figure 3) or other manufacturing constraints (Subsection 3.1). $\mathcal{W}$ comprises a sequence of sampling points $\{p_0, p_1, ..., p_n, p_{n+1}\}$ and a sequence of *bending segments* $\{s_0, s_1, ..., s_n\}$, where s_i can be *circular segment* (bent by the interpolated bending strategy) or *line segment* (flexion bending or strike bending strategy), with $\{p_i, p_{i+1}\}$ as the two end points of $s_i, i = 0, 1, ..., n$. The points $\{p_1, p_2, ..., p_n\}$ represent the *bending points* of $\mathcal{W}$, associated with the desired *bending angles* $\{\alpha_1, \alpha_2, ..., \alpha_n\}$ operated by wire-bending machine. Here, α_i denotes the bending angle between consecutive segments s_{i-1} and s_i for $i = 1, 2, ..., n$.

To apply the Machine-And-Then-Human-Bending strategy, we first generate a segment-bendable wire $\mathcal{W}'$ by fitting a sequence of fabricable bending segments of $\mathcal{W}$, then generate a collision-free tuned wire $\mathcal{W}^*$ by tuning a subset of bending angles of $\mathcal{W}'$. Assume that $\mathcal{W}'$ and $\mathcal{W}^*$ consist of $m+1$ fabricable bending segments. Let p_i' and α_i' be the bending point and bending angle of $\mathcal{W}'$, p_i^* and α_i^* be the bending point and bending angle of $\mathcal{W}^*$, s_i' and s_i^* be the bending segment of $\mathcal{W}'$ and $\mathcal{W}^*$, separately. p_i^* is a *tuned point*, as long as α_i^* is different to α_i', otherwise, it's a *constant point*.

We aim to minimize the number of tuned points of $\mathcal{W}^*$, in which $\mathcal{W}^*$ must meet two key self-collision criteria: 1) non-adjacent bending segments of $\mathcal{W}^*$ should not collide with each other; 2) during the machine-bending stage, $\mathcal{W}^*$ must avoid any collisions with wire-bending machine ($\mathcal{M}$). Additionally, the manufacturing constraints impose further requirements on $\mathcal{W}^*$, including the following: bending angle α_i^* must be smaller than α_{max} (*bending angle range constraint*), bending segment s_i^* must be shorter than L_{min} (Minimal length constraint). Furthermore, because the tuned points are manually bent, it becomes challenging for users to accurately position them if the bending angle is too small. To address this, our algorithm sets a minimum angle of 10° for the bending angles.

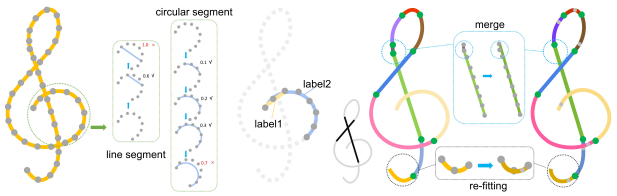
The OWR optimization problem described above presents threefold challenges: 1) the initial bending segments of $\mathcal{W}$ might not be fabricable under the manufacturing constraints outlined in Subsection 3.1. 2) exponential search space $[-\alpha_{max} : \alpha_{max}]^m$, where each bending angle α_i^* has its tuning freedom $[-\alpha_{max} : \alpha_{max}]$. 3) bending angles are not mutually independent, where the feasible bending angles of α_i^* are determined by all preceding bending angles, $\{\alpha_1^*, \alpha_2^*, ..., \alpha_{i-1}^*\}$, as shown in Figure 10.

To address the above problem, we generate $\mathcal{W}'$ with a curve fitting process in Section 4. Second, we compute $\mathcal{W}^*$ by determining tuned points with a heuristic-based search procedure and assigning the bending angles of $\mathcal{W}^*$ with a *collision resolving operator* (CRO) in Section 5.1. Moreover, a weighted beam search strategy is applied to trade off search efficiency and quality (Subsection 5.2). Parameters and terminologies defined in this paper are in Appendix M.

4 SEGMENT-BENDABLE WIRE GENERATION

This section generates a segment-bendable wire $\mathcal{W}'$ from a non-fabricable input wire $\mathcal{W}$ by fitting $\{s_0, s_1, ..., s_n\}$ into a series of fabricable bending segments $\{s_0', s_1', ..., s_m'\}$. This involves first creating a set of candidate fabricable segments starting from each bending segments of $\mathcal{W}$ via a forward-and-backward traverse procedure, then generating non-overlapping bending fabricable segments with a graph-cut decomposition process.

Candidate fabricable segments. This step fits a candidate fabricable segment ($\dot{s}_i$) as long as possible from each bending segment (s_i) of $\mathcal{W}$, where $\dot{s}$ can be line segment ($\bar{s}$) or circular segment ($\check{s}$). From s_i, we traverse along $\mathcal{W}$'s bending segments in both the backward and forward directions, which is initialized by the backward s_{i-1} and the forward s_{i+1}. During each iteration, we separately fit a $\bar{s}$ and a $\check{s}$ with the traversed bending segments. Such traversal will continue until the fitting error (E) of the fitted bending segment ($E(\bar{s})$ or $E(\check{s})$) exceeds a certain threshold (ϵ), which is set at 0.6 mm in our implementation. The fitting error (E) is calculated as



(a) (b) (c) (d)

Fig. 6. Demonstration of segment-bendable wire generation. (a) Input a sequence of line segments s_i. Each s_i generates a line segment and circular segment with a forward-and-backward traverse. (b) shows the generated two candidate fabricable segments. Non-overlapping segment-bendable wire is obtained by graph-cut optimization (c), where line segments and circular segments are illustrated with black and gray lines, separately. (d) shows a re-fitting process to meet the manufacturing constraints, with small gray points representing the intersection point in G1 line segment constraint.

the maximum Euclidean distance from the fitted bending segment to the traversed bending segments. Finally, we choose the longer fitted bending segments among $\bar{s}$ and $\check{s}$ to be the candidate fabricable segment $\dot{s}_i$ from s_i. Ultimately, we obtain $n+1$ candidate fabricable segments $\{\dot{s}_0, \dot{s}_1, ..., \dot{s}_n\}$, some of which may be redundant and overlapping, as illustrated in Figure 6(b). To address the first problem, we simply retain one candidate fabricable segments from the duplicated ones and get $\{\dot{s}_0, \dot{s}_1, ..., \dot{s}_d\}$. Note that the bending segment fitting procedure with the traversed bending segments is provided in Appendix C.

