Pacific Graphics 2019
C. Theobalt, J. Lee, and G. Wetzstein
(Guest Editors)

Volume 38 (2019), Number 7

RodSteward: A Design-to-Assembly System for Fabrication using 3D-Printed Joints and Precision-Cut Rods

Alec Jacobson, University of Toronto

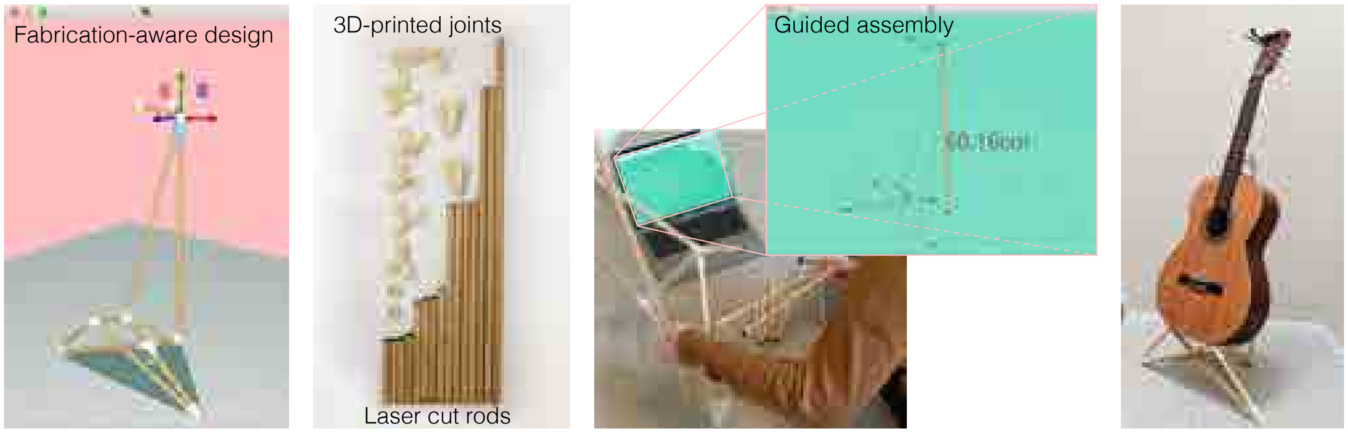


Figure 1: RSDesigner, *our fabrication-aware interface for joint-rod structures, helps a user design a custom-fit guitar stand with a unique aesthetic discovered during modeling. Joint geometries are 3D-printed and wooden dowels are laser cut to length. The user physically assembles the structure guided by an on-screen aid,* RSAssembler.

Abstract
We present RodSteward*, a design-to-assembly system for creating furniture-scale structures composed of 3D-printed joints and precision-cut rods. The RodSteward systems consists of:* RSDesigner*, a fabrication-aware design interface that visualizes accurate geometries during edits and identifies infeasible designs; physical fabrication of parts automatically generated 3D-printable joint geometries and cutting plans for rods; and* RSAssembler*, a guided-assembly interface that prompts the user to place parts in order while showing a focus+context visualization of the assembly in progress. We demonstrate the effectiveness of our tools with a number of example constructions of varying complexity, style and parameter choices.*

1. Introduction

Advanced manufacturing processes dramatically increase the complexity of physically fabricable geometries. For example, a 3D printer can directly fabricate an intricate, high genus shape, so long as it fits in the machine's build volume. In contrast, standard laser cutters have a much larger albeit two-dimensional cutting bed. Unfortunately, these complementary strengths are not easily leveraged harmoniously in a single design. In particular, large three-dimensional objects are not well suited for either process in isolation. Further complicating design, construction of fabricable parts is a non-trivial task, often requiring slow iterations between virtual design and physical prototyping. For example, a design that *looks* feasible, may turn out to have overlapping or corrupted geometry.

