Computational design and fabrication of reusable multi-tangent bar structures

Yijiang Huang[a,*], Ziqi Wang[a,b,*], Yi-Hsiu Hung[a], Chenming Jiang[a], Aurèle Gheyselinck[a], Stelian Coros[a]

[a] *ETH Zurich, Zurich, Switzerland*
[b] *Hong Kong University of Science and Technology, Hong Kong, China*

Abstract

Temporary bar structures made of reusable standardized components are widely used in construction, events, and exhibitions. They are economical, easy to assemble, and can be disassembled and reused in various structural arrangements for various purposes. However, existing reusable temporary structures are either limited to modular yet repetitive designs or require bespoke components, which restricts their reuse potential. Instead of designing bespoke kit of parts for limited reuse, this paper investigates how to design and build diverse freeform structures from one homogeneous kit of parts. We propose a computational framework to generate multi-tangent bar structures, a widely used jointing system, which allows bars to be joined at any point along their length with standard connectors. We present a mathematical formulation and a numerical scheme to optimize the bar spatial positions and contact assignment simultaneously, while ensuring that the constraints of tangency, collision, joint connectivity, and bar length are satisfied. Together

*Corresponding author and equal contribution
Email addresses: `yijiang.huang@inf.ethz.ch` (Yijiang Huang), `ziqi.wang@epfl.ch` (Ziqi Wang)

with simulated case studies, we present two physical prototypes that reuse the same kit of parts using an augmented reality-guided assembly workflow.
Keywords: Reuse, kit of parts, computational design, space frames, multi-tangent, joints, mixed reality, assembly

---

1. Introduction

This paper concerns the design and rapid making of freeform, temporary bar structures with limited material resources that can be reused across several service cycles. Temporary bar structures are used broadly and in great variety in architecture, construction, engineering, and arts. Built out of kits of parts consisting of linear elements and joints to connect them, 3D bar structures can have complex geometries and topologies, for reasons related to structural efficiency, aesthetics, site conditions, or functional constraints.

Unambiguously defining a spatial bar network requires specifying the lengths of each bar and its relative pose to its neighbors. Existing bar systems usually address this by cutting bars to unique lengths and using customized joints to encode the spatial orientations. Examples include bespoke ball-and-socket joints [4] and 3D printed connectors [16, 14]. However, due to their customized geometries, these complex joints require long fabrication time and high costs. Moreover, such joint customization also limits the part kit's design possibility and reuse potential, since the parts are "frozen" once fabricated and their geometric and topological information are engrained into the bar length and joint configurations. In this work, our aim is to design and build structures that can adapt to rapidly changing needs, but with minimal virgin materials and manufacturing efforts. In contrast to using a bespoke kit of parts for each bespoke structure, we propose using a standardized, mass-produced kit for designing many bespoke structures (fig. 1).

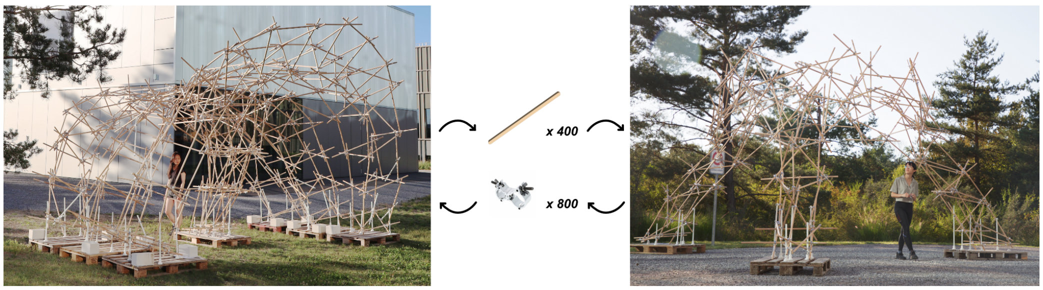

Figure 1: Diverse freeform multi-tangent structures can be built from a homogeneous kit of parts consisting of bars and standard connectors. Bars and connectors are assembled, disassembled, and reused among structures across service cycles.

We propose a new computational framework to augment an existing jointing system, called a *multi-tangent structure*, to achieve freeform structures. A multi-tangent bar system offsets bars in tangent contact with one or more other bars, which are then joined through reusable connectors. From indigenous building cultures to modern construction standards, such systems are widely used to rapidly build spatial trusses where mass customization of joints is not economical. Common examples include rope ties for bamboo structures, wire ties for rebar cages, and construction scaffolding (fig. 2). Because no custom joints are required and bars can remain uncut, no physical trace will be left on the kit once disassembled, and the kit can be reused to build diverse structures. However, the multi-tangent structures used in practice are mostly designed manually and prior research on computationally designing them restricts design freedom to a small subset of possible topologies, e.g., rectilinear or reciprocal patterns. We are inspired by the few existing freeform multi-tangent structures, e.g., the temporary bamboo theaters in Hong Kong (fig. 2-2) and scaffolding sculptures (fig. 2-4). However, to discover new functional and aesthetic potentials of multi-tangent structures, a geometric problem will first arise: is it possible to map an arbitrary design intention (represented as a line graph) to a multi-tangent structure?

Figure 2: Examples of multi-tangent bar structures: (1) rope-lashed bamboo roof structures, (2) a temporary theater in Hong Kong made with bamboos, (3) construction scaffolding connected by swivel couplers, (4) a sculpture made out of scaffolding elements and couplers, by artist Ben Long.

Generalizing the multi-tangent systems to freeform structures requires a systematic approach to convert arbitrary design intention to conform to the geometric constraints of the system. The computational problem involved is challenging because it consists of both *discrete* variables, assignments to indicate which bars are in contact, and *continuous* variables, the position of the bar axes. A feasible multi-tangent design must satisfy constraints that couple the discrete and continuous variables: pairs of bars with assigned contact should stay tangent, while pairs of bars with unassigned contact must not collide.

To address these challenges, we propose a novel mathematical formulation and a numerical solver to explore the design space of multi-tangent bar structures. With an arbitrary initial geometry and topology of the structure given as input, our method optimizes the bar spatial positions and contact assignment simultaneously, while ensuring that the tangency, collision, joint connectivity, and bar length constraints are satisfied among bars and connectors. The generated structures can use multiple types of physical connectors, including metal wire ties, ropes, zip ties, and swivel couplers. For swivel couplers, we provide an extension on the formulation to accommodate additional collision constraints for these bulkier but more reliable connectors. To test its applicability to real structures, we apply our method to design two human-scale pavilions made of standardized, off-the-shelf wooden bars and swivel couplers.

Our core contributions include the following:

- A mathematical optimization formulation for designing free-form, multi-tangent structures that use reusable, standardized bars and connectors. Our formulation transforms the originally intractable bi-level mixed-integer problem into a sequence of mixed-integer linear programming (MILP) subproblems that can be solved by off-the-shelf tools.

- Modeling of practical considerations as linear constraints in the MILP formulation, such as bar tangency and collisions, joint connectivity, maximum bar length bounds, and clamp collisions.

- An efficient numerical algorithm that adopts a trust-region-like outer optimization scheme with the linearized MILP as sub-steps.

- Validation of our design algorithm on a variety of shapes in simulation, and two human-scale physical realizations of the generated multi-tangent designs that reuse the same kit of parts.

1.1. Multi-tangent structures

Multi-tangent structure, in its most general form, is an efficient way to join together linear or curved elements, in which elements are jointed not necessarily at their ends, but at any point along their length. Computational design of such structures has been studied in many physical forms in the computer graphics literature. Examples include curved networks [25], ribbon structures [30, 35], wire meshes [7], welded steel sculptures [22, 26], and structures made of planar bent rods [23, 20].

For multi-tangent systems with straight linear elements, structures with reciprocal patterns have received most attention in research. Reciprocal frames (RF) consist of multiple reciprocal units, in which three or more sloped bars form a closed circuit by having the inner end of a bar resting on and supported by its adjacent bars. For many centuries, RFs have been used in design and construction, including Leonardo Da Vinci's bridge sketch in *Codex Atlanticus*, the roof of Nagasaki Castle in Japan, and Inuit tents [17]. Due to their intrinsic beauty and their potential as a cost-effective deployable system [17], Pugnale et al. [29] stressed the need for computational tools to discover RF designs. Traditionally, designers manually experiment with physical mock-up models to create RF structures [28, 33], which provides full design control, but the making process is time consuming and disconnected from digital design workflows.

Given a fixed RF-pattern, genetic algorithms has been used to offset the bars to achieve bar tangency [2, 27]. To enable a broader range of RF patterns, Song et al. [32] present a two-stage method that first generates the RF pattern and then perform geometric optimization to offset the bars.

The RF pattern is obtained by tessellating a plane with a user-specified RF unit, and then the pattern is mapped to the target surface. However, the geometric optimization models bar tangency and other design considerations as cost terms and uses unconstrained optimization to minimize the weighted sum. Because the bar tangency is not modeled as a hard constraint and collisions between bars are not captured, the resulting solution may have gaps between contact bars and collisions. In section 5.1, we will show a detailed comparison with this work and demonstrate that our work can enforce exact bar tangency and prevent collisions among bars.

In contrast to these top-down approaches that aims to digitally finalize a design before the assembly starts, recent work on human-robot collaborative assembly proposes to enable humans to make design decisions during construction while robots temporarily support floating bars [24, 1]. However, an RF pattern is fixed during the process and designers can only change the bar positions.

By definition, RF-patterns are only applicable to a manifold graph (a mesh) but little is known about the design possibilities beyond these 2.5D patterns. In [28], generative rules are developed to extend the RF concept beyond surface-based structures based on the making of physical models. Parascho et al. [26] propose a procedural generation logic based on tetrahedron cells to generate double-tangent structures, which means that each bar is tangent to two other bars at each of its ends. In contrast to these works that focus on specific contact patterns, our work aims to explore the geometric possibilities of converting any given line graph (including non-manifold ones) to multi-tangent structures by designing an algorithm to automatically choose the contact patterns and bar positions while enforcing practical constraints.

2. Modeling multi-tangent systems with optimization

2.1. Overview

Given a line graph as an input, our system allows the users to explore a multi-tangent realization of their design intention. The system automatically tries to compute a geometrically feasible multi-tangent configuration. A graphical overview of the design workflow is illustrated in fig. 3. Finding a feasible configuration is challenging due to the complexity of the design space: At each joint of the line graph, there is a combinatorial choice of the contact assignment, i.e., which bars are in contact with each other. Furthermore, the contact assignment is coupled with the bar axes' positions, which are continuous variables, through the tangency, collision-free, joint connectivity, and bar-length constraints. A graphical illustration of these constraints is provided in fig. 3. These constraints have a very distinct mathematical nature, including a nested constrained quadratic optimization problem, a local graph connectivity problem, and a discrete assignment problem.

In this section, we first introduce the mathematical model to describe a multi-tangent system, including the notation, the decision variables (section 2.2), and the constraints (section 2.3 - section 2.6). Then we explain the challenges of the general formulation (section 2.8), which motivates our new formulation and the solving strategy described in section 3 and section 4.

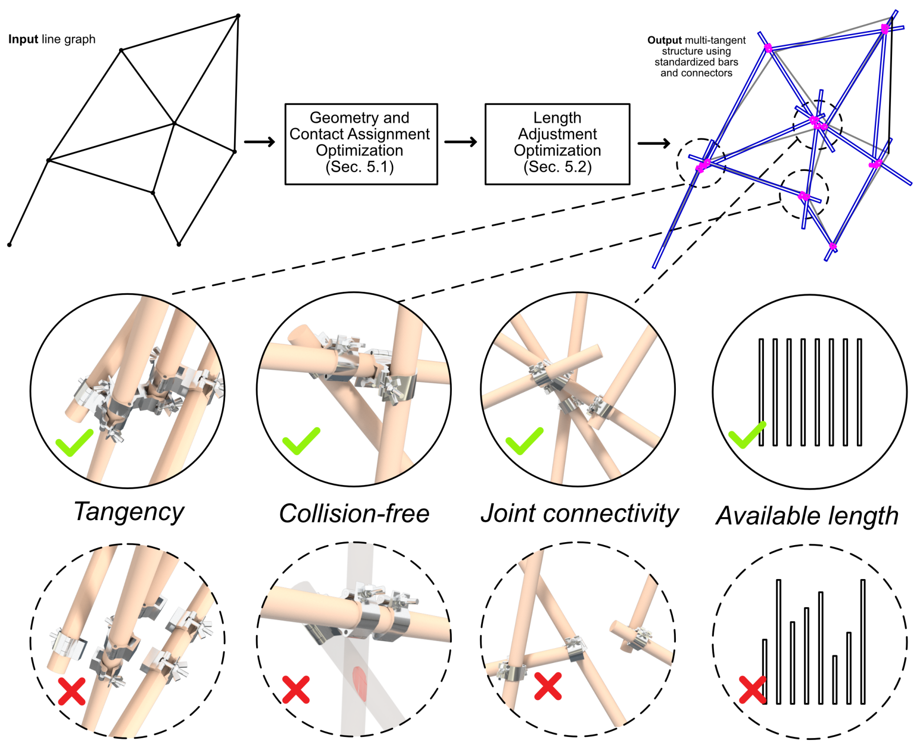

Figure 3: Overview of our workflow. The user provides a target line graph and a set of available bar lengths. The optimization algorithm automatically computes the bar axis positions and contact assignments for a multi-tangent system realization of the line graph, while ensuring tangency, collision-free, joint connectivity, and available bar length constraints are satisfied.

2.2. Decision variables

Given an undirected line graph $G = \langle V, E \rangle$ with vertex set V and edge set E, we aim to find a multi-tangent realization of this graph. The graph vertices are embedded in the 3D Euclidean space (i.e. $\mathbf{p}_v \in \mathbb{R}^3, \forall v \in V$). In a multi-tangent system, each edge in E will be converted to a linear bar element, which is allowed to connect to other elements in its connected neighborhood in G through contact at any point on its physical surface.