None-overlapping fabricable segments. As shown in Figure 6, each bending segment (s_i) of $\mathcal{W}$ can be taken as a graph node, bending point (p_i) of $\mathcal{W}$ can be taken as a graph edge between graph nodes s_i and s_{i+1}. Hence, the generation of non-overlapping bending fabricable segments from these candidate fabricable segments is a typical multi-label graph-cut problem [Schmidt et al. 2009], where the candidate fabricable segments serve as labels in the graph-cut:

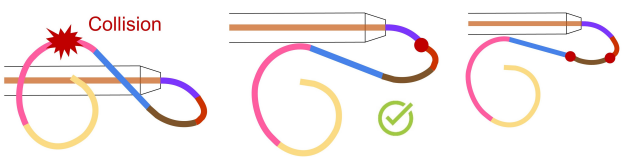
$$\mathcal{E} = \sum_{s_j \in S_w} D(s_j, l_i) + \sum_{(s_j, s_{j+1}) \in S_w} S(s_j, s_{j+1}, l) + L(l) \tag{1}$$

where l_i indicates the i-th candidate fabricable segment $\dot{s}_i$, $S_w = \{s_0, s_1, ..., s_n\}$ is the bending segments of $\mathcal{W}$. $D(s_j, l_i)$ is the data term, estimating the cost of assigning s_j to label l_i:

$$D(s_j, l_i) = \begin{cases} \lambda_1 * d(s_j, l_i), \ if \ s_j \ in \ \dot{s}_i \\ \infty, \ otherwise \end{cases} \tag{2}$$

where $d(s_j, l_i)$ indicates the fitting error distance between s_j and $\dot{s}_i$, λ_1 serves as a scaling coefficient of $d(s_j, l_i)$ (λ_1 is 30 in our implementation). The smooth term $S(s_j, s_{j+1}, l)$ measures the cost of assigning different labels to two adjacent s_j and s_{j+1}. We set $S(s_j, s_{j+1}, l)$ to 1 if $l(s_j) \neq l(s_{j+1})$, otherwise, $S(s_j, s_{j+1}, l) = 0$. Label term $L(l)$ is to minimize the number of none-overlapping fabricable segments after optimization, $L(l) = \lambda_2 * l(s_j), s_j \in S_w$, where λ_2 is the penalty coefficient of the number of labels (50 in our implementation).

Until now, we get a set of non-overlapping candidate fabricable segments $\{\dot{s}_0, \dot{s}_1, ..., \dot{s}_k\}$. However, these segments may not meet the manufacturing constraints in Subsection 3.1. This is because G1 line segment constraint has never been considered in the above process and minimal length constraint may be broken during the graph-cut process. To address this problem, we perform a *re-fitting* process on $\{\dot{s}_0, \dot{s}_1, ..., \dot{s}_k\}$, which first merge line segments whose length is less than L_{min} with its adjacent candidate fabricable segments. After that, for each circular segment $\dot{s}_i$, we separately fit a $\bar{s}'_i$ and a pair of $(\bar{s}'_i, \check{s}'_i)$, where $\bar{s}'_i$ is a fitted line segment of $\dot{s}_i$, $(\bar{s}'_i, \check{s}'_i)$ are two fitted segments of $\dot{s}_i$ that ensure G1 line segment constraint. Finally, we choose the one with the smaller fitting error (E) from $\bar{s}'_i$ and $(\bar{s}'_i, \check{s}'_i)$, as the result bending segment of $\dot{s}_i$. In the end, we get the segment-bendable wire $\mathcal{W}'$, with bending segments $\{s'_0, ..., s'_m\}$ and bending points $\{p'_1, ..., p'_m\}$. Note that the fitting procedure of $\bar{s}'_i$ and $(\bar{s}'_i, \check{s}'_i)$ is in Appendix C.



(a) (b) (c)

Fig. 7. Demonstration of CRO algorithm. (a) Collision happens when bending the purple segment of the wire. (b) Our CRO algorithm optimizes one tuned point contained in the manufactured wire. (c) If collisions are not resolved by one tuned point, two tuned points are included in the CRO.

5 BILEVEL TUNED WIRE OPTIMIZATION

$\mathcal{W}'$ guarantees fabricable for each of its bending segment $\{s'_0, ..., s'_m\}$. However, $\mathcal{W}'$ may still be non-fabricable due to violations of the collision-free constraints. To resolve the collision problem, this section generates a collision-free tuned wire $\mathcal{W}^*$ by tuning the minimal subset of bending angles $\{\alpha'_1, ..., \alpha'_m\}$ of $\mathcal{W}'$.