In response, we present the *RodSteward* system, a design-to-assembly system for creating furniture-scale structures composed of 3D-printed joints and precision-cut rods (see Fig. 1). This design space is especially interesting because nearly all geometric complexity is shifted onto the small joint shapes, harmonizing with the qualities of the 3D printer. Meanwhile, the long rods can be purchased *en masse* at any hardware store and diced up with any tool capable of simple-but-precise perpendicular cuts (e.g., a laser cutter, but also a handsaw and miter box). Sparse, wireframe designs are also a currently trendy modern furniture aesthetic.

The RodSteward system has three stages: 1) *RSDesigner*, a fabrication-aware design interface; 2) part geometry realization and physical printing and cutting; and 3) *RSAssembler*, a guided-assembly interface. RSDesigner allows the user to edit a virtual structure while interactively maintaining an accurate visualization of the fabricated parts. Our emphasis on real-time feedback allows a user to fine-tune and evaluate designs on-the-fly. The interface will highlight and alert the user to potential problems with the design such as overlapping parts or structurally unstable designs. Complementing this interface, we propose a novel joint geometry construction algorithm, which generates solid, watertight and 3D-printable

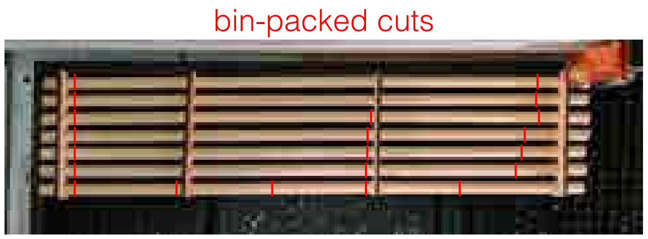


Figure 2: *We pack all of the rod-lengths into a single cut plan over a small number of raw rods. Rods are positioned in place using a "comb" jig with holes cut at regular intervals matching the spacing of the automatically generated cut plan.*

joints given the user's rod-joint network description. A subset of existing methods require manual intervention to generate the literal geometry of each joint, breaking the exploratory design loop. In contrast, our method belongs to the class of automatic methods with a tight design loop that allows the user to focus on the high-level creative aspects of the overall design, rather than the geometry of each joint itself. With varying physical accuracy, some previous methods have simulated the structural stability of rod-joint structures. Our contributions complement this particular well-explored feature, and we therefore leave incorporating this aspect as an incremental improvement to RodSteward. Instead, we focus on first-order design issues such as rod-intersections and balance.

Upon design completion, we automatically engrave each joint with a visible I.D. and send the parts to the 3D-printer. For rods, we generate a cutting plan that packs the segments into a minimal number of standard-size rods, so all can be cut in a single, quick job (see Fig. 2). After fabricating the individual parts, RSAssembler visualizes the partial structure as the user places each part. The user presses a hotkey to advance and the guide suggests the next part to place and updates the visualization. Without the guided assembly interface, assembling these complex structures would reduce to solving a frustratingly difficult 3D puzzle.

We demonstrate the effectiveness of RodSteward as a design-to-assembly system, by constructing structures (e.g., Fig. 3) that highlight the simplicity and generality of our system to accommodate non-manifold edge-networks, circular and polygonal rod profiles, acute angles between adjacent rods, and complex yet non-self-intersecting and balancing structures.

2. Related Work

In the past decades, designers, researchers and hobbyists alike have looked for ways to leverage the geometric complexity afforded by 3D printing with traditional or unconventional parts. We focus the discussion of previous works on those similar in terms of interface aspirations or methodologies. The main differences with our work are: our end-to-end, design-to-assembly system; the fabrication-aware tight interactive design loop, and interactive guided assembly plan. To our knowledge, no such complete system exists.

2.1. Design and assembly

The human-computer interaction and graphics communities have embraced computational fabrication and its evolution beyond classic computed-aided design and manufacturing (see, e.g., [MBM*15, UBM15, BKLZ17, LEM*17]). We join this field of research in rejecting the idea that the existence of mass-production should preclude an individual's opportunity to participate in the unique design and customization of everyday objects (e.g., see Fig. 1).