Mathematically, we want to convert each edge $e_i = \langle v, v' \rangle \in E$ in the original graph to a bar L_i, i.e., a cylinder of revolution about the straight line segment $\mathbf{x}_i^0 \rightarrow \mathbf{x}_i^1$, where $\mathbf{x}_i^0, \mathbf{x}_i^1$ are the endpoints of the bar axis to be determined (fig. 4), corresponding to v, v' respectively.

For each pair of connected edge e_i, e_j that shares a vertex v, i.e., $v = e_i \cap e_j$, we assign a binary variable $z_{i,j}^v$. Let $z_{i,j}^v = 1$ if the bar L_i has a joint connection with bar L_j, and $z_{i,j}^v = 0$ if the two bars do not have a connection (fig. 4).

In summary, a multi-tangent system can be determined by two types of variables:

- Continuous variables for bar axis's end points $\mathbf{x} = \{[\mathbf{x}_i^0; \mathbf{x}_i^1]^T \in \mathbb{R}^6 \mid e_i \in E\}$.
- Binary variables for joint assignment $\mathbf{z}$, whose entry $z_{i,j}^v \in \{0, 1\}$.

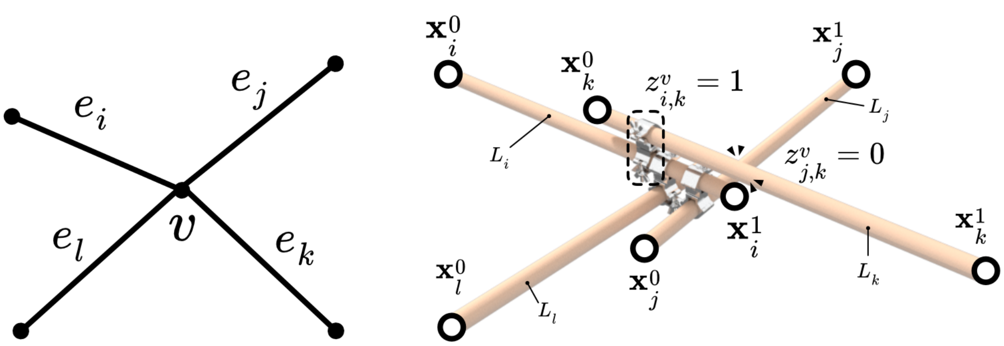

Figure 4: Design variables for a multi-tangent system. For edges e_i connected to the same node v in the input graph G, we aim to determine their bar realization L_i by finding the bar end points $\mathbf{x}_i^s$ and the joint assignment $z_{i,j}^v$ for each pair of edges $e_i, e_j \in N(v)$. In this example, we assign a joint between L_i and L_k by setting $z_{i,k}^v = 1$ (marked by the dashed area), and no joint between L_j and L_k by setting $z_{j,k}^v = 0$.

A feasible multi-tangent system must satisfy three types of constraints as illustrated in fig. 3: *tangency* (section 2.3), *collision* (section 2.4) and *joint connectivity* (section 2.5). In addition, an infinitely reusable multi-tangent system must conform to the *available length* constraint (section 2.6) to avoid cutting the part kit to fit a specific design.

2.3. Tangent constraint

Bar pairs with assigned joints need to be *tangent*, i.e., the distance between bars equals to the sum of bar radius (fig. 3-a):

$$d[L_i, L_j] = 2R + D_c, \text{ if } z_{i,j} = 1$$

where R is the bar radii and D_c is the thickness of the connector used. When the two bars are in direct contact, $D_c = 0$. $d[\cdot,\cdot]$ is a distance function that computes the shortest distance between two line segments of finite lengths, which involves the following constrained quadratic optimization:

$$d[L_i, L_j](t^v_{i,j}, t^v_{j,i}) = \min_{t^v_{i,j}, t^v_{j,i}} ||(\mathbf{x}^0_i + t^v_{i,j}(\mathbf{x}^1_i - \mathbf{x}^0_i)) - (\mathbf{x}^0_j + t^v_{j,i}(\mathbf{x}^1_j - \mathbf{x}^0_j))|| \quad (1)$$
$$s.t. \quad t^v_{i,j}, t^v_{j,i} \in [0, 1]$$

where $t^v_{i,j}, t^v_{j,i}$ are the unitized arc-length parameters for determining the contact point's projection on the bar's central axis.

2.4. Collision constraint

Pairs of bars without assigned joints must not collide (fig. 3-b). This means that the distance between the bars, computed using eq. (1), should be larger than the sum of the bar radius:

$$d[L_i, L_j] \geq 2R, \text{ if } z_{i,j} = 0$$

2.5. Joint connectivity constraint

For edges that are connected to the same vertex, their bar realization should form a connected component. An unconnected example can be seen at the bottom of fig. 3, where the bars form two separated components and therefore do not function as one structure. There are many ways to formulate this constraint, and we defer our particular choice of formulation for this constraint to section 3.4.

2.6. Available length constraint

We require an infinitely reusable multi-tangent system to only use bars from a pre-defined set of available bar lengths A:

$$||\mathbf{x}_i^0 - \mathbf{x}_i^1|| \in A, \forall e_i \in E$$

In contrast to previous work on availability-driven design that constrains design to use elements from a limited-sized inventory [5, 12], we do not restrict the number of bars for each length set in A. For example, if $A = \{0.5, 1.0\}$, a multi-tangent system can use any number of bars with length 0.5 and 1.0 meter, but not any other length. This conforms to the industrial setting where structural elements are manufactured and sold in a given catalog of lengths, and the designer can use any number of elements from the catalog to build a structure assuming the supply is always larger than the demand. Since we do not cut the bars to fit a specific design, the chosen set of bars can be disassembled after serving its purpose and returned to the material inventory without any physical trace left on the material. Thus, the inventory will maintain the same length distribution and thus could be reused to build other structures.

2.7. The general optimization formulation

Putting together all the constraints described above, we can formulate the optimization problem for finding a reusable multi-tangent system of an input line graph as:

$$
\begin{array}{lrr}
\text{Find } \mathbf{x}, \mathbf{z}, \mathbf{t} \text{ s.t.} & & (2) \\
d[L_i, L_j](t^v_{i,j}, t^v_{j,i}) = 2R + D_c, & \text{if } z^v_{i,j} = 1, \forall e_i \cap e_j = v & \text{Tangency} \\
d[L_i, L_j](t^v_{i,j}, t^v_{j,i}) \geq 2R, & \text{if } z^v_{i,j} = 0, \forall e_i, e_j \in E & \text{Bar collision} \\
- & & \text{Joint connectivity} \\
||\mathbf{x}^0_i - \mathbf{x}^1_i|| \in A & & \text{Available lengths} \\
\mathbf{x} \in \mathbb{R}^{6|E|},\ \mathbf{z} \in \{0,1\}^{N_{cp}},\ \mathbf{t} \in [0,1]^{N_{cp}} & &
\end{array}
$$

where $N_{cp} = \sum_{v \in V} C^2_{|N(v)|}$ represents the total number of potential number of contact pairs and $|N(v)|$ is the valence of each node.

Our goal is to find a feasible solution that satisfies all the constraints. The first two constraints on tangency and collision involve an inner-layer, box-constrained quadratic optimization of eq. (1) that binds the contact point parameter $\mathbf{t}$ to the end points of the bar axis $\mathbf{x}$ while gated by the joint assignment $\mathbf{z}$. The available length constraint is a discrete catalog constraint. All of these together make the optimization problem a bilevel, mixed-integer programming problem with combinatorial constraints that is challenging to solve. In the next section, we use some concrete examples to provide some further insight into the challenges of solving this problem, and how this is a significant departure from previous work on computational design of multi-tangent systems.

2.8. Challenges

To illustrate the challenging coupling between the geometric and contact assignment variables and the design complexity therein, fig. 5 provides some design solutions for star-shaped input graphs, which represent the simplest topological example of connecting multiple edges at a single graph node.

For planar star-shaped graphs like the ones fig. 5-a and b, heuristics based on reciprocity can be used to assign the contact between bars, and optimization can focus on the geometric realization of the bars. However, we show that the reciprocal pattern, widely used in practice [18] and explored in research [32], is only a small part of the design space. The second row under fig. 5-a and b shows two designs with more joints used than a typical reciprocal pattern, and they demonstrate new potentials for more structural rigidity and different aesthetic expressions using tectonics.

For nonplanar star-shaped graphs such as the one in fig. 5-c, the contact assignment is not trivial and has a significant impact on finding a feasible bar configuration. For example, under fig. 5-c, we show two designs with different contact assignments that use 8 and 12 joints, respectively. These very distinct solutions show the nontriviality of defining rules for finding a solution for a spatial graph that satisfies all the constraints mentioned in the previous section. Such complexity also explains why most of the previous work on multi-tangent systems focuses on 2.5D reciprocal structures that resemble a surface [32, 22], and the few that consider spatial networks follow specific aggregation-based generation rules [26].

Finally, given all the design possibilities displayed in fig. 5, these star-shaped graphs represent only a small neighborhood in an input graph that is commonly used in practice. In those cases, resolving the constraints in a global fashion is much more challenging. Thus, a systematic approach is needed to overcome the technical challenges mentioned above and allow users to explore the design space of multi-tangent systems for arbitrary input graphs. This goal motivates our new formulation and solving strategy described in the next two sections.

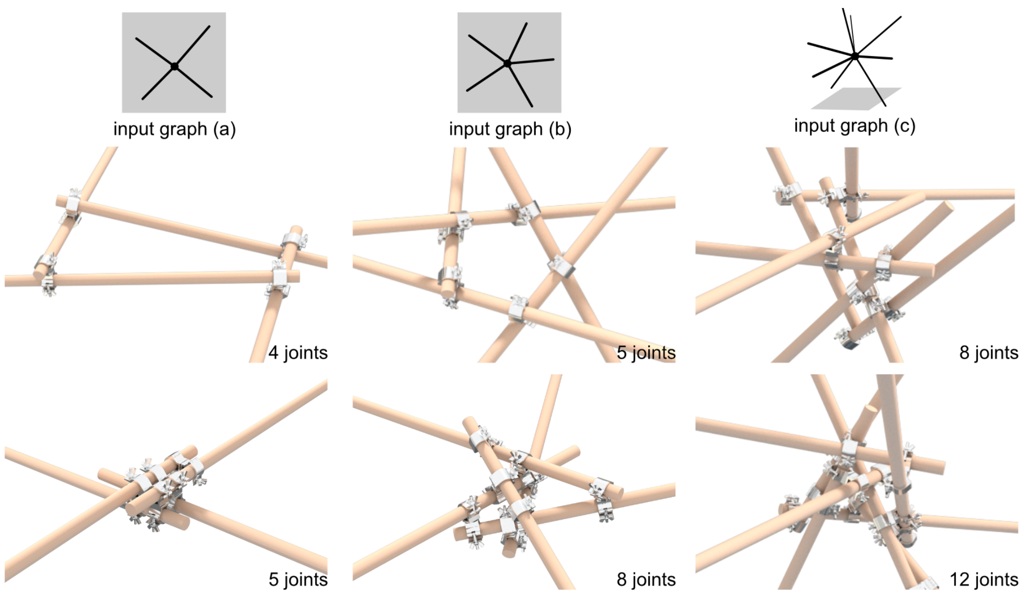

Figure 5: Multi-tangent design solutions for a valence-4 planar graph (a), a valence-5 planar graph (b), and a valence-8 spatial graph (c). Different contact assignments lead to drastically different bar configurations, which demonstrates the complexity of the multi-tangent design space.

3. New formulation: domain linearization for infinitely long bar elements

In this section, we describe modeling tricks to make the general optimization formulation described in section 2.7 tractable. The key insights are:

- Instead of finite bar elements, we model each bar as an *infinitely long cylinder* to simplify the distance computation in eq. (1). This turns the bi-level optimization problem into a single-level optimization problem. This also means that the length of the bars can be trimmed according to the available length set A *after* the optimization, and the available-length constraint can be removed from the optimization. (section 3.1)

- We use the first-order linearization of the tangency and collision constraints, and change the geometric variables from absolute positions $\mathbf{x}, \mathbf{n}$ to relative delta position changes $d\mathbf{x}, d\mathbf{n}$. This turns distance-related constraints from quadratic to linear. (section 3.2)

- New binary variables are introduced to model the top and down position between two bars in the tangent and collision constraints, which act like binary switches to help the optimization navigate disconnected feasible regions. (section 3.3)

- The joint connectivity constraint is modelled as a graph flow problem, which forms a set of linear constraints. (section 3.4)

- Connector locations on the same bar are constrained to ensure that bar length is smaller than the maximum available length. (section 3.5)

- A scalar variable is introduced to gradually inflate the bar radius towards the target radius R, which provides a curriculum with increasing difficulty and helps the convergence of the overall optimization problem. (section 3.6)

In addition, we provide an extension to model collision constraints between swivel couplers in section 3.7.

3.1. Modeling of infinitely long bar elements

The main challenges of the original formulation in section 2.7 are two-fold: (1) the bilevel nature of the problem caused by the distance computation embedded in the tangency and collision constraints, and (2) the combinatorial nature of the available length constraint. To overcome these challenges, we propose a conservative modeling trick to simplify the problem by modeling each bar as if they have infinite length in the optimization, and then trim the bar length as a post-processing step. This modeling is conservative because we ensure that collision is avoided for bars with infinite length, which is more restrictive than the actual case where the bars are finite.