5.1 Greedy-based searching method

A greedy-based strategy is proposed to generate $\mathcal{W}^*$, where we try to realize $\mathcal{W}'$ directly by a simulated wire-bending machine, following the sequence of bending angles $\{\alpha'_1, ..., \alpha'_m\}$ at bending points $\{p'_1, ..., p'_m\}$. During the simulation, set p'_i to tuned point once a collision occurs when the pin bends angle of α'_i, which can be done in three steps: (i) first compute the feasible bending angle range $([0, \tilde{\alpha}^i_{max}])$ of p'_i by the Feasible Angle Operator which is detailed in Appendix D; (ii) if $\alpha'_i \in [0, \tilde{\alpha}^i_{max}]$, it indicates no collision after bending p'_i to α'_i, set p^*_i as constant point and $\alpha^*_i = \alpha'_i$; (iii) if $\alpha'_i \notin [0, \tilde{\alpha}^i_{max}]$, it indicates collision occurring, set p^*_i as tuned point and $\alpha^*_i = \alpha'_i$. Then resolve the collision by tuning the tuned points in $\{p'_1, p'_2, ..., p'_i\}$ with the Collision Resolving Operator.

Collision Resolving Operator (CRO). This operator aims to resolve the collisions between the realized segments $\{s^*_0, s^*_1, ..., s^*_{i-1}\}$ of p^*_i and the wire-bending machine, by tuning the bending angles $\{\alpha^*_1, ..., \alpha^*_i\}$ of tuned points in $\{p'_1, p'_2, ..., p'_i\}$; see Figure 7. As a typical combinatorial problem, we apply a heuristic-based searching strategy to tune these tuned points. We start by tuning a single tuned point randomly using an enumeration method. This involves locally increasing or decreasing the bending angle of that tuned point. If the collision disappears, we return the result setting of bending angles. If not, we randomly tune two tuned points, with different combinations of bending angles for the two tuned points. We continue to increase the number of tuned points for exploration until there is no collision. If the collision is still there finally, it indicates that the current setting of $\mathcal{W}^*$ is not fabricable. The pseudocode is

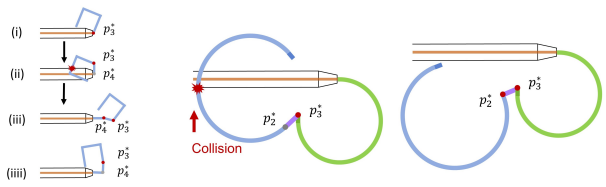


Fig. 8. (a) A demonstration of the greedy approach, in which collisions occur when p_3' and p_4' bend to α_3' and α_4', resulting in two tuned points p_3^* and p_4^*. However, if we set p_4^* to constant point and its collision can be resolved by tuning p_3^*, resulting in one single tuned point, p_3^*. (b) A demonstration of the necessity to set tuned point even if there are no collisions, in which the collision of p_3^* cannot be resolved with the constant point p_2^*. However, if we set p_2^* to tuned point, the collision of p_3^* can be resolved by tuning the bending angle of p_2^*.

given in Appendix N.2. A nonlinear continuous optimization has been attempted to resolve collisions. Please refer to Appendix E for a more detailed description and discussion.

5.2 Beam search optimization

Figure 8 shows that the greedy approach can be improved to further reduce the number of resulting tuned points by exploring various options to set each bending point as constant point or tuned point. A beam search optimization strategy is proposed, with its solution space forming a search tree ($\mathcal{T}$), where each node encodes the setting of bending points $\{p_1^*, p_2^*, ..., p_i^*\}$, including the bending point type and bending angle of each p_j^*, $j = 1, 2, ..., i$; see Figure 11. A comparison between the greedy-based method and the beam search strategy is presented in Appendix F, which clearly shows the results of the beam search algorithm in fewer tuned points. A more detailed discussion on the performance of the proposed beam search optimization is provided in Appendix O. Next, we present the three core steps of the beam search algorithm.

Candidate nodes generation. Each bending point p_i' can be set to constant point or tuned point: 1) the option of constant point will be definitely explored to generate a node of $\mathcal{T}$, regardless of whether a collision occurs when p_i' bends to α_i'. 2) the option of tuned point will be explored to generate a node of $\mathcal{T}$, if a collision occurs. If there are no collisions, p_i' will be set to tuned point and generate a node of $\mathcal{T}$, depending on the probability of ω (ω is 0.3), to deal with the situation shown in Figure 8. If CRO cannot resolve collisions after setting the bending point type to p_i', this node is abandoned. If all nodes in $\mathcal{T}$ of the i beam iteration are abandoned, where $i < m$, it indicates the beam search fails to generate a tuned wire. We propose a strategy to deal with such situation in Appendix G.

Candidate nodes scoring. We sort all candidate nodes according to a radix-sorting strategy, where the first rule is the number of constant points in $\{p_1^*, p_2^*, ..., p_i^*\}$, and the second rule is the sum of $\mathcal{L}(p_j^*)$, if p_j^* is a tuned point. $\mathcal{L}(p_j^*)$ is to estimate the potential priority of setting a tuned point, which is detailed in Appendix H. We set the beam width $W_{Beam} = 5$.

Final selection of tuned wire. The search ends when the height of $\mathcal{T}$ reaches m. $\mathcal{W}^*$ is selected from the leaf node with the minimum number of tuned points at the level m of $\mathcal{T}$.

Table 1. *Statistics of the results.* L_t is the length of target wire (mm). L_d is the diagonal length of the bounding box of wire (mm). $\#N_p$ is the number of sampling points on the target wire. $\#N_i$ is the number of self-intersection points. $\#N_s$ indicates the total number of segments after fitting. $\#N_l$ indicates the number of line segments after fitting. $\#N_c$ indicates the number of circular segments after fitting. E_{fit} is the average fitting error (mm), which is the Euclidean distance (L2) to the original wire. $\#N_{bs}$ is the total number of nodes in beam search tree. $\#N_{tp}$ indicates the total number of tuned points after beam search.