We are especially interested in hybrid or heterogeneous systems that combine 3D-printing with other materials to create larger objects. For example, Kovacs et al. [KSW*17] build room- and architectural-scale objects with 3D-printed joints and recycled PET bottles. While their 3D-printed geometry construction is also automatic, their trusses result from intersecting a 3D shape with a tetrahedral honeycomb, so joints are less general with fixed topology. Kovacs et al. [KIL*18] incorporate actuation to create articulated structures, but the fixed joint configuration remains. In contrast to our design-to-assembly system, the interface contributions of these methods stop at fabrication: the user is left to build a complex structure with many labeled parts and no explicit instructions. Unlike this and other tools that only focus on design and fabrication, we consider the end-to-end system from design to assembly.

Leen et al. [LRL17] introduce a tangible, modular magnet-based interface for designing wireframe objects. This work complements ours and could provide input to our RSDesigner tool, although their sensor rods have upper and lower bounds on length and joints can only accommodated a fixed number of incident rods at bounded angles. Meanwhile, Agrawal et al. [AUK*15] physically sketch very general, yet temporary 3D wireframe structures made of tape.

Mueller et al. [MIG*14] break away from layer-by-layer 3D printing to fabricate wireframe structures by extruding plastic in 3D. Wu et al. [WPGM16] and Huang et al. [HZH*16] extend this idea to a larger class of wireframe surfaces using a 5DOF printer, while Huang et al. [HGM18] plan paths for wireframe prints. Peng et al. consider the design of such wire-print objects via a traditional virtual surface modeling tool [PWMG16] and later an augmented reality 3D drawing interface [PBW*18]. These methods focus on wireframe *surfaces* and the design constraints are largely governed by printhead clearance during toolpathing and structural concerns. No assembly is necessary, but structures are smaller and denser.

Recently, Chidambaram et al. [CZS*19] introduce a design tool for wireframe objects constructed via 3D-printed connectors and metal wires. While their tool provides design suggestions, their method does not detect infeasible designs due to overlapping rods and does not alert the user if their design will balance. Their tool computes a stress visualization, but neither complete description of the method nor accuracy validations are provided. In this design space, the (strong) wood undergoes bending and stress concentrates at the (significantly weaker) plastic joints. It is unclear whether the space frames of Chidambaram et al. are the appropriate model. Their method is also restricted both by a hard bound on the length of wires (3cm) and the angle between rods (35°) in order to safely construct 3D-printed connectors by unioning sphere and cylinder geometries. Due to this strict minimum angle constraint, it would be impossible for their system to accommodate the designs in Figures 1 (16°), 3 (26°), 12 (22°), or 14 (30°). Instead, we propose a more general joint construction algorithm that accommodates arbitrary angles, sizes, thicknesses, tolerances and polygonal rod profiles. As a result, our design space is larger and less constraining to the user.

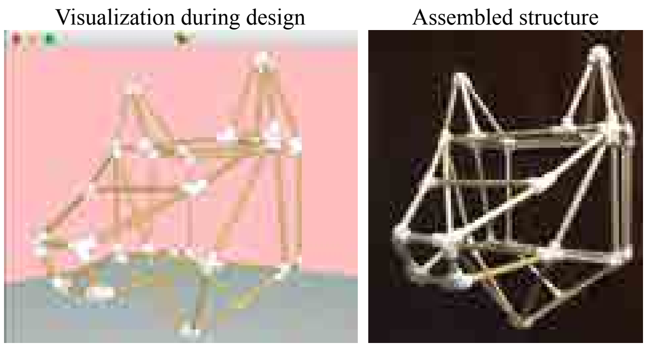


Figure 3: *RSDesigner displays a visualization closely matching the eventual fabrication, reducing surprise at assembly time.*

Chidambaram et al. provide assembly guidance, but only in the form of connector/wire indices and a printed lookup table of rod lengths. Our RSAssembler interface suggests an assembly ordering guided by a focus+context visualization.