Mathematically, the central axis of an infinite-length bar can be represented by a point $\mathbf{x}_i$ on the line and a vector $\mathbf{n}_i$. When not in parallel, the distance between two such bars can be simplified as an analytical expression:

$$d_\infty[L_i, L_j] = (\mathbf{n}_i/\|\mathbf{n}_i\| \times \mathbf{n}_j/\|\mathbf{n}_j\|)^T(\mathbf{x}_i - \mathbf{x}_j) \tag{3}$$

Compared to eq. (1) where we have to use a constrained quadratic optimization to compute an unsigned distance between two finite-length line segments, this closed-form expression can be computed directly. Unlike the unsigned distance in eq. (1), this is a signed distance depending on the relative position of the two bars (fig. 6). When the two bars are close parallel, eq. (3) degenerates, and we provide the numerical treatment for this special case in Appendix A.

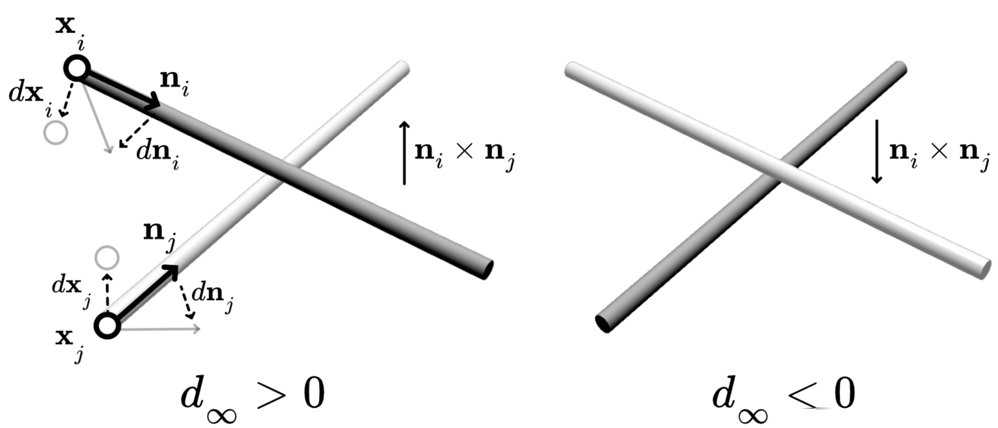

Figure 6: The distance computation for two infinitely long bars, each of which is modeled by a point $\mathbf{x}$ and a normal vector $\mathbf{n}$. The distance is signed and depends on the relative position of the two bars. After linearization, we solve for the change variables $d\mathbf{x}, d\mathbf{n}$, instead of the absolute position variables $\mathbf{x}, \mathbf{n}$.

3.2. Delta design variables and linearized distance computation

Despite its simplicity, the distance computation in eq. (3) is still nonlinear with respect to $\mathbf{x}$ and $\mathbf{n}$ and thus still leads to a nonconvex optimization problem that is hard to solve. So we make another important decision for modeling: instead of solving the mixed integer nonlinear optimization problem with the absolute position and normal variables $\mathbf{x}, \mathbf{n}$ in one shot as in eq. (2), we linearize the distance computation and iteratively solve for the *change variables* $d\mathbf{x}, d\mathbf{n}$ as a mixed-integer *linear* programming problem.

The rationale behind this is similar to the well-known trust-region method in nonlinear optimization ([9]-chap 11.6): for a given design $\mathbf{x}, \mathbf{n}$, we perturb the design with a small change $d\mathbf{x}, d\mathbf{n}$ in a trust region of size Δ and solve for the new design $\mathbf{x} + d\mathbf{x}, \mathbf{n} + d\mathbf{n}$ with a linearly approximated model, and then accept the new design if it improves the objective function. This process is repeated with a dynamically adjusted trust region until convergence. We delay the detailed description of the solving technique to section 4.

Formally, we switch from using the absolute variables $\{[\mathbf{x}_i^0; \mathbf{x}_i^1]^T \in \mathbb{R}^6 \mid e_i \in E\}$ to the change variable $\{[d\mathbf{x}_i; d\mathbf{n}_i]^T \in \mathbb{R}^6 \mid e_i \in E\}$ (fig. 6). We constrain the change variables to be within a small trust region Δ, and ensure that after the perturbation, the normal vector remains unit length:

$$-\Delta_k \leq d\mathbf{x}_i, d\mathbf{n}_i \leq \Delta_k, \forall e_i \in E \tag{4}$$

$$\mathbf{n}_i^k \cdot d\mathbf{n}_i = 0, \forall e_i \in E \tag{5}$$

where the trust region size Δ_k will be adjusted dynamically during the optimization process. $\mathbf{n}_i^k$ is the normal vector of the bar L_i computed from the previous optimization iteration, which is fixed during the current iteration.

The first-order Taylor approximation $\hat{d_\infty}$ of the distance function in eq. (3) is:

$$\begin{aligned}
&\hat{d_\infty}[L_i, L_j](d\mathbf{x}_i, d\mathbf{n}_i, d\mathbf{x}_j, d\mathbf{n}_i) := \\
&d_\infty(\mathbf{x}_i, \mathbf{n}_i, \mathbf{x}_i, \mathbf{n}_j) + \frac{\partial d_\infty}{\partial \mathbf{x}_i}^T d\mathbf{x}_i^0 + \frac{\partial d_\infty}{\partial \mathbf{n}_i}^T d\mathbf{n}_i + \frac{\partial d_\infty}{\partial \mathbf{x}_j}^T d\mathbf{x}_j + \frac{\partial d_\infty}{\partial \mathbf{n}_j}^T d\mathbf{n}_j
\end{aligned} \tag{6}$$

With $\mathbf{x}, \mathbf{n}$ fixed, this distance formula is linear with respect to the change variables $d\mathbf{x}, d\mathbf{n}$. When L_i and L_j are close to parallel, special treatments are needed to avoid numerical instability, which is provided in Appendix A.

3.3. Linearized tangent and collision constraints with side switches

Since we are using the signed distance, the original tangency and collision constraints for the unsigned distance in eq. (2) turn into absolute constraints:

$$
\begin{aligned}
&|d_\infty[L_i, L_j]| = 2R + D_c, && \text{if } z_{i,j} = 1 \\
&|d_\infty[L_i, L_j]| \geq 2R, && \text{if } z_{i,j} = 0
\end{aligned}
$$

To upper bound the absolute value of the distance between two bars L_i, L_j when they are tangent, we use the following linearized constraints:

$$\hat{d_\infty}[L_i, L_j] \leq 2R + D_c + M(1 - z^v_{i,j}) \tag{7}$$

$$\hat{d_\infty}[L_i, L_j] \geq -(2R + D_c) - M(1 - z^v_{i,j}) \tag{8}$$

where M is a large positive constant to only activate the constraint when $z^v_{i,j} = 1$, and $\hat{d_\infty}[L_i, L_j]$ is the linearized distance function in eq. (6). D_c is the thickness of the connector.

To lower bound the absolute value of the distance, since $|\hat{d_\infty}| \geq 2R + D_c$ specifies two disconnected feasible regions depending on the sign of $\hat{d_\infty}$:

$$
\begin{aligned}
&\hat{d_\infty} \geq 2R + D_c && \text{if } \hat{d_\infty} > 0 \\
&\hat{d_\infty} \leq -(2R + D_c) && \text{if } \hat{d_\infty} < 0
\end{aligned}
$$

we introduce a new binary variable $s^v_{i,j}$ to activate only one of the equations. Intuitively, $s^v_{i,j}$ models the up-down side of the bar L_i with respect to L_j (fig. 6), and helps the optimization algorithm to "jump" to the other feasible region when staying on one side is infeasible. Thus, we have the following linearized constraints for lower bounding the distance when the bars are tangent:

$$\hat{d_\infty}[L_i, L_j] \geq 2R + D_c - M(1 - s^v_{i,j}) \tag{9}$$

$$\hat{d_\infty}[L_i, L_j] \leq -(2R + D_c) + M s^v_{i,j} \tag{10}$$

Note that the constraints $\hat{d_\infty} \geq 2R + D_c$ are only activated when $s = 1$, and the constraints $\hat{d_\infty} \leq -2R - D_c$ are only activated when $s = 0$.

Similarly, we can model the collision constraints as follows:

$$\hat{d_\infty}[L_i, L_j] \geq 2R - M z^v_{i,j} - M(1 - s^v_{i,j}) \tag{11}$$

$$\hat{d_\infty}[L_i, L_j] \leq -2R + M z^v_{i,j} + M s^v_{i,j} \tag{12}$$

Here, the constraints $\hat{d_\infty} \geq 2R$ are only activated when $z = 0$ and $s = 1$, and the constraints $\hat{d_\infty} \leq -2R$ are only activated when $z = 0$ and $s = 0$.

3.4. Modeling joint connectivity as a commodity flow problem

To ensure bars that are connected to the same vertex in the input graph still form a connected component in its multi-tangent realization, we want to ensure that there exists a path connecting each pair of vertices in the connectivity graph. This is equivalent to finding a connected subgraph in the local connectivity graph at the joint v.

We introduce a joint connectivity graph $J(v)$, a fully connected graph with its vertex set $V_{J(v)} = \{L_i \,|\, e_i \in N(v)\}$ corresponding to each bar connected to v, and its edge set $E_{J(v)} = \{(L_i, L_j) \,|\, e_i, e_j \in N(v), i \neq j\}$ corresponding to all potential joint assignments (fig. 7-1). $N(v) \subset E$ denotes all the edges connected to v in the original linegraph G. $|V_{J(v)}|$ is equal to the valence of v in G. Then, we can use the joint assignment variables $z^v_{i,j}$ as the

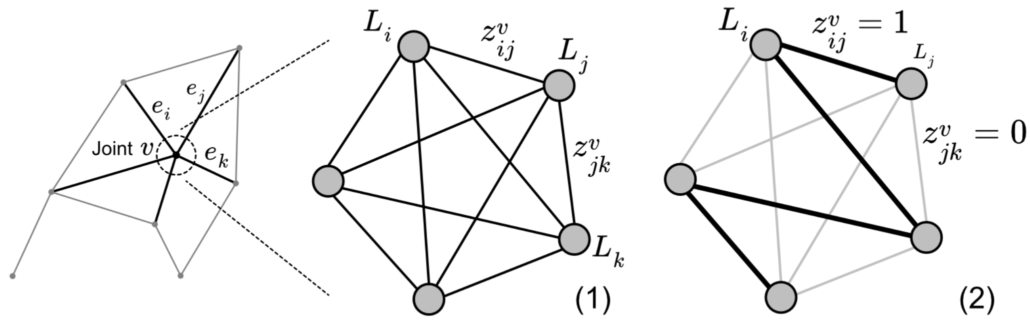

Figure 7: Joint connectivity graph representation. (1) The joint connectivity graph at a joint v with its vertices corresponding to each bar connected to v and edges corresponding to the joint assignments. (2) The joint assignment variables $z^v_{i,j}$ can be used as the indicator function to identify a subgraph. An example connected subgraph is shown with bold edges.

indicator function to identify a subgraph, where the edge is in the subgraph if and only if $z^v_{i,j} = 1$ (fig. 7-2).

Modeling graph connectivity appears in abundance in the literature on computational political districting [34]. In this work, we choose a simple variant that models it as a commodity flow problem [31], which can be expressed as a set of linear constraints:

$$0 \leq y^v_{i,j} \leq (|V_{J(v)}| - 1) z^v_{i,j}, \forall (i,j) \in E_{J(v)} \tag{13}$$

$$\sum_{j \in V_{J(v)}} y^v_{s,j} = |V_{J(v)}| - 1 \tag{14}$$

$$\sum_{j:(i,j) \in E_{J(v)}} y^v_{i,j} = \sum_{j:(i,j) \in E_{J(v)}} y^v_{j,i} - 1, \forall i \in V_{J(v)} \setminus \{s\} \tag{15}$$

$$y^v_{i,j} = y^v_{j,i} \tag{16}$$

where $y^v_{i,j}$ is the real-valued flow variable that indicates the commodity flow from L_i to L_j in the joint connectivity graph (fig. 8-1). Assuming that the graph is undirected, we assume that if (i, j) is an edge with flow variable $y^v_{i,j}$, (j, i) represents the same edge and $y^v_{j,i}$ also exists and is constrained to be equal.

The goal of these constraints is to ensure that there exists a feasible flow from an arbitrarily chosen source vertex $L_s \in V_{J(v)}$ that has a supply of $|V_{J(v)}| - 1$ unit of commodity to all other vertices, where each vertex has demand 1. An example of such flow is depicted in fig. 8-2. Equation (13) ensures that the flow is only allowed on the edges of the selected subgraph where the joint assignment is active ($z^v_{i,j} = 1$). Equation (14) states $|V_{J(v)}| - 1$ units of commodity are supplied from the chosen source vertex L_s. Equation (15) says that at every vertex, one unit of supply gets absorbed and anything left is passed along. Since we introduce a new flow variable $y^v_{i,j}$ for each edge in the fully connected joint connectivity graph around v, $|V_{J(v)}|\ (|V_{J(v)}| - 1)$ number of new variables will be introduced. Because the subgraph-gated flow constraint (eq. (13)) is assigned for each $y^v_{i,j}$, and the flow conservation constraint (eq. (14)) is assigned for each vertex $v \in V_{J(v)}$, the number of variables and constraints introduced are both quadratic to the valence of v in the original graph. We add this set of constraints for each vertex $v \in G$, so the total amount of constraints introduced is at the scale of $|V|\ |N(v)|^2$.

3.5. Maximum length of bar constraint

The maximum length of bar constraint ensures that the clamps on the same bar that are farthest apart do not exceed the longest element in the provided inventory, so that the post-processing step in section 4.2 can trim the infinitely long bar to the available length.