					Sec. 4				Sec. 5	
Model	L_t	L_d	$\#N_p$	$\#N_i$	$\#N_s$	$\#N_l$	$\#N_c$	E_{fit}	$\#N_{bs}$	$\#N_{tp}$
Bird	2473.5	710.9	340	5	96	60	36	0.376	362	3
Bull	3307.8	784.4	313	5	107	86	21	0.499	506	7
Cat	2043.5	748.2	314	1	91	75	16	0.326	353	3
Dolphin	2253.5	778.6	339	2	107	82	25	0.249	462	3
Leaf	3429.8	826.0	345	3	126	92	34	0.348	843	9
Ma	3304.2	668.3	157	1	64	48	16	0.322	639	4
Motorbike	4488.8	908.2	384	7	118	72	46	0.648	912	8
Woman	2064.5	791.3	329	5	110	91	19	0.326	646	9

6 RESULTS AND DISCUSSIONS

This section presents the OWR outcomes across various wire shapes that exhibit varying levels of geometric intricacy. We evaluate our algorithm's performance across diverse parameter configurations. Furthermore, we conduct comparisons with multiple baseline methods to confirm our algorithm's effectiveness. Physical assessments are conducted using our DIY wire-bending machine.

6.1 Implementation and Parameters

Our algorithm is developed using C++ and incorporates several third-party libraries including CGAL [Fabri and Pion 2009] for geometric computations, libhgp [Zhao 2023], libigl [Jacobson et al. 2018] and Eigen [Guennebaud et al. 2010] for tasks related to linear algebra, matrices, and vectors, and gco-v3.0 [Veksler and Delong 2015] for graph cut optimizing. We run our program on a PC with an Intel Core i7-13700 CPU operating at 2.10 GHz and 32 GB of memory. In candidate fabricable segments generation phase, we set the fitting error threshold ϵ to 0.6 mm. In graph-cut optimization, we set the λ_1 of the data term to 30, λ_2 of the label term to 50. Within the beam search, the beam width W_{Beam} is set to 5, and the probability ω of designating a collision-free point as a tuned point is set to 0.3. The machine parameter L_{min} is set at 7 mm, R_{min} for interpolated bending is 20 mm, L_{in} for strike bending is 2 mm, and α_{max} is 110 °.

6.2 Optimizing Wire Reconfiguration results

OWR results for eight 2D models in Figure 9 and Figure 1, and we list the statistical data in Table 1 and the running time in Table 2. As shown, our algorithm successfully maintains a low fitting error, averaging 0.38675 mm across the eight results. In these cases, the upper limit for tuned points is 9, considered suitable for human-bending stage. Typically, the required quantity of tuned points escalates in correlation with the incidence of self-intersections within the wire, exemplified by the *Leaf* model, which has 9 tuned points with 3 self-intersections. This is due to the fact that resolving these self-intersections is essential for maintaining collision-free during the machine-bending stage, consequently leading to an increase in the number of tuned points. We validate our algorithm further using multi-view wire art and complex 2D shapes (Appendices P and Q).

Table 2. *Program running time of each step.* Fit is the time of getting candidate bending segments sequences (s). *GR* is the time of graph-cut and re-fitting (s). *Sc* indicates the time to set the score for each bending point (s). *BS* indicates the time of beam search (s). *CRO* is the time of running CRO algorithm (s). $\#N_{CRO}$ is the number of CRO executions. *Algo* is the total time of running the algorithm (s), which is the sum of Fit, *GR*, *Sc*, *BS* and *CRO*. *Mac* is the bending time of wire-bending machine (min). Hum is the bending time of human (min). Fab is the total time of fabrication (min), which is the sum of *Mac* and Hum.

	Sec. 4		Sec. 5					Fabrication		
Model	*Fit*	*GR*	*Sc*	*BS*	*CRO*	$\#N_{CRO}$	*Algo*	*Mac*	*Hum*	*Fab*
Bird	5.01	0.32	2.53	0.58	9.52	104	17.96	53.37	2.73	56.10
Bull	2.03	0.24	0.35	0.47	42.92	426	46.01	56.28	5.87	62.15
Cat	3.20	0.25	1.01	0.10	1.63	136	6.19	41.60	2.38	43.98
Dolphin	4.43	0.27	5.46	0.30	4.18	146	14.64	51.30	2.80	54.10
Leaf	3.03	0.24	3.06	1.10	147.99	1346	155.42	66.25	6.20	72.45
Ma	0.95	0.09	2.35	0.17	4.25	134	7.81	34.62	3.42	38.04
Motorbike	2.55	0.29	4.38	3.43	303.36	1656	314.01	71.57	6.13	77.70
Woman	2.83	0.25	0.55	0.16	4.76	566	8.55	49.12	5.48	54.60

Performance. Our algorithm takes 71.32 seconds on average for the eight examples in Table 2. It is noted that execution time escalates with an increase in bending segments, leading to a higher number of tuned points, exemplified by models like *Leaf* and *Motorbike*. It is readily apparent that the primary efficiency bottleneck of our algorithm is due to CRO, requiring an average of 64.83 seconds for the eight models. This arises because the beam search process triggers numerous CROs, 564 on average, despite the rapid execution of individual CROs, 0.11 seconds on average.

Experiments. We also perform experiments on hyperparameters such as the fitting error threshold (Figure 13), different types of fitting segments (Figure 9), minimum segment length, wire size, and bending order. In addition, we test the algorithm on scalability (Figure 15), 3D shapes, and failure case (Figure 16). All details are provided in Appendix I.

6.3 Physical evaluation

Based on [Dejan 2018], we constructed a desktop-level wire-bending machine that had significant errors in the length of the wire feed and the angle of bending, reaching 30% − 50%, due to the materials of the machine parts. Therefore, we made several improvements to alleviate this problem. The modified machine is shown in Figure 4, with extremely improved accuracy. The wire used is aluminum, with a diameter of 1.5 mm and a Vickers Hardness of 20 HV. We use G-code [Arduino 2024] to translate $\mathcal{W}^*$ into wire-bending machine, with a Arduino MEGA2560 R3 development board. Detailed machine improvements and hardware setup are included in Appendix J.