Dritsas et al. [DCS17] create a sequence of GRASSHOPPER scripts to aid in the design of structures similar to our results. Given a desired rod diameter, they determine minimum angle of incident joints allowed by their scripts and prevent/reject designs that do not meet this criteria. The generated joints are not guaranteed to be solid models which may cause printer failures. The interactive design or assisted assembly problems are not considered, so the user must (presumably) assemble a collection of similar looking parts.

Magrisso et al. [MMZ18] propose a user-assisted process to generate 3D-printable carpentry joinery. Their goal is different from ours. They seek to enhance traditional manual carpentry with advanced manufacturing of individual joints, without placing a strong emphasis on real-time feedback of a tight design loop for the overall object. This process creates intricate joineries. The design remains creative, but also relies on the user for non-creative tasks such as supervision of the heuristic when it fails and tuning parameters to recover a feasible design. Our, in comparison, modest joint generation is fully automatic. This allows the user to focus on the creative task of designing the overall object, facilitated by immediate feedback and accurate visualization. The user never concerned with the precise meshing or representation of the joint geometry, only the high-level design of the structure. Tian et al. [TSC*18] create a library of CNC-millable joineries to create a woodworking interface. These beautiful results utilize a different and complementary fabrication process and design space.

We are inspired by the early interactive exploration work of Umetani et al. [UIM12]. Our contributions are complementary: their method considers loads on panel-based furniture, but does not consider intersections that would prevent construction during design exploration. Later, Garg et al. [GJG16] visualize collisions during choreography and arrangement of space-time reconfigurables, but do not consider geometric modeling.

On a larger scale than ours, Yoshida et al. [YIO*15] propose a design tool and additive manufacturing process to construct architecture-scale structures out of unstructured chopsticks and glue. At this scale fused rods behave as a 3D texture or homogenized material for the shell of the structure. In contrast, we focus on designs where the rods dominate both the structure's form and function.

Our design-to-assembly system shares common high-level goals as [AGWF15, HAW16], who consider the guided design and assembly of pop-up books and dynamic papercraft objects. Agrawala et al. [APH*03] distill instructions diagrams from an input 3D object, while Shao et al. [SLR*16] reverse engineer an editable 3D object from instruction drawings.

2.2. Joint Geometry Construction

Joint geometry generation requires more than simple wireframe meshing. For example, BLENDER's Wiremesh Modifier is guaranteed to generate quadrilateral meshes which is convenient for Catmull-Clark subdivision and other post-processing, but this method only takes as input edges of a *surface* mesh. Panotopoulou et al. [PRW*19] extend this idea to arbitrary edge-networks by connecting together variable diameter quadrilateral meshes along each input edge. Their method minimizes but does not remove the *twisting* of the mesh faces along the segment. Unfortunately, any amount of twist is problematic for non-circular profile rods (see Fig. 17).

Tonelli et al. [TPCS16] create structures from 3D-printed joints and wooden rods. Their process is not fully automatic and they only consider the wireframe of a surface mesh specifically designed to avoid acute angles between edges. From visual inspection, the method is unlikely to generalize. Assembly is even more tedious without a guide like RSAssembler: the joints and rods have been implicitly optimized to have slightly different geometry. Their main example took roughly two days to assemble.

Many examples of 3D-printed joints and connectors for furniture-scale structures can be found online. For example, Gellért [Gel15] has gathered a library of modular 3D-printed connectors for panels to create shelving. Cegar [Ceg14] constructs 3D-printed joints to connect wooden rods at 0° and 90° angles. The startup DesignLibero has a series of furniture and light fixtures composed of wooden rods and (presumably custom-designed) 3D-printed joints [Rut18]. Fried [Fri16] has posted a GRASSHOPPER script to generate node geometry for connecting (presumably only) circular profile rods. We are inspired by these designs and hope that our reproducible technical description of joint geometry construction as well as our novel user interfaces encourage this direction of hybrid design.

Hart's wiremesh generation method [Har06] (e.g., as implemented in PYMESH [Zho19] or LIBIGL [JP*19]) provides the foundation for our method. We identify and correct a few flaws in this method and then extend it