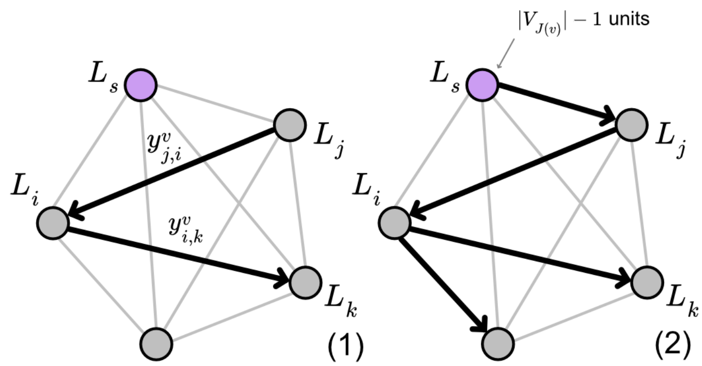

Figure 8: Commodity flow representation to model subgraph connectivity. (1) illustration of the directed flow variable $y^v_{i,j}$ on the undirected joint connectivity graph. (2) an example flow that connects the source vertex L_s to all other vertices.

To compute the distance between a pair of clamps on a bar, we will need to compute the arc-length parameter $t_{i,j}$. This parameter expresses the distance from the bar end point $\mathbf{x}_i$ of bar L_i to the its connector with L_j, which can be computed by:

$$T_{i,j} = [t_{i,j}, t_{j,i}]^T = \arg\min ||(\mathbf{x}_i + t_{i,j}\mathbf{n}_i) - (\mathbf{x}_j + t_{j,i}\mathbf{n}_j)||^2$$

where $t_{i,j}, t_{j,i}$ express the same connector's arc-length parameters on bar L_i and L_j, respectively. We only consider non-parallel pairs of L_i and L_j here, since in the parallel case we are free to choose t. Since this is an unconstrained, quadratic optimization problem, the optimum can be found by setting the gradient with respect to t to zero:

$$T_{i,j}(\mathbf{x}_i, \mathbf{n}_i, \mathbf{x}_j, \mathbf{n}_j) = \begin{bmatrix} \mathbf{n}_i^T\mathbf{n}_i & -\mathbf{n}_i^T\mathbf{n}_j \\ -\mathbf{n}_i^T\mathbf{n}_j & \mathbf{n}_j^T\mathbf{n}_j \end{bmatrix}^{-1} \begin{bmatrix} (\mathbf{x}_j - \mathbf{x}_i)^T\mathbf{n}_i \\ (\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{n}_j \end{bmatrix} \tag{17}$$

Recall that our decision variables are the change variables $d\mathbf{x}, d\mathbf{n}$ and the unknown arc-length parameters $T_{i,j}$. We use a first-order approximation to link them with $\hat{T}_{i,j}$ computed by using $\hat{\mathbf{x}}, \hat{\mathbf{n}}$ from the previous trust region iteration, by ignoring the higher-order terms in the Taylor expansion of $T_{i,j}$:

$$T_{i,j} - \hat{T}_{i,j}(\hat{\mathbf{x}}_i, \hat{\mathbf{n}}_i, \hat{\mathbf{x}}_j, \hat{\mathbf{n}}_j) - \frac{\partial T_{i,j}}{\partial \mathbf{x}_i}^T d\mathbf{x}_i - \frac{\partial T_{i,j}}{\partial \mathbf{n}_i}^T d\mathbf{n}_i - \frac{\partial T_{i,j}}{\partial \mathbf{x}_j}^T d\mathbf{x}_j - \frac{\partial T_{i,j}}{\partial \mathbf{n}_j}^T d\mathbf{n}_j = 0 \tag{18}$$

where all the partial derivatives are evaluated at $\hat{\mathbf{x}}, \hat{\mathbf{n}}$.

Finally, the maximum length of bar constraint can be formulated by ensuring that the arc-length parameters of the clamps on the same bar are smaller than the maximum available length l_{max}:

$$T_{i,j} - T_{i,k} \leq l_{max} + M(1 - z_{i,j}) + M(1 - z_{i,k}) \tag{19}$$

$$T_{i,k} - T_{i,j} \leq l_{max} + M(1 - z_{i,j}) + M(1 - z_{i,k}) \tag{20}$$

$$\forall e_i \cap e_j = v, e_i \cap e_k = v'$$

3.6. The linearized optimization formulation with radius curriculmn

Although directly solving the feasibility problem with all the linearized constraints described above should work in theory, in practice, we found that the optimization struggles to turn the initial guess into a feasible solution. To overcome this challenge, we introduce a strategy inspired by curriculum learning to gradually inflate the radius of the bar toward the target radius R. We introduce a new scalar variable r, and replace the radius R in the tangency and collision constraints with r. Intuitively, when the radius of the bar is small, the tangency and collision constraints are easier to satisfy, and the optimization can gradually increase the radius to the target value (fig. 9).

Then, we can formulate the objective of the MILP to maximize the radius

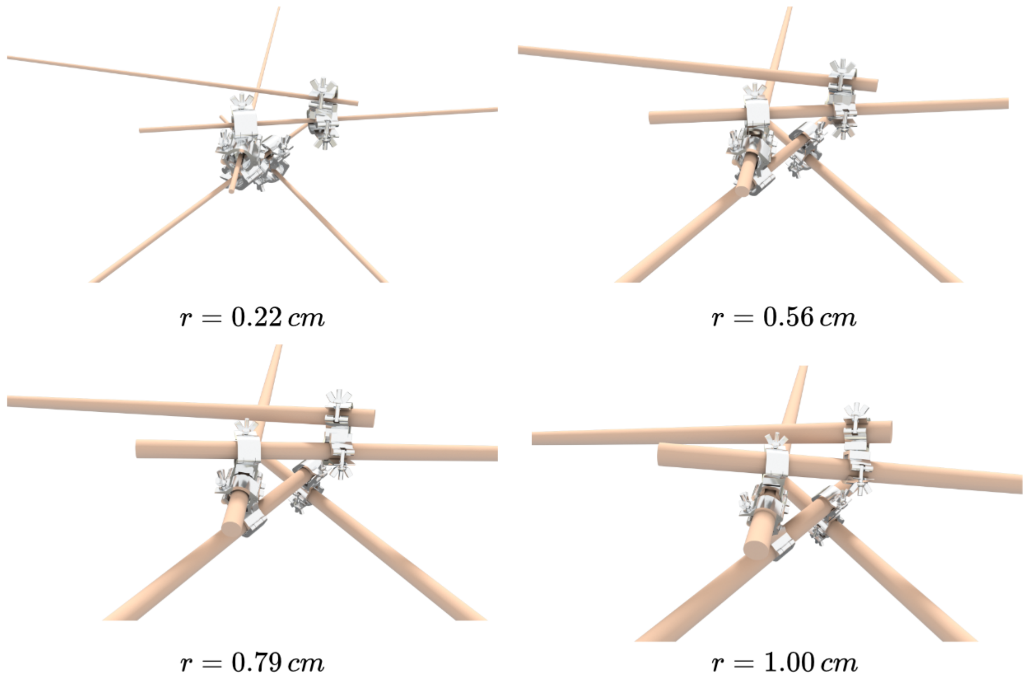

Figure 9: Illustration of inflating radius with a fixed trust region size and contact assignment.

r with a bound on the target radius R:

$$\max_{d\mathbf{x}, d\mathbf{n}, \mathbf{z}, \mathbf{t}, \mathbf{y}, r} \quad r \tag{21}$$

$$\begin{aligned}
s.t. \quad & eq.~(4) - eq.~(5) && \text{Delta variable feasibility} \\
& eq.~(7) - eq.~(10) && \text{Tangency with } r \\
& eq.~(11) - eq.~(12) && \text{Bar collision with } r \\
& eq.~(13) - eq.~(16) && \text{Joint connectivity} \\
& eq.~(18) - eq.~(20) && \text{Max bar length} \\
& r \leq R && \text{Radius constraint} \\
& d\mathbf{x}, d\mathbf{n} \in \mathbb{R}^{6|E|},\ \mathbf{z} \in \{0,1\}^{N_{cp}},\ \mathbf{t} \in [0,1]^{N_{cp}},\ \mathbf{y} \in \mathbb{R}^{2N_{cp}} &&
\end{aligned}$$

where, as in eq. (2), N_{cp} represents the total number of potential number of contact pairs. The complete formulation is provided in Appendix B.

Before describing the solving technique in section 4, we provide an extension to model collision constraints between external connectors like the swivel coupler in section 3.7.

3.7. Extension: Account for clamps

When external connectors are used to connect bars, additional constraints need to be introduced to prevent collisions between the connectors. This will happen when the location of two connectors is too close on the same bar (fig. 10).

Formally, for each pair of clamps (i, j) and (i, k) on the same bar L_i, we constrain their distance to be larger than a threshold d_{cc}:

$$T_{i,j} - T_{i,k} \geq d_{cc} - M(1 - z^v_{i,j}) - M(1 - z^v_{i,k}) - Mu^v_{i,j,k} \tag{22}$$

$$T_{i,k} - T_{i,j} \geq d_{cc} - M(1 - z^v_{i,j}) - M(1 - z^v_{i,k}) - Mu^v_{i,j,k} \tag{23}$$

where $T_{i,j}$ is computed as in eq. (17), and $u^v_{i,j,k}$ is a new sign variable that help the optimization to switch between left and right positions of the clamps, similar to the side switch variables in section 3.3. The number of additional constraints and variable added is linear in the number of bars.

In practice, we find that the tangency constraint involving the bar radius r often compete against the satisfaction of the collision constraints. To make sure that the radius and collision constraints' satisfaction are approached in the same rate, we swap the d_{cc} in eq. (22) with a scalar collision variable c, and add a constraints $c/d_{cc} = r/R$ to the optimization. In fig. 10, we show that these constraints make the optimizer adjust the design globally to avoid the collision, where a local perturbation is not enough.

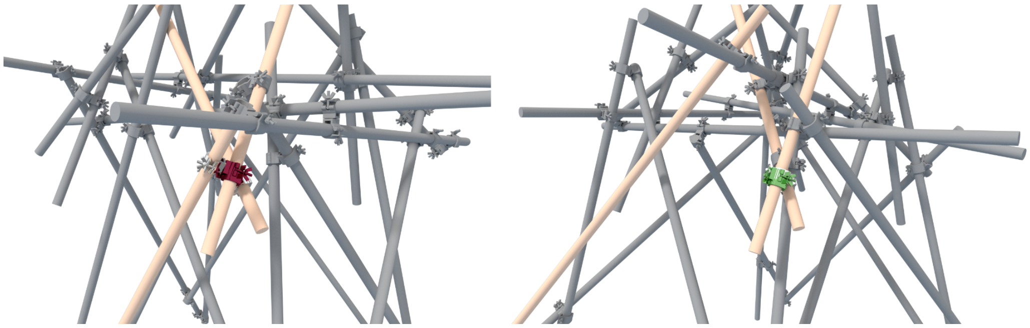

Figure 10: Clamp collision constraint. (1) two clamps on the same bar L_i are in collision because their locations on the bar are two close. (2) a collision-free solution is obtained with the introduction of the clamp collision constraint. Note that the optimizer needs to adjust the design globally to avoid the collision, and a local perturbation is not enough.

4. Solving techniques

In eq. (21), we describe a mixed integer linear programming problem as a sub-iteration in a trust-region-like optimization scheme. In this section, we first describe the overall solving strategy in section 4.1, and then describe the post-processing step to assign the bar length according to the available length set A in section 4.2, which we also formulate as a separate MILP problem.

4.1. Core algorithm: Sequential MILP

The algorithm starts with an initial trust region size Δ_k, and iteratively solves the MILP problem in eq. (21) with the current trust region size. When the MILP sub-problem fails to converge, Δ_k is enlarged to further explore the design space. Otherwise, Δ_k is halved to increase the accuracy of the linear approximation and refine the solution. The algorithm converges when the trust region size reaches the lower bound l_{tr} or fails when it reaches the upper bound u_{tr}.

Figure 11 shows the convergence behavior of the algorithm on a 5-bar star-shaped example. At the beginning, when the trust region size is large, the first-order approximation used in the MILP does not accurately represent the real contraints.Thus, although the MILP converges and all of its constraints are satisfied, we can see the design still violates the actual tangency and collision constraints, marked by the red regions in Figure 11. However, the algorithm changes design dramatically to explore different parts of the solutions pace. As the trust region gets smaller, the linearization becomes more accurate, and the optimizer gradually fine-tunes the design to arrive at a feasible solution.