As shown in Figure 18, to test the effectiveness of our method, we engaged a group of novices to participate in the "Machine-And-Then-Human-Bending " strategy to create wire art samples. These participants, despite their lack of prior experience in wire bending, were able to easily bend tuned points to the specified angles with the help of a goniometer. For each tuned point, they first set the goniometer to the target angle, then align the tuned point with the goniometer's origin, and finally, bend the wire by hand to achieve the desired angle. Compared the Machine-And-Then-Human-Bending wire with the input wire art in Figure 14, we can see that the proposed method achieves our goal that no skilled person can obtain the desired shape in collaboration with bending machine. Although due to the error of bending machine, the shape has some deviation from the initial shape. Further discussion of our physical evaluation can be found in Appendix K.

6.4 Comparisons

Assembly-based bending vs. two-stages-bending. For a given non-fabricable wire $\mathcal{W}$, the two-stages-bending method presented in this paper offers a method for bending wire without assembly, ensuring it is free from collisions by incorporating tuned points. In contrast, [Lira et al. 2018] suggests that decompose $\mathcal{W}$ into fabricable Eulerian wires, manufacture individual wire separately, and then assemble them with connectors to create the final wire product. Figure 17 compares the assembly-based bending approach with the two-stages-bending method using the same model. In our approach, we use a single continuous wire with 13 tuned points to fabricate the entire wire art, while Lira's method employs six separate wires. This eliminates the need for connectors to join individual wire segments.

Human-only vs. two-stages-bending. We recruited two novice craftsmen with no prior experience in wire bending. As illustrated in Figure 19, each participant initially shapes a wire (a) into the desired configuration totally manually (b), subsequently completing the human-bending stage following the machine-bending stage provided by our technique (c). Figure 19 demonstrates that our method significantly surpasses the "human-only" strategy in the precision of the result wire. Both of them indicate the difficulty in ensuring the accuracy of angles and circular segment during human-only bending, whereas our method simplifies the process significantly. As for the fabrication efficiency, it only takes 2.25 min and 2.80 min for the craftsmen to complete the human-bending stage, which takes 24.65 min and 37.32 min of the "human-only" strategy. Although machine-bending stage requires more time than "human-only" approach in our existing DIY wire-bending machine, we consider it suitable for an automated bending process and anticipate that it can be accelerated with advanced wire-bending machines. The experiment shows that our method achieves smaller errors and improves the user experience to produce wire sculptures with intricacy and geometric details.

7 DISCUSSION, LIMITATION AND FUTURE WORK

This paper presents a computational approach to fabricate wire sculptures with intricate geometric details, which is implemented by the proposed Machine-And-Then-Human-Bending strategy. The key technique challenge in this paper lies in the generation of a collison-free fabricable tuned wire from the input wire, with a minimum number of tuned points. The method we develop contains a segment fitting strategy and a bilevel optimization strategy to make the formulation OWR tractable. Physical evaluation is applied to validate the proposed two-stages-bending strategy with a set of various wire sculptures.

Limitation. Our current solution for the OWR problem is purely geometric. We do not consider physical factors such as gravity, wire stability, or other functional aspects of actual wire products. During machine-bending stage, the wire may sag due to its weight, leading to unintended collisions and manufacturing errors. We also do not

assess wire stability or strength while searching for tuned points to generate a collision-free tuned wire. Additionally, our method may generate an excessive number of tuned points for certain input wires, which can require a potentially consuming effort from humans to complete the follow-up bending process.

Future work. We propose several directions for future work in the field of wire sculpture fabrication. First, we suggest incorporating physical simulation into the wire reconfiguration planning process to enhance the accuracy and realism of the wire sculptures. Second, we recommend generalizing the proposed two-stages-bending strategy for wire assembly by decomposing complex wire sculptures into multiple tuned wire paths. Third, recognizing the limitation of not considering physical factors suggests promising directions for future work. For example, one can take the weight of the wire into account to compensate for potential deformation during bending. Fourth, due to the material elasticity of the wire sculpture, it may be beneficial to allow for a certain amount of collision during the machine-bending stage, transforming the hard constraint of collision-free into a soft constraint. Fifth, to achieve a continuously curvature-changing wire, consider exploring the simultaneous bending strategy, which involves moving both the wire and the bin setup simultaneously. Sixth, optimize the bending sequence for multiple pins bending machines. Seventh, optimize the distribution of sampling points during the segment-bendable wire generation.

Furthermore, it promises to expand our solution to the robotics domain, allowing optimal reconfiguration and reduced energy consumption in general graph-like robot movement. While this paper primarily focuses on wire sculpture fabrication, we believe that exploring wire abstraction design is also an intriguing avenue to explore, especially using the AIGC techniques.

ACKNOWLEDGMENTS

We thank all reviewers for their valuable comments and constructive suggestions. We extend our gratitude to Ralph Martin, Jiaye Wang, Oliver Deussen, Daniel Cohen-Or, and Dani Lischinski for their insightful advice on our project. We also thank Pengbin Tang and Shiqing Xin for their guidance on CRO, and Fengchi He for providing support in modifying the wire bending machine. The authors thank Zhenmin Zhang, Yang Wang, and Shuai Feng for their assistance in completing the photo and human-bending experiments, Kenji Tojo and Nobuyuki Umetani for providing models of multi-view wires. This work was supported by the National Key R&D Program of China (2022YFB3303200), the National Natural Science Foundation of China (U23A20312).

REFERENCES

Arduino. 2024. Arduino. (2024). https://www.arduino.cc/en/software

Andrea Baraldo, Luca Bascetta, Fabrizio Caprotti, Sumit Chourasiya, Gianni Ferretti, Angelo Ponti, and Basak Sakcak. 2022. Automatic computation of bending sequences for wire bending machines. *International Journal of Computer Integrated Manufacturing* 35, 12 (2022), 1335–1351.