4.2. Post-processing: bar length adjustment according to available bar stock

The final step is to trim the infinitely long cylinders $\mathbf{x}, \mathbf{n}$ computed from the last section according to the available bar stock. The goal here is to assign a bar length in the stock to each bar, such that the length is long enough to cover the longest pair of clamps on the same bar, while minimizing extra, unused length. While simply assigning the closest stock bar length that is larger than the furthest pair of clamps provides a feasible solution, we solve the following MILP problem for each bar L_e to minimize the distance between assigned bar's end points to the furthest pair of clamps:

Algorithm 1 Sequential Mixed-Integer Linear Programming

procedure SMILP$(V, E; l_{tr}, u_{tr}, \epsilon)$
    $\Delta_k = init_{tr}$ ▷ Initialize trust region size
    while $l_{tr} \leq \Delta_k \leq u_{tr}$ **do**
        $\mathbf{x}^k, \mathbf{n}^k, \mathbf{z}, r, c, converged = \text{MILP}(V, E, \mathbf{x}^{k-1}, \mathbf{n}^{k-1}, \Delta_k)$ ▷ eq. (21)
        if $converged$ **then**
            if $r >= (1-\epsilon)d_{bt}$ **and** $c >= (1-\epsilon)d_{cc}$ **then**
                $\Delta_k \mathrel{/}= 2$ ▷ a solution found under Δ_k, shrink trust region
            if Exceeds max iterations for the same trust region value **then**
                $\Delta_k * = 2$ ▷ current Δ_k timeout, enlarge trust region
        else
            $\Delta_k * = 2$ ▷ not converged, enlarge trust region
    if $\Delta_k \geq u_{tr}$ **then** ▷ trust region size reaches upper bound
        return Failed
    else
        J = [ ] ▷ solution found, extract joint assignment
        for $z^v_{i,j} \in \mathbf{z}$ **do**
            if $z^v_{i,j} = 1$ **then**
                $J.append(\langle e_i, e_j \rangle)$
        return $\mathbf{x}, \mathbf{v}, J$ ▷ Return bar starting point, bar direction, and joint assignment

$$\min_{t_0^e, t_1^e} (t_0^e - t_{c0}^e)^2 + (t_1 - t_{c1}^e)^2 \tag{24}$$

$$s.t. \sum_{i=1}^{|A|} s_i = 1 \tag{25}$$

$$\sum_{i=1}^{|A|} s_i l_i = t_0^e - t_1^e \tag{26}$$

$$t_0^e \leq t_{c0}^e, t_1^e \geq t_{c1}^e \tag{27}$$

$$s_i \in \{0, 1\}, i = 1, \cdots, |A| \tag{28}$$

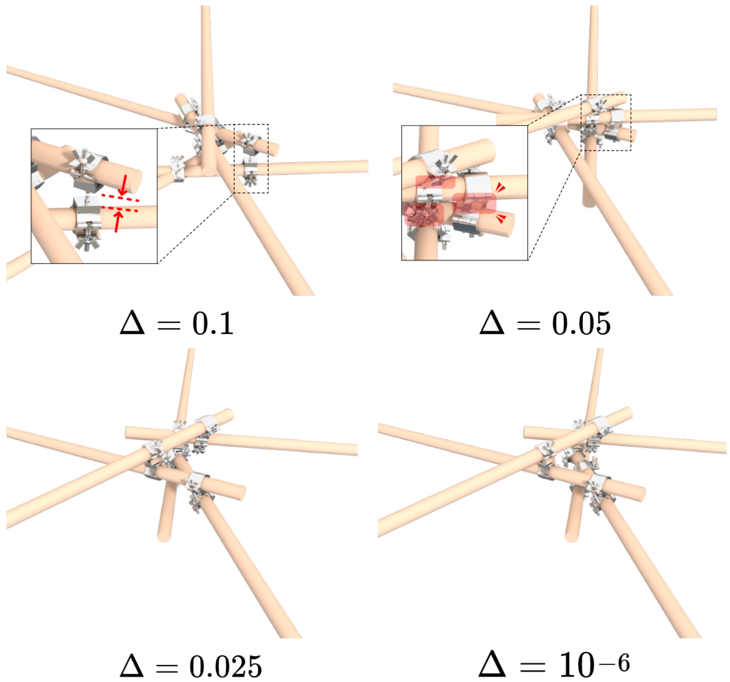

Figure 11: Sequential MILP with shrinking trust region Δ_k. The optimization starts with a large trust region size and explore the design space with certain constraints violated. After a few iterations, the inner MILP problem converges and the trust region gets smaller, and the optimizer fine-tunes the design to a feasible solution.

where t^e_{c0}, t^e_{c1} are the arc-length parameters of the furthest pair of clamps on the bar L_e, which can be computed from $\mathbf{x}, \mathbf{n}$ and joint assignment $\mathbf{z}$ in the previous step. t^e_0, t^e_1 are the arc-length parameters of the assigned bar's end points, and s_i is the binary variable indicating the selected bar length l_i in the stock. Equation (25) ensures that only one bar length is selected, and Equation (26) ensures that the assigned bar length covers the furthest pair of clamps.

5. Results

We implement the proposed algorithm in Python. We use Gurobi 9 [10] to solve the MILP subproblems and the automatic differentiation of JAX [3] to obtain the gradient expressions in eq. (6) and eq. (18). The code is open-source online[1]. All experiments were performed on a consumer-grade laptop without parallelization and GPU acceleration. We demonstrate our algorithm in two simulated and two real-world case studies. In section 5.1, we show that our work can overcome existing method's limitations on enforcing tangency and collisions, and our method can be configured to reproduce traditional reciprocal patterns as well as automatically assigned new contact patterns. In section 5.2, we show that, given the same input line graph, our method can generate structures that respond to different available bar stock. In section 5.3, a simulated benchmark study is presented to evaluate the algorithm's performance on various geometries and topologies. In section 5.4, two real-scale case studies are conducted to demonstrate the physical feasibility of the computed structures and the concept of reusability.

5.1. Comparison with previous work on reciprocal structures

As mentioned in section 1.1, previous work on generating multi-tangent structures has two limitations: (1) the contact patterns are limited to surface-based graph inputs and are assumed to be fixed during the bar position optimization phase, and (2) collisions between bars are not modeled. To demonstrate that our work overcomes these limitations, we compare our work with the geometric optimization formulation for reciprocal structures [32],

[1] https://github.com/KIKI007/Scaffold

which is formulated as follows:

$$
\begin{aligned}
\min_{\mathbf{t},\mathbf{x}}\ & w_1 E_1(\mathbf{x}) + w_2 E_2(\mathbf{t}) + \\
& w_3 \sum_{(L_i,L_j)\in C} \|(\mathbf{x}_i^0 + \mathbf{t}_{i,j}(\mathbf{x}_i^1 - \mathbf{x}_i^0)) - (\mathbf{x}_j^0 + \mathbf{t}_{j,i}(\mathbf{x}_j^1 - \mathbf{x}_j^0)) - 2R\mathbf{n}_{i,j}\|_2^2
\end{aligned}
\tag{29}
$$

$$
\text{s.t.} \quad (\mathbf{x}_k^1 - \mathbf{x}_k^0)\cdot \mathbf{n}_{i,j} = 0,\ k \in \{i,j\}, \forall (L_i, L_j) \in C
$$

where the first objective term E_1 minimizes deviation of $\mathbf{x}$ from the node positions in the input line graph and the second term E_2 uses a quadratic soft barrier to bound t within $[0, 1]$. The third objective term minimizes the deviation between the difference vector of the two contact points and the target contact normal. Note that the direction of the target normal $\mathbf{n}_{i,j}$ (forward or reverse) will dictate the top/down positions of the two bars. In [32], this is decided a priori based on the reciprocal contact pattern and remains fixed during the optimization. In our work, we introduce binary size switching variables to give the optimization the flexibility to choose them automatically (see section 3.3).

In our experiment, we choose a 2-by-2 box array as the test example (fig. 12). We set the thickness of the connector D_c to zero to simulate contexts when rope joints [24] or welding [26] are used to connect pairs of tangent bars. We use the contact assignment C computed by our method since the target line graph is a non-manifold one and does not admit a reciprocal pattern. We optimize eq. (29) using the Newton-CG method with analytic gradient and Hessian and a weight of $w = [1, 1, 200]$. The optimization converged successfully in 19 Newton iterations. In contrast to our work where collision constraints among bars are strictly enforced (fig. 12-2), the result from eq. (29) contains multiple collisions (fig. 12-1). To check the accuracy of the bar tangency, we plot the shortest distances between each pair of contact bars in fig. 13 and observe that our method can achieve exact tangency with an ignorable numerical error and the result from eq. (29) has larger deviations from the target contact distance.

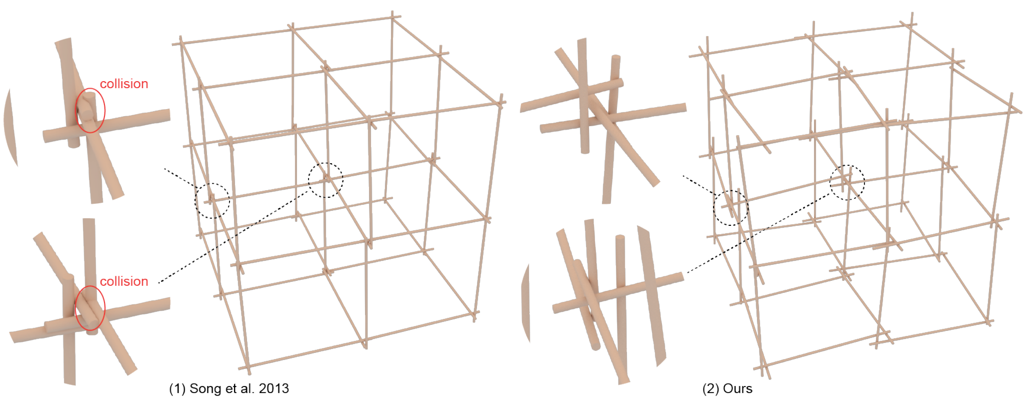

Figure 12: (1) The geometric optimization from previous work [32] can lead to a result with multiple bar collisions. (2) Our method can ensure that the result is collision-free. The two results are using the same contact pattern computed by our method.

Our work can not only automatically generate new contact patterns for shapes that do not admit existing patterns, but can also be configured to use a given pattern. In fig. 14-1, we show our method can reproduce the well-known reciprocal structure by constraining the contact assignment variables $z^v_{i,j}$ to follow the reciprocal pattern, As a comparison, we can relax the pattern constraints and let the algorithm choose the contact assignment freely, and we get a structure using the same set of bars but with a different contact pattern (fig. 14-2).

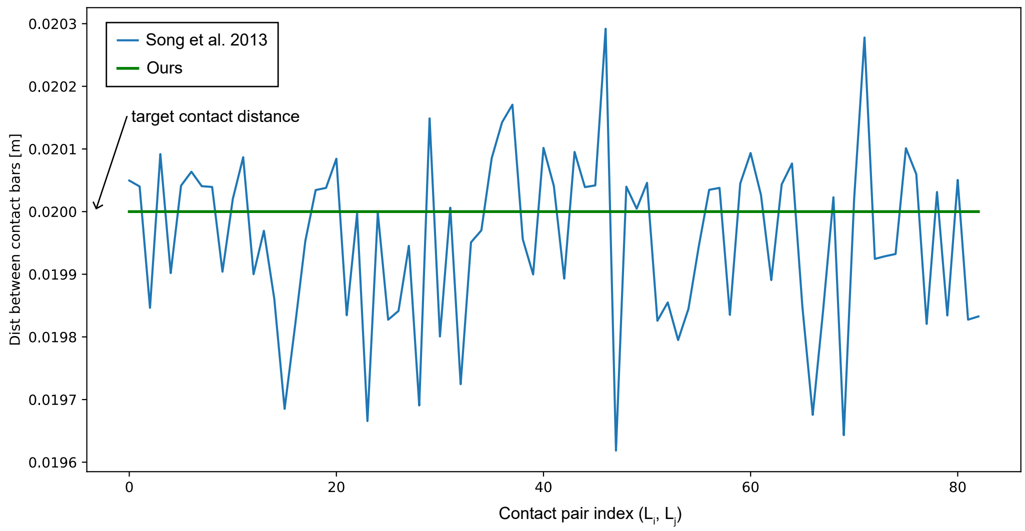

Figure 13: The shortest distance plot between each pair of contact bars' central axes for the results in fig. 12. Two bars are tangent when their shortest distance is 0.02 meter (sum of their radius).

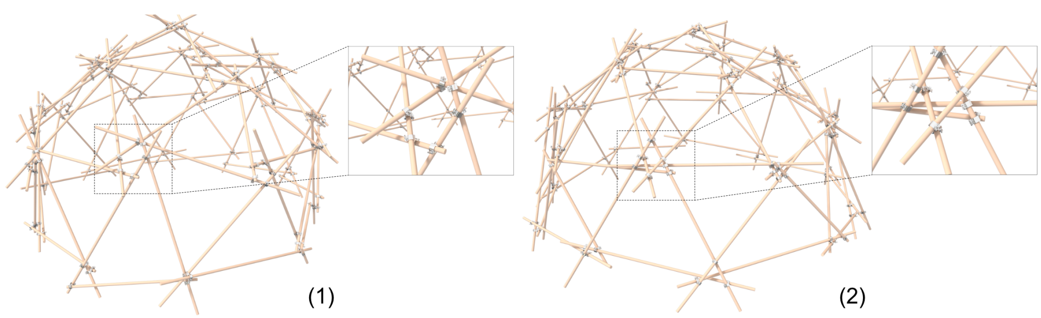

Figure 14: Reciprocal contact pattern and auto-generated contact pattern. Our algorithm can be configured to perform only geometric optimization on the bars to obtain a feasible design following a prescribed contact assignment. (1) a dome structure with a reciprocal contact pattern. (2) the same dome input with an auto-generated contact pattern, using the same set of bars.

5.2. *Design responding to available materials*

An important advantage of our algorithm is its ability to automatically adapt the design according to the available bar stock. In fig. 15, three different multi-tangent configurations are generated for the same input line graph, but with different available bar lengths. The bar-length distribution on the second row shows that the algorithm is capable of regulating the length distribution to fit the given length types. We also observe that with more bar length available, the final structures are more faithful to the input line graph.

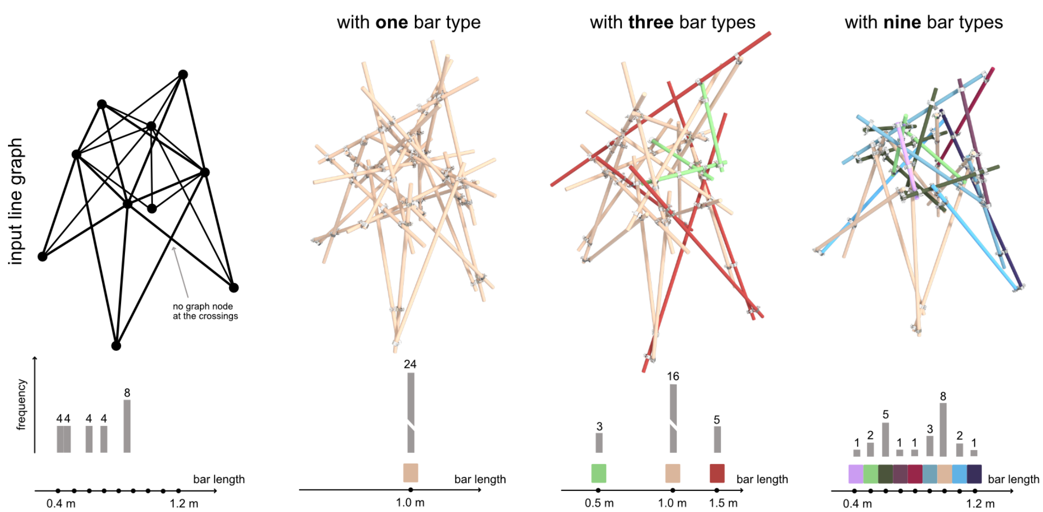

Figure 15: Computed multi-tangent structures responding to different available bar stocks. The algorithm shows the ability to change the design globally to fit different bar length types. Bars are colored according to their length displayed in the bar length distribution.