Harsh G Bhundiya and Zachary C Cordero. 2023. Bend-Forming: A CNC deformation process for fabricating 3D wireframe structures. *Additive Manufacturing Letters* 6 (2023), 100146.

Bernd Bickel, Paolo Cignoni, Luigi Malomo, and Nico Pietroni. 2018. State of the art on stylized fabrication. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 325–342.

Dejan. 2018. Arduino 3D Wire Bending Machine. (2018). https://howtomechatronics.com/projects/arduino-3d-wire-bending-machine/?utm_content=cmp-true

Andreas Fabri and Sylvain Pion. 2009. CGAL: The computational geometry algorithms library. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*. 538–539.

Akash Garg, Andrew O Sageman-Furnas, Bailin Deng, Yonghao Yue, Eitan Grinspun, Mark Pauly, and Max Wardetzky. 2014. Wire mesh design. *ACM Transactions on Graphics* 33, 4 (2014).

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

Feili Hou and Wei-Min Shen. 2010. On the complexity of optimal reconfiguration planning for modular reconfigurable robots. In *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2791–2796.

Emmanuel Iarussi, Wilmot Li, and Adrien Bousseau. 2015. WrapIt: computer-assisted crafting of wire wrapped jewelry. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 1–8.

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

Wallace Lira, Chi-Wing Fu, and Hao Zhang. 2018. Fabricable eulerian wires for 3D shape abstraction. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–13.

Min Liu, Yunbo Zhang, Jing Bai, Yuanzhi Cao, Jeffrey M Alperovich, and Karthik Ramani. 2017. WireFab: mix-dimensional modeling and fabrication for 3D mesh models. (2017), 965–976.

Ruishuang Liu, Weiwei Wan, and Kensuke Harada. 2023. TAMP for 3D Curving—A Low-Payload Robot Arm Works Aside a Bending Machine to Curve High-Stiffness Metal Wires. *IEEE Transactions on Automation Science and Engineering* (2023).

Jonàs Martínez, Jérémie Dumas, and Sylvain Lefebvre. 2016. Procedural voronoi foams for additive manufacturing. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–12.

Wojciech Matusik and Adriana Schulz. 2019. Computational fabrication. In *ACM SIGGRAPH 2019 Courses*. 1–305.

Eder Miguel, Mathias Lepoutre, and Bernd Bickel. 2016. Computational design of stable planar-rod structures. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–11.

Koya Narumi, Kazuki Koyama, Kai Suto, Yuta Noma, Hiroki Sato, Tomohiro Tachi, Masaaki Sugimoto, Takeo Igarashi, and Yoshihiro Kawahara. 2023. Inkjet 4D print: Self-folding tessellated origami objects by inkjet UV printing. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–13.

Frank R Schmidt, Eno Toppe, and Daniel Cremers. 2009. Efficient planar graph cuts with applications in computer vision. In *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 351–356.

Peng Song, Chi-Wing Fu, Yueming Jin, Hongfei Xu, Ligang Liu, Pheng-Ann Heng, and Daniel Cohen-Or. 2017. Reconfigurable interlocking furniture. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–14.

Kenji Tojo, Ariel Shamir, Bernd Bickel, and Nobuyuki Umetani. 2024. Fabricable 3D Wire Art. In *ACM SIGGRAPH 2024 Conference Proceedings (SIGGRAPH '24)*. https://doi.org/10.1145/3641519.3657453

Cesar Torres, Wilmot Li, and Eric Paulos. 2016. ProxyPrint: Supporting crafting practice through physical computational proxies. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*. 158–169.

Nobuyuki Umetani, Bernd Bickel, and Wojciech Matusik. 2015. Computational tools for 3D printing. *SIGGRAPH courses* 9 (2015).

Olga Veksler and Andrew Delong. 2015. gco-v3.0. https://github.com/nsubtil/gco-v3.0. (2015).

Yinan Wang, Xi Yang, Tsukasa Fukusato, and Takeo Igarashi. 2019. Computational design and fabrication of 3D wire bending art. (2019), 1–2.

Wikipedia. 2024. Wire Sculpture. (2024). https://en.wikipedia.org/wiki/Wire_sculpture

Rundong Wu, Huaishu Peng, François Guimbretière, and Steve Marschner. 2016. Printing arbitrary meshes with a 5DOF wireframe printer. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–9.

Hongyi Xu, Espen Knoop, Stelian Coros, and Moritz Bächer. 2018. Bend-it: design and fabrication of kinetic wire characters. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–15.

Zhijin Yang, Pengfei Xu, Hongbo Fu, and Hui Huang. 2021. WireRoom: model-guided explorative design of abstract wire art. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–13.

Minjing Yu, Zipeng Ye, Yong-Jin Liu, Ying He, and Charlie CL Wang. 2019. Lineup: Computing chain-based physical transformation. *ACM Transactions on Graphics (TOG)* 38, 1 (2019), 1–16.

Ya-Ting Yue, Xiaolong Zhang, Yongliang Yang, Gang Ren, Yi-King Choi, and Wenping Wang. 2017. Wiredraw: 3d wire sculpturing guided with mixed reality. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 3693–3704.

Yuxiao Zhang, Jin Wang, Dongliang Zhang, and Guodong Lu. 2023. Foldable chain-based transformation method of 3D models. *Complex & Intelligent Systems* (2023), 1–18.

Yuxiao Zhang, Jin Wang, Dongliang Zhang, Guodong Lu, and Long Chen. 2022. Construction and Deformation Method of 3d Model Based on Rubik's Snake. *Available at SSRN 4136041* (2022).