5.3. *Scalability analysis*

We tested our method on a benchmark of 10 models with varying numbers of elements and topologies, some of which are adapted from [36]. The generated multi-tangent structures are visualized in fig. 16, together with the total runtime. Detailed statistics of the models and the optimization are provided in table 1.

From fig. 16, we can observe that the runtime grows exponentially to the number of elements. However, in some cases, a model might have smaller number of elements but longer runtime (see `bridge` and `roboarch` in table 1). This is because `bridge` has many high-valence nodes and thus the size of the MILP will increase due to the additional variables and constraints introduced by the extra potential contact pairs.

In fig. 17, we show the evolution of the trust region size and the runtime for each MILP subproblem for the `box` 2×2 and `cshape` models. We can observe that the MILP typically runs longer at the beginning of the optimization when the trust region is large and it spends more time to explore until it finds a feasible solution. But as the optimizer gets into a feasible region, its runtime decreases quickly since it is fine tuning an almost feasible solution by shrinking the trust region size for a more accurate linearized model. In the case of `box` 2×2, we can also see that the trust region size plateaus between 7- to 13-th iterations. This is because the MILP solver converges but the resulting radius objective r^* is not close to the target radius R. Thus, it keeps optimizing for a few more iterations until the r^* is close to R and then progresses to a smaller trust region size. In step 14, the trust region size is briefly relaxed due to a failure of MILP convergence and back on track again when the subsequent iteration finds a feasible solution.

While our method can solve moderately sized problem in a reasonable time budget, it shows limitation when the model has too many elements and a large node valence distribution. In `box` 4×4 (last row of table 1), we add one more layer of boxes in x,y,z direction to `box` 3×3. The optimizer only succeeded in solving the first MILP iteration with the initial trust region value, but unable to converge until timeout for the next four iterations.

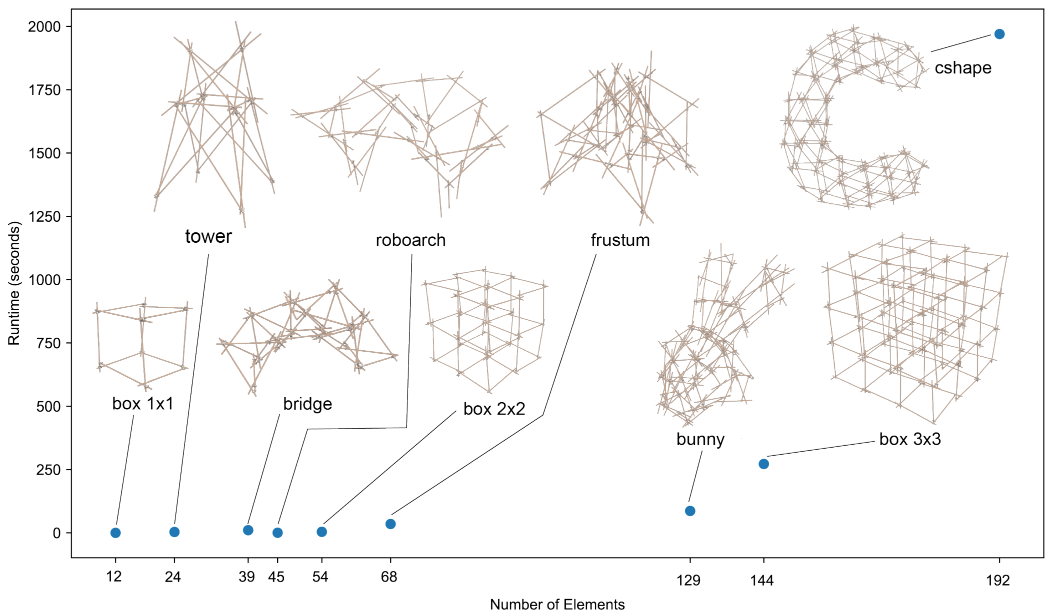

Figure 16: The runtime of our method on the benchmark models.

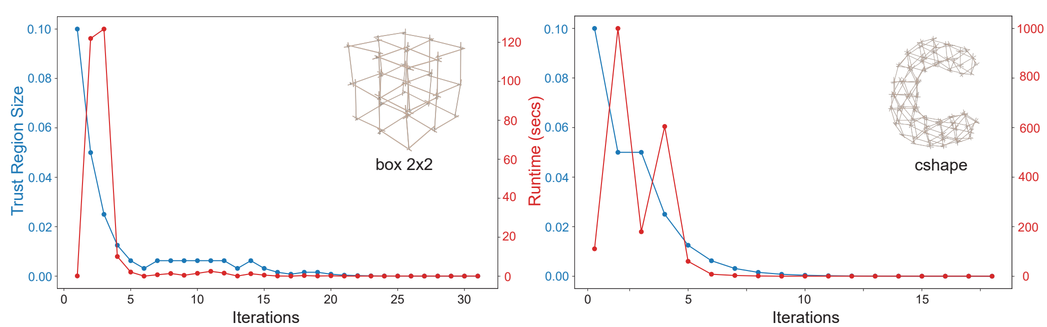

Figure 17: The evolution of the trust region size Δ_k and the MILP runtime over the course of the trust region iterations in the optimization process.

Table 1: Detailed model and optimization statistics of the benchmark. The second to fifth columns indicate: number of elements; the average/standard deviation of node valence; number of continuous, discrete variables and constraints in the MILP; number of trust region (t.r.) iterations. The starting t.r. size $init_{tr}$, the t.r. lower bound l_{tr}, upper bound u_{tr}, and convergence tolerance ϵ are set to be $10^{-1}, 10^{-6}, 1.0, 10^{-2}$ respectively across all experiments. We set a timeout of 1000 seconds for the each MILP iteration.

Model	#elems	Valence avg/std	#cont. vars #disc. vars #constraints	T.r. iters	Runtime (s)
`box` 1×1	12	3.00/0.00	170, 72, 668	25	0.30
`tower`	24	4.8/1.8	578, 654, 4449	17	3.51
`bridge`	39	5.2/1.6	964, 1108, 7746	23	10.65
`roboarch`	45	2.5/0.9	596, 242, 2272	17	0.53
`box` 2×2	54	4.0/0.8	1007, 656, 5557	25	44.14
`frustum`	68	4.3/0.8	1335, 1006, 7901	18	35.06
`bunny`	129	3.9/1.6	2607, 2212, 16511	17	86.57
`box` 3×3	144	4.5/0.9	2971, 2177, 18267	31	272.09
`cshape`	192	5.2/1.5	4670, 4812, 35335	18	1969.90
`box` 4×4	300	4.8/0.8	6440, 4930, 40660	5	4001.67 (timeout)

5.4. Real-world case studies

To demonstrate the design flexibility and the reuse concept of our approach, we present two real-scale built case studies. In this section, we describe two design strategies for the input line graph (section 5.4.1), fast assembly method using Augmented Reality (AR) (section 5.4.2), the analysis of the assembly results (section 5.4.4), and the potential to reconfigure the structure (section 5.4.3). This section is adapted from our previous conference publication [15].

5.4.1. Input line graph design strategies

The first design strategy uses a "bottom-up" approach, where the line graph is procedurally aggregated from the same type of modules (fig. 18-1). Each module contains two juxtaposed triangular prisms, which is kinematically stable. By aggregating these modules, we can create three arches of different heights, and by connecting these arches with a top chord and a central column, we obtain a doubly curved line graph structure (fig. 18-2-3).

While the results in section 5.1 and section 5.2 are computed by directly inputting the line graphs into the algorithm, this design consists of 430 bars and the optimization is too complex to be solved in one shot. Thus, we use a procedural computation strategy by dividing the line graph in groups and solve for each subgroup incrementally. We use the decomposition shown in fig. 20-1, which is also used for the modularized prefabrication discussed in the next section. When computing for a new subgroup, the previously computed groups are fixed. Using this strategy, a feasible multi-tangent structure can be generated that uses only one-meter-long bars (fig. 18-4). This particular design instance, named `Archolumn`, consists of 430 bars and 750 couplers, with a dimension of 5.45 m x 6.35 m x 3.4 m (width x length x height).

The second design strategy uses a "top-down" approach, where the line graph is first designed as a dome structure, and then it is subdivided into a line graph. 3D graphic statics [19, 21] is used to generate a compression-only

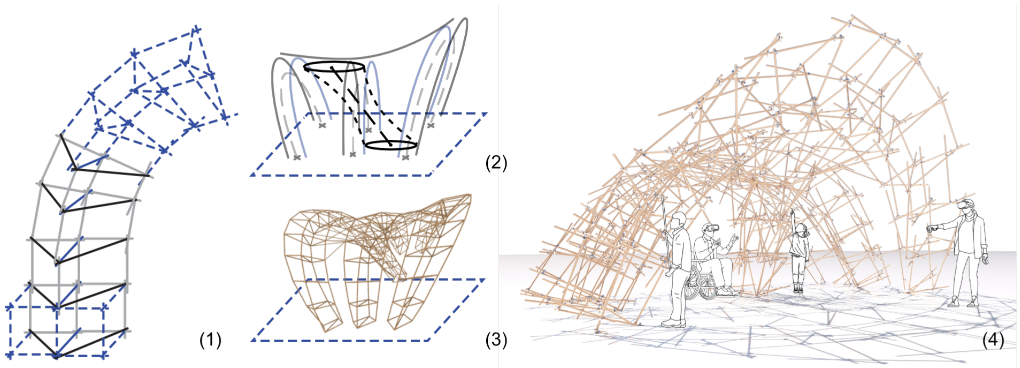

Figure 18: "Bottom-up" design of the line graph. (1) structurally stable modules can be aggregated to form an arch. (2) overall design sketch: three arches and a central column to form a doubly-curved structure. (3) following the design sketch, final line graph is created by connecting three arches and a central column, all aggregated from modules that share the same topology, but morphed geometrically. (4) the optimization algorithms takes the line graph and generates the design `Archolumn` that only uses one-meter-long bars and swivel couplers.

grid shell structure, starting from a volumetric boundary fig. 19-(1) and then converted into a dome-shaped skeleton fig. 19-(2-3). The Grasshopper Plugin "3D Graphic Statics" [8] is used in this form finding process, which takes the bounding volume, the location of the support, and the load conditions as input and generates a funicular structure accordingly. To avoid using overly long bars, the dome is further subdivided into tetrahedrons while maintaining structural stability (fig. 19-(4)). Tetrahedrons also help stabilize the structure during the assembly process. Using the same procedural computation strategy discussed above with the decomposition in fig. 20-2, we get a final design instance, named `Bloomdome`, which uses 210 one-meter-long bars and 445 couplers, with a dimension of 6.5 m x 5.6 m x 2.8 m (fig. 19-(5)).

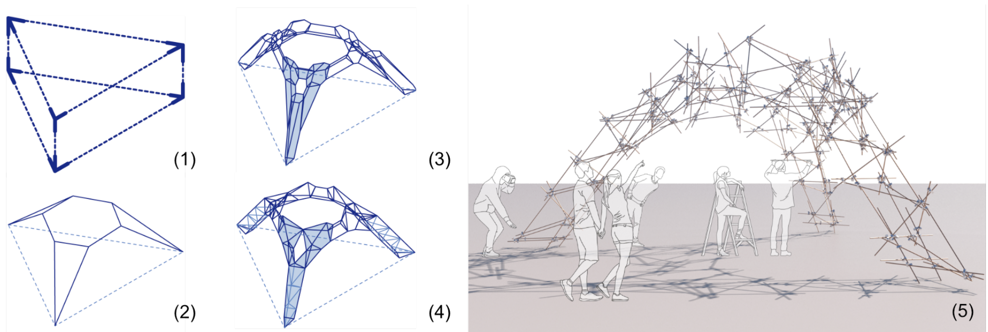

Figure 19: Top-down design of the line graph. (1) volumetric boundary of the dome. (2-3) dome-like skeleton generated by 3D graphic statics. (4) subdivided dome into tetrahedrons to avoid overly long bars. (5) the final, optimized multi-tangent design, called `Bloomdome`, which uses only one-meter-long bars and swivel couplers.

5.4.2. Augmented Reality-assisted assembly strategies

Building multi-tangent structures with the traditional, manual assembly method comes with formidable complexity, as each bar has its unique spatial position and connectivity with other bars. To simplify the assembly process, we propose two strategies: (1) prefabricating modules and (2) using Augmented Reality (AR) to guide the assembly process. We first decompose the structure into modules, grouped with colors in fig. 20, based on an engineering judgment that considers structural stability and ease of assembly. We use Microsoft Hololens 2 [11] and the Fologram app [6] for AR visualization.

AR is first used in the prefabrication stage, where workers can see projected spatial positions of the bars and couplers (fig. 21-1-2). Extra anchors and bars are used to temporarily reinforce the modules during assembly (white bars in fig. 21-1), preventing the modules from collapsing or shifting. Two completed modules are shown in fig. 21-3.