Haisen Zhao. 2023. libhgp. https://github.com/haisenzhao/libhgp. (2023).

Haisen Zhao, Fanglin Gu, Qi-Xing Huang, Jorge Garcia, Yong Chen, Changhe Tu, Bedrich Benes, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2016. Connected fermat spirals for layered fabrication. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–10.

Fanchao Zhong, Haisen Zhao, Haochen Li, Xin Yan, Jikai Liu, Baoquan Chen, and Lin Lu. 2023. VASCO: Volume and Surface Co-Decomposition for Hybrid Manufacturing. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–17.

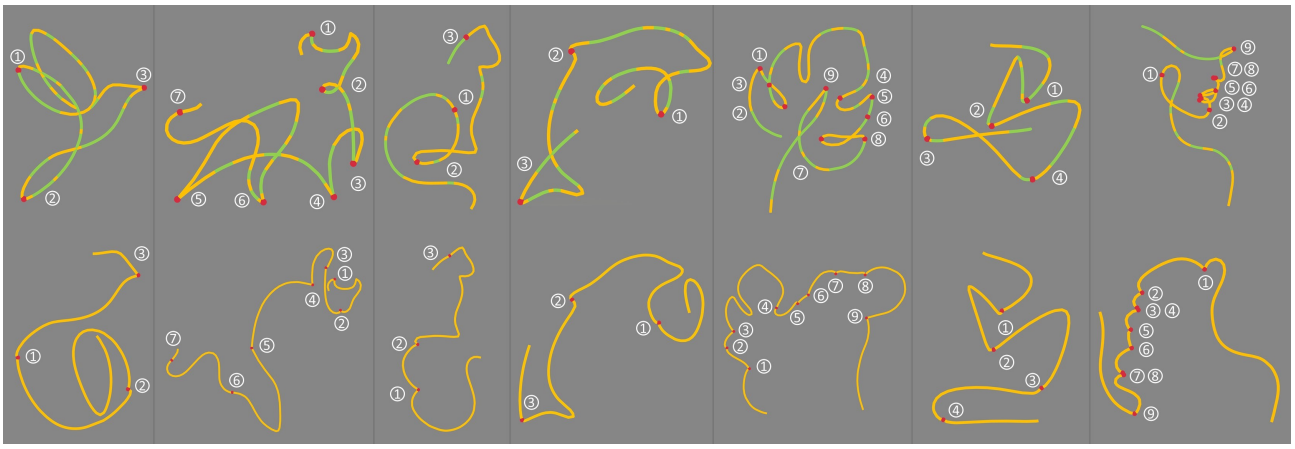


Fig. 9. Results gallery generated by our algorithm. The models are arranged in the order of *Bird*, *Bull*, *Cat*, *Dolphin*, *Leaf*, *Ma,i.e. the simplified Chinese character of horse*, *Woman*. We show both the fitting result (line segment: Green, circular segment: Yellow) and the deformed wire after bending the tuned points.

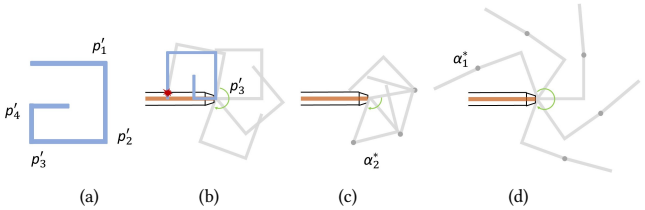


Fig. 10. Feasible bending angle range. (a) shows the input wire. (b) Collision occurs when bending p_3^* to α_3'. The green arrows indicate the feasible bending angle range for p_3^*, and the gray shapes are several particular options in the range. (c) Changing α_2^* makes the feasible search space of p_3^* smaller. (d) Changing α_1^* makes the feasible search space of p_3^* larger.

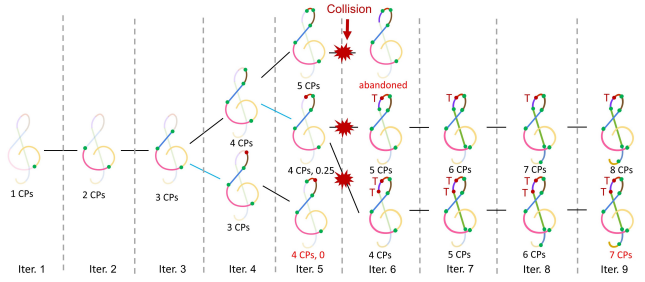
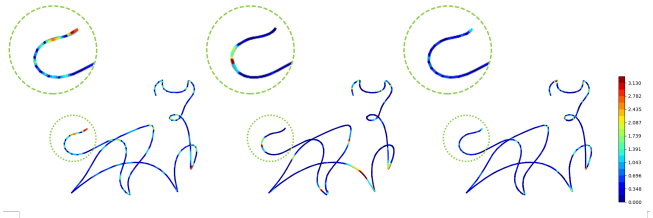


Fig. 11. An example of our beam search algorithm. We set the beam width $W_{Beam} = 2$ in this demo. The blue line indicates that the random value is smaller than ω during the expanding stage, thus expanding a node by setting it as a tuned point. The CPs means the number of constant points and the score means the sum of $\mathcal{L}(p_j^*)$. A collision is detected in the 6-th iteration. Three different nodes run CRO algorithm separately and calculate different tuned points determination solutions. "T" indicates that the angle of the point is not equal to the target angle after CRO optimization.



(a) Pure line segments (b) Pure circular segments (c) Combined strategies

Fig. 12. Ablation experiment on different fitting strategies. (a): Only use line segments to fit, results in 0.682 mm fitting error. (b): Only use circular segments to fit input wire, results in 0.908 mm fitting error. (c): our method takes two strategies simultaneously, only results in 0.499 mm fitting error.