The prefabricated modules are then transported on site for combination,

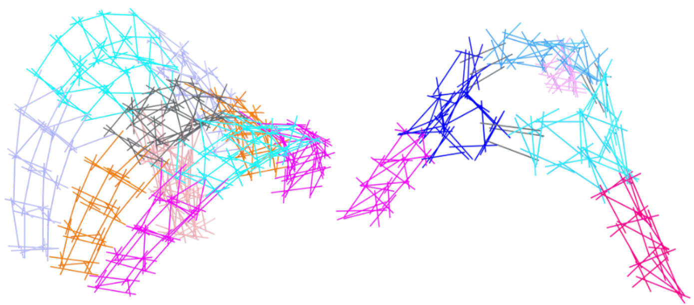

Figure 20: The structures are decomposed into modules manually for prefabrication and onsite assembly. Modules are colored differently for illustration.

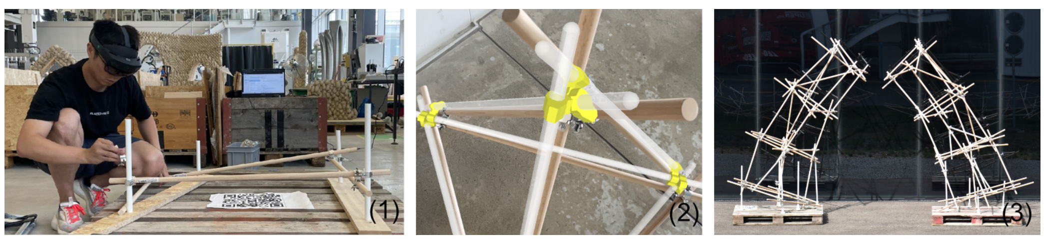

Figure 21: AR-assisted prefabrication of modules. (1-2) Workers are equipped with Hololens to see the spatial positions of the bars and couplers. The white bars are used for temporary reinforcement and support. (3) Two completed modules.

where AR is again used to provide guidance on the spatial location and connectivity among the modules (fig. 22-1). After completion, the extra bars used for reinforcement are removed (fig. 22-2).

5.4.3. Potential for reconfiguration

As an extension of our system, the multi-tangent structures can change its function and appearance by re-distributing the bars. In fig. 23, we show a reconfiguration of the `Archolumn` structure, where the central column is designed to be removable to create more open space inside the structure (fig. 23). To compensate for the support provided by the column, 65 bars

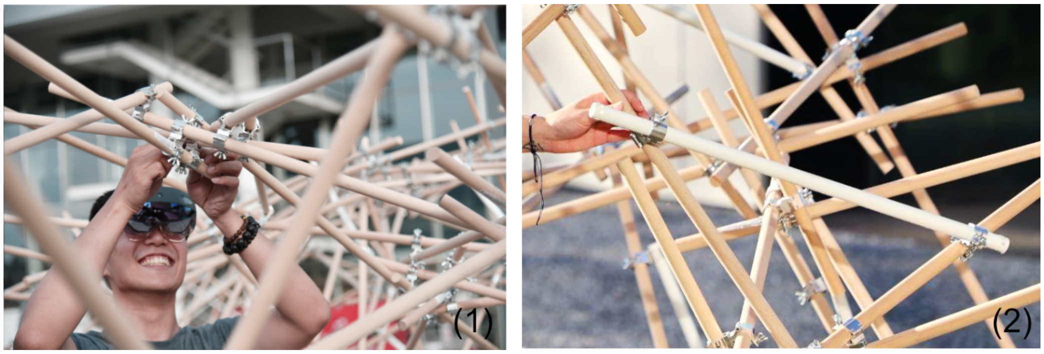

Figure 22: AR-assisted onsite assembly. (1) modules are transported on-site and assembled with AR guidance for accurate placement and connectivity. (2) after completion, the extra bars used for reinforcement (white bars) are removed.

of the column are disassembled and relocated to the top chord to reinforce the arch (blue bars in fig. 23-2), with the remaining 49 bars recycled into the stock. This experiment demonstrates the potential of our system to adapt to different requirements by reconfiguring the structure with minimal material waste and rework.

5.4.4. Analysis of the assembly results

Detailed statistics on the equipment, materials, and assembly time of the two case studies are summarized in table 2. Our data show that both pavilions can be erected within one day of prefabrication and six hours of assembly on site. Both pavilions are built using the same set of bars and couplers. The first pavilion `Archolumn` is assembled first and then disassembled to build the second pavilion `Bloomdome`. This shows that freefrom temporary structures can be built quickly and without waste using our method.

Both structures achieve expected structural stability. As a qualitative analysis of the final assembly results, we use a point-cloud scan (fig. 24-1)

Table 2: Detailed statistics of the assembly results, including labor, requipment, materials, and assembly time.

	`Archolumn` (without reconfiguration)	`Archolumn` (reconfigured)	`Bloomdome`
Workers Equipment	2 person 2 Hololens	2 person 2 Hololens	1 person 1 Hololens
Bars	430	381	210
Couplers	750	670	445
Dimension (WxLxH)	5.5m 6.2m 3.4m	same as left	6.5m 5.6m 2.8m
Module assembly time	10 modules 24 hrs	N/A	7 modules 18 hrs
On-site assembly time	6 hrs	5 hrs	3 hrs

Figure 23: `Archolumn` is designed for two scenarios: with (1) and without the central column (3). Transformation is achieved by removing the central column (pink) and redistributing the bars to reinforce the arch (blue). While the structure before configuration shows a more enclosed, intimate space (1), the reconfigured structure provides a more open, spacious interior space that invites more daylight to come in (4).

and an AR-overlay (fig. 24-2)to compare the assembled structure with the digital model. In (fig. 24-1), we observe that the longest overhang (2.6 m) of `Archolumn` has a deviation of 32 cm. This can be attributed to the flexibility of the bars and the imprecision of the assembly. The deviation is within the acceptable range for temporary structures (within 9% of the total height) according to the design judgment, and it does not affect the overall stability of the structure. We leave the detailed analysis and improvement to reduce such deformation for future work.

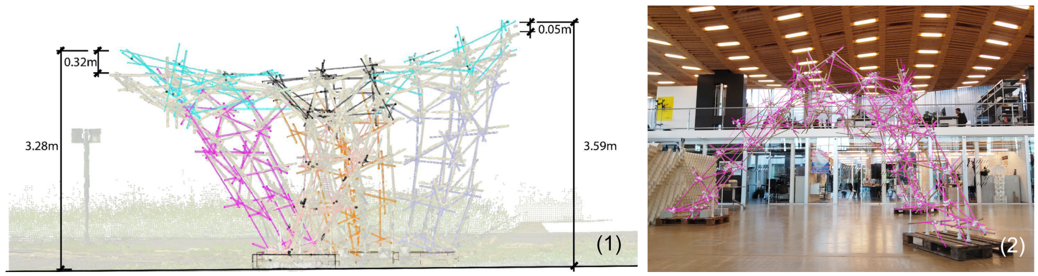

Figure 24: Qualitative analysis of the assembly results. (1) A point cloud scan of the assembled `Archolumn` structure shows a maximum deflection of 32 cm at the longest overhang (2.6 m), which is 10% of the total height. (2) AR overlay of the digital model (pink) on the assembled `Bloomdome` structure.

6. Discussions

6.1. Discussions of the results

The proposed design optimization formulation and algorithm are demonstrated to automatically resolve contact assignment and bar configuration to find a feasible multi-tangent configuration for free-form line graph input. Various results have shown that by combining infinitely long bar formulation, linearization, and various constraint modeling techniques, our algorithm can turn the originally mixed-integer, bilevel optimization problem into a sequence of mixed-integer linear programs and solve them successfully to obtain a feasible design solution.

In contrast to prior work that constraints input line graph's topology, our approach opens up the design space of multi-tangent structures by offering the flexibility to find new contact patterns for any input line graph. While previous work fails to achieve exact tangency and model collisions, our method can enforce both as hard constraints, ensuring the physical feasibility of the resulting structure. We also show results that the proposed algorithm can adapt the design to different available bar stocks, where we observe that a stock with more diverse bar lengths can lead to a design that represents the original design more faithfully.

Finally, we present two full-scale, real-world case studies to demonstrate the efficiency of assembly and the concept of reusability. AR-assisted assembly strategies are deployed for both off-site assembly of modules and on-site installation of them. The computational design approach and the assembly strategy allow us to rapidly design and build free-form structures using a given kit of parts, and disassemble one to realize another design with completely different structural typology.

6.2. Limitations and future work

For computational design of multi-tangent structures, we see a number of limitations and opportunities for future work.

Integration of structural and functional constraints. The current design algorithm only considers the geometric and connectivity aspects of the structure and does not consider functional objectives of the structure, e.g., as a load-bearing structure or a temporary shelter. Future work could integrate structural equilibrium or elastic stiffness constraint into the optimization formulation, or integrate the multi-tangent constraints into a topology optimization framework. New physics simulation techniques will also be needed to accurately capture the behaviors of multi-tangent structures under self-weight or external loads, which includes global kinematic response due to local joint mechanism, contact between bars, and elastic deformation. Additional geometric constraints could be incorporated to facilitate the installation of cladding for sheltering or spatial separation purposes.

Computational scalability. For a large design case, such as the `box` 4×4 shown in section 5.3 and two case studies shown in section 5.4, our current algorithm struggles to find a feasible solution within a reasonable time frame. This is likely due to the discrete contact assignment variables growing exponentially with the number of edge counts in the input line graph, and the branch-and-bound algorithm of the MILP solver can not prune the search space effectively and easily get stuck in local minima. Although we have shown that the manual decomposition strategy offers a practical way to decompose the problem into modules and solve each one in a reasonable amount of time, future work could investigate ways to automate the decomposition process, e.g. [13, 36].

Multi-solution and user-control. While our proposed algorithm can find one feasible solution for a given line graph, there often exist many feasible solutions that could present different aesthetics or structural performance. Currently, our algorithm does not offer users any direct control over the final design, which, although automated, may appear to be too inflexible for designers who want to have more granular control over the design. Future work thus could explore a more interactive, procedural design process where the user can guide the optimization process by providing feedback or constraints, or a design optimization algorithm that can generate a set of diverse feasible solutions.

Material degradation and active bending. Although this work primarily utilizes industrially produced linear wooden bars, future studies should investigate the effects of material degradation due to repeated reuse cycles, including wear, environmental exposure, and load-induced fatigue. Understanding how these factors impact structural performance and longevity is crucial for substantiating claims of extended reusability. Future research should conduct experimental testing or historical data studies to document material longevity in practical reuse scenarios. Further exploration of materials that leverage inherent elasticity—such as natural bamboo or fiber-reinforced composites—could also enable innovative active bending structures. These alternative materials, however, will similarly require rigorous evaluation of degradation and fatigue mechanisms over multiple reuse cycles.

7. Conclusion

This work has presented a new way to design and build freeform bar structures with limited material resources that can be reused multiple times. We have proposed a computational framework to open up the design space of an existing construction system, called the multi-tangent bar structures, to accommodate freeform design intentions. Our core contribution is a new way to formulate this design problem, so that a naive, intractable optimization formulation can be transformed into a sequence of MILP problems that can be solved effectively by combining off-the-shelf MILP tools and a trust-region-like optimization outer loop. We show that we can model several practical considerations, such as tangency, collisions, joint connectivity, etc., as linear constraints in the MILP formulation. Our simulated result demonstrate that the design algorithm can simultaneously offset bars and assign joints, and generate multi-tangent structures out of complex graph input, without constraining the graph to have certain fixed topologies as in previous work. To validate our design algorithm and the concept of reusability, we physically built a generated design, disassembled it, and reused the kit to build another generated design.

The tools presented in this paper provide temporary structures designers with an automated framework to assist in the design of freeform structures using a given kit of parts. The mathematical formulation presented could inspire future research on the use of mathematical optimization to model part connectivity, collision, and resource availability of other structural systems. Specific to multi-tangent structures, this paper invites future research on integrating structural consideration into the form-finding process, algorithmic scalability, and automatic structural decomposition for assembly.

In a world that is facing resource scarcity and environmental challenges, designing more with less material becomes imperative for the designers of the built environment. We believe that the methods presented in this paper represents a fresh approach towards this vision.

Funding

YH was supported by the ETH postdoc fellowship and the SNSF Ambizione Grant (Grant No. 223384). ZW was supported by the Swiss National

Science Foundation through the National Centre of Competence in Digital Fabrication (Grant No. 200021 200644).

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

YH thanks Caelan Garrett for his joint initial explorations of multi-tangent design back in 2021. The authors thank Victor Pok Yin Leung for suggesting the usage of swivel couplers for the case studies. The authors also thank colleagues from the ETH Zurich MAS DFAB program: Carl P-Conquilla, Keng Chia Chang, Lihin Weera, Nijat Mahamaliyev, Yo Cheng Lee, Zac Zhuo Zhang, Kai Hsun Yeh, and Georg Bauer, for their dedicated help during the assembly of the two real-world case studies.

References

[1] Alexi, E.V., Kenny, J.C., Atanasova, L., Casas, G., Dörfler, K., Mitterberger, D., 2024. Cooperative augmented assembly (CAA): Augmented reality for on-site cooperative robotic fabrication. Construction Robotics 8, 28. URL: https://doi.org/10.1007/s41693-024-00138-6, doi:10.1007/s41693-024-00138-6.

[2] Baverel, O., Nooshin, H., Kuroiwa, Y., 2004. Configuration processing of nexorades using genetic algorithms. Journal of the International Association for Shell and Spatial Structures 45, 99–108.

[3] Bradbury, J., Frostig, R., Hawkins, P., Johnson, M.J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., Zhang, Q., 2018. JAX: Composable transformations of Python+NumPy programs. URL: http://github.com/google/jax.