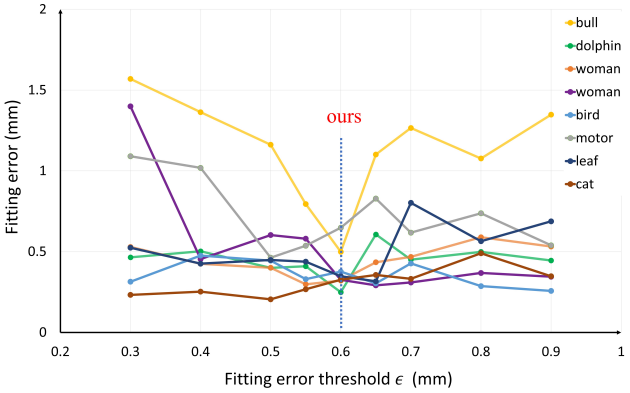


Fig. 13. Illustration of the relationship between the fitting error and ϵ. Small ϵ leads to short segments that require extensive re-fitting, increasing error. In contrast, a large ϵ inherently has a significant error. The number of tuned points is almost the same (7 vs. 8 vs. 7).

Fig. 14. Results gallery generated by our Machine-And-Then-Human-Bending strategy. The models are arranged in the order of *Bird*, *Woman*, *Cat*, *Ma,i.e. the simplified Chinese character of horse*, *Leaf*, *Dolphin*, *Bull*. Each set of images includes photos of the machine bending completed (golden) and the human bending completed (golden). The small image (white) in the corner of each photo is the theoretical rendering of that image. These photos use the maximum filter with parameter 2.8 for visualization, but the process does not change any shape.

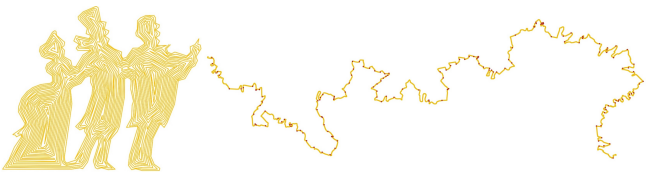


Fig. 15. Illustration of the scalability experiment of *Three-People* curve [Zhao et al. 2016]. The left side image shows the fitted shape and the right side shows the result after running the algorithm. With such a complex input, our algorithm takes about 61 hours to generate 160 tuned points.

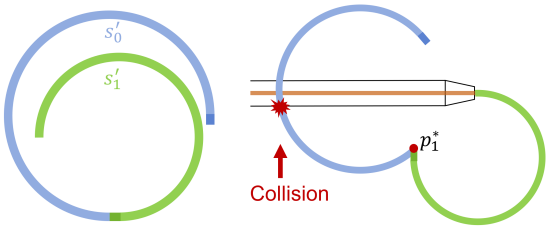


Fig. 16. Failure Case. (a) Input wire. (b) The angle value of the single bending angle α_1^* of p_1' is set to the maximum bending angle α_{max} and there is still a collision, so it is not possible to generate a collision-free wire W^*.

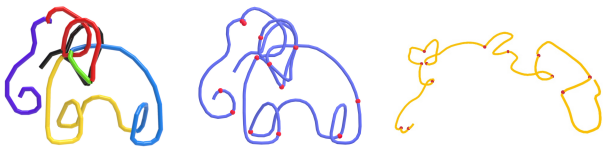


Fig. 17. Assembly-based bending vs. two-stages-bending experiment. (a) The elephant model decomposed into 6 wires generated by [Lira et al. 2018]. (b) The elephant model utilized a single, continuous wire, with 13 tuned points (red). (c) The tuned wire.

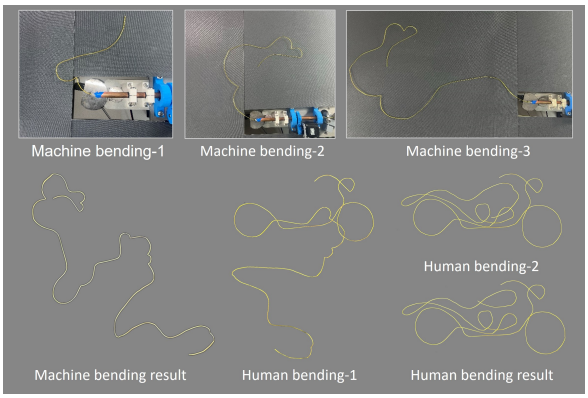


Fig. 18. Illustration of Machine-And-Then-Human-Bending result. The top figures show the machine bending process. The bottom figures demonstrate the human bending process.

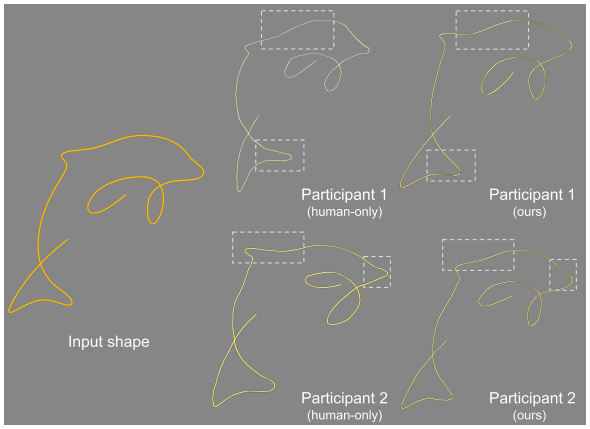


Fig. 19. Illustration of the comparison of the human-only vs. two-stages-bending. (a) The input shape model dolphin. (b) The "human-only" strategy results of two novice craftsmen. (c) The Machine-And-Then-Human-Bending strategy results of two novice craftsmen. The comparison of details (in the white box) shows that the shape produced by our method is more accord with the input shape in angle accuracy, shape smoothness and so on.