[4] Brütting, J., Senatore, G., Fivet, C., 2021. Design and fabrication of a reusable kit of parts for diverse structures. Automation in Construction 125, 103614. URL: https://www.sciencedirect.com/science/article/pii/S0926580521000650, doi:10.1016/j.autcon.2021.103614.

[5] Brütting, J., Senatore, G., Schevenels, M., Fivet, C., 2020. Optimum Design of Frame Structures From a Stock of Reclaimed Elements. Frontiers in Built Environment 6. URL: https://www.frontiersin.org/articles/10.3389/fbuil.2020.00057, doi:10.3389/fbuil.2020.00057.

[6] Fologram, . Fologram: Mixed reality for the workshop. URL: https://fologram.com/.

[7] Garg, A., Sageman-Furnas, A.O., Deng, B., Yue, Y., Grinspun, E., Pauly, M., Wardetzky, M., 2014. Wire mesh design. ACM Transactions on Graphics 33, 1–12. URL: https://dl.acm.org/doi/10.1145/2601097.2601106, doi:10.1145/2601097.2601106.

[8] Graovac, O., 2019. 3D Graphic Statics. URL: https://www.food4rhino.com/en/app/3d-graphic-statics.

[9] Griva, I., Nash, S.G., Sofer, A., 2008. Linear and Nonlinear Optimization 2nd Edition. SIAM.

[10] Gurobi Optimization, LLC, 2024. Gurobi optimizer reference manual. URL: https://www.gurobi.com.

[11] HoloLens, M., . Microsoft HoloLens: Mixed Reality Technology for Business. URL: https://www.microsoft.com/en-us/hololens.

[12] Huang, Y., Alkhayat, L., De Wolf, C., Mueller, C., 2021. Algorithmic circular design with reused structural elements: Method and tool, in: International Fib Symposium - Conceptual Design of Structures 2021, pp. 457–468. URL: https://www.fib-international.org/publications/fib-proceedings/i-fib-i-conceptual-design-of-structures-solothurn-2021-proceedings-em-pdf-em-html, doi:10.35789/fib.PROC.0055.2021.CDSymp.P056.

[13] Huang, Y., Zhang, J., Hu, X., Song, G., Liu, Z., Yu, L., Liu, L., 2016. FrameFab: Robotic Fabrication of Frame Shapes. ACM Trans. Graph. 35, 224:1–224:11. URL: http://doi.acm.org/10.1145/2980179.2982401, doi:10/gctq9n.

[14] Jacobson, A., 2019. RodSteward: A Design-to-Assembly System for Fabrication using 3D-Printed Joints and Precision-Cut Rods. Computer Graphics Forum 38, 765–774. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13878, doi:10.1111/cgf.13878.

[15] Jiang, C., Hung, Y.H., Wang, Z., Huang, Y., Gheyselinck, A.L., Aejmelaeus-Lindstrom, P., 2024. Computational Design and AR-assisted Assembly of Infinitely Reusable Temporary Structures, in: Proceedings of IASS Annual Symposia, Zurich.

[16] Kladeftira, M., Leschok, M., Skevaki, E., Tanadini, D., Ohlbrock, P.O., D'Acunto, P., Dillenburger, B., 2022. A Study on Bamboo, 3D Printed Joints, and Digitally Fabricated Building Components for Ultralight Architectures, in: Proceedings of the 41st Annual Conference of the ACADIA 2022, Philadelphia.

[17] Larsen, O.P., 2007. Reciprocal frame architecture. Routledge.

[18] Larsen, O.P., Larsen, J., 2022. Reciprocal frame structures. Structure Magazine , 20–23.

[19] Lee, J., 2018. Computational Design Framework for 3D Graphic Statics. Ph.D. thesis. ETH Zurich, Department of Architecture. Zurich. doi:10.3929/ethz-b-000331210.

[20] Lira, W., Fu, C.W., Zhang, H., 2018. Fabricable eulerian wires for 3D shape abstraction. ACM Transactions on Graphics 37, 1–13. URL: https://dl.acm.org/doi/10.1145/3272127.3275049, doi:10.1145/3272127.3275049.

[21] Lu, Y., Hablicsek, M., Akbarzadeh, M., 2024. Algebraic 3D Graphic Statics with Edge and Vertex Constraints: A Comprehensive Approach to Extend the Solution Space for Polyhedral Form-Finding. Computer-Aided Design 166, 103620. URL: https:

//linkinghub.elsevier.com/retrieve/pii/S0010448523001525, doi:10.1016/j.cad.2023.103620.

[22] Ma, Z., Walzer, A., Schumacher, C., Rust, R., Gramazio, F., Kohler, M., Bächer, M., 2020. Designing Robotically-Constructed Metal Frame Structures. Computer Graphics Forum 39, 411–422. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13940, doi:10.1111/cgf.13940.

[23] Miguel, E., Lepoutre, M., Bickel, B., 2016. Computational design of stable planar-rod structures. ACM Transactions on Graphics 35, 86:1–86:11. URL: https://doi.org/10.1145/2897824.2925978, doi:10.1145/2897824.2925978.

[24] Mitterberger, D., Atanasova, L., Dörfler, K., Gramazio, F., Kohler, M., 2022. Tie a knot: Human–robot cooperative workflow for assembling wooden structures using rope joints. Construction Robotics 6, 277–292. URL: https://link.springer.com/10.1007/s41693-022-00083-2, doi:10.1007/s41693-022-00083-2.

[25] Neveu, W., Puhachov, I., Thomaszewski, B., Bessmeltsev, M., 2022. Stability-Aware Simplification of Curve Networks, in: Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings, ACM, Vancouver BC Canada. pp. 1–9. URL: https://dl.acm.org/doi/10.1145/3528233.3530711, doi:10.1145/3528233.3530711.

[26] Parascho, S., Kohlhammer, T., Coros, S., Gramazio, F., Kohler, M.,

2018. Computational Design of Robotically Assembled Spatial Structures: A sequence based method for the generation and evaluation of structures fabricated with cooperating robots, in: AAG 2018: Advances in Architectural Geometry 2018, Klein Publishing. pp. 112–139. URL: https://www.research-collection.ethz.ch/handle/20.500.11850/298876.

[27] Parigi, D., Kirkegaard, P.H., Sassone, M., 2012. Hybrid optimization in the design of reciprocal structures, in: Proceedings of the IASS Symposium 2012: from spatial structures to spaces structures.

[28] Parigi, D., Pugnale, A., 2014. Three-Dimensionality in Reciprocal Structures: Concepts and Generative Rules. Nexus Network Journal 16, 151–177. URL: https://doi.org/10.1007/s00004-014-0183-y, doi:10.1007/s00004-014-0183-y.

[29] Pugnale, A., Parigi, D., Kirkegaard, P.H., Sassone, M.S., 2011. The principle of structural reciprocity: history, properties and design issues, in: IASS: Intl. Conference on Space Structures 2011.

[30] Ren, Y., Panetta, J., Chen, T., Isvoranu, F., Poincloux, S., Brandt, C., Martin, A., Pauly, M., 2021. 3D weaving with curved ribbons. ACM Trans. Graph. 40, 127:1–127:15. URL: https://doi.org/10.1145/3450626.3459788, doi:10.1145/3450626.3459788.

[31] Rubin, P., 2021. Ensure a graph is connected in a linear programming problem. Mathematics Stack Ex-

change. URL: https://math.stackexchange.com/q/4204346, arXiv:https://math.stackexchange.com/q/4204346.

[32] Song, P., Fu, C.W., Goswami, P., Zheng, J., Mitra, N.J., Cohen-Or, D., 2013. Reciprocal frame structures made easy. ACM Transactions on Graphics 32, 1. URL: http://dl.acm.org/citation.cfm?doid=2461912.2461915, doi:10/gbdg7g.

[33] Thönnissen, U., Werenfels, N., 2011. Reciprocal frames–teaching experiences. International Journal of Space Structures 26, 369–371.

[34] Validi, H., Buchanan, A., Lykhovyd, E., 2022. Imposing Contiguity Constraints in Political Districting Models. Operations Research 70, 867–892. URL: https://pubsonline.informs.org/doi/10.1287/opre.2021.2141, doi:10.1287/opre.2021.2141.

[35] Vekhter, J., Zhuo, J., Fandino, L.F.G., Huang, Q., Vouga, E., 2019. Weaving geodesic foliations. ACM Trans. Graph. 38, 34:1–34:22. URL: https://doi.org/10.1145/3306346.3323043, doi:10.1145/3306346.3323043.

[36] Wang, Z., Kennel-Maushart, F., Huang, Y., Thomaszewski, B., Coros, S., 2023. A Temporal Coherent Topology Optimization Approach for Assembly Planning of Bespoke Frame Structures. ACM Transactions on Graphics 42, 1–13. URL: https://dl.acm.org/doi/10.1145/3592102, doi:10.1145/3592102.

Appendix A. Numerical treatment for infinite-length bar distance

When the two bars are close parallel, the formula provided in eq. (3) to calculate the distance between two infinite-length bar degenerates. We use the following formulas to first compute a unit vector $\mathbf{n}_{ij}$ that is orthogonal to both L_i, L_j and then compute the distance:

$$\mathbf{n}_{ij} = ((\mathbf{x}_i - \mathbf{x}_j) \times \mathbf{n}_i) \times \mathbf{n}_j \tag{A.1}$$

$$d_{\infty}[L_i, L_j] = \mathbf{n}_{ij} / \|\mathbf{n}_{ij}\| \cdot (\mathbf{x}_i - \mathbf{x}_j) \tag{A.2}$$

When $\mathbf{n}_{ij}$'s norm is close to zero, we randomly sample a vector that is orthogonal to $\mathbf{n}_i$ to replace $\mathbf{n}_{ij}$ in eq. (A.1). The first-order Taylor approximation $\hat{d_{\infty}}$ of the distance function in eq. (A.2) is:

$$\begin{aligned} &\hat{d_{\infty}}[L_i, L_j](d\mathbf{x}_i, d\mathbf{n}_i, d\mathbf{x}_j, d\mathbf{n}_i) := \\ &\mathbf{n}_{ij} / \|\mathbf{n}_{ij}\| \cdot (\mathbf{x}_i - \mathbf{x}_j) + \mathbf{n}_{ij} \cdot (d\mathbf{x}_i - d\mathbf{x}_j) \end{aligned} \tag{A.3}$$

Appendix B. Complete MILP formulation

$$\max_{d\mathbf{x}, d\mathbf{n}, \mathbf{z}, \mathbf{t}, \mathbf{y}, r} \quad r \tag{B.1}$$

$$s.t. \; -\Delta_k \leq d\mathbf{x}_i, d\mathbf{n}_i \leq \Delta_k, \forall e_i \in E \tag{B.2}$$

$$\mathbf{n}_i^k \cdot d\mathbf{n}_i = 0, \forall e_i \in E \tag{B.3}$$

$$\forall e_i \cap e_j = v, e_i, e_j \in E$$

$$\hat{d_\infty}[L_i, L_j] \leq 2r + D_c + M(1 - z_{i,j}^v) \tag{B.4}$$

$$\hat{d_\infty}[L_i, L_j] \geq -(2r + D_c) - M(1 - z_{i,j}^v) \tag{B.5}$$

$$\hat{d_\infty}[L_i, L_j] \geq 2r + D_c - M(1 - s_{i,j}^v) \tag{B.6}$$

$$\hat{d_\infty}[L_i, L_j] \leq -(2r + D_c) + M s_{i,j}^v \tag{B.7}$$

$$\hat{d_\infty}[L_i, L_j] \geq 2r - M z_{i,j}^v - M(1 - s_{i,j}^v) \tag{B.8}$$

$$\hat{d_\infty}[L_i, L_j] \leq -2r + M z_{i,j}^v + M s_{i,j}^v \tag{B.9}$$

$$\forall v \in V$$

$$0 \leq y_{i,j}^v \leq (|V_{J(v)}| - 1) z_{i,j}^v, \forall (i,j) \in E_{J(v)} \tag{B.10}$$

$$\sum_{j \in V_{J(v)}} y_{s,j}^v = |V_{J(v)}| - 1 \tag{B.11}$$

$$\sum_{j:(i,j) \in E_{J(v)}} y_{i,j}^v = \sum_{j:(i,j) \in E_{J(v)}} y_{j,i}^v - 1, \forall i \in V_{J(v)} \setminus \{s\} \tag{B.12}$$

$$y_{i,j}^v = y_{j,i}^v \tag{B.13}$$

$$\forall e_i \cap e_j = v, e_i \cap e_k = v'$$

$$T_{i,j} - \hat{T}_{i,j}(\hat{\mathbf{x}}_i, \hat{\mathbf{n}}_i, \hat{\mathbf{x}}_j, \hat{\mathbf{n}}_j) - \frac{\partial T_{i,j}}{\partial \mathbf{x}_i}^T d\mathbf{x}_i - \frac{\partial T_{i,j}}{\partial \mathbf{n}_i}^T d\mathbf{n}_i - \frac{\partial T_{i,j}}{\partial \mathbf{x}_j}^T d\mathbf{x}_j - \frac{\partial T_{i,j}}{\partial \mathbf{n}_j}^T d\mathbf{n}_j = 0 \tag{B.14}$$

$$T_{i,j} - T_{i,k} \leq l_{max} + M(1 - z_{i,j}) + M(1 - z_{i,k}) \tag{B.15}$$

$$T_{i,k} - T_{i,j} \leq l_{max} + M(1 - z_{i,j}) + M(1 - z_{i,k}) \tag{B.16}$$

$$r \leq R$$

$$d\mathbf{x}, d\mathbf{n} \in \mathbb{R}^{6|E|},\ \mathbf{z} \in \{0,1\}^{N_{cp}},\ \mathbf{t} \in [0,1]^{N_{cp}},\ \mathbf{y} \in \mathbb{R}^{2N_{cp}}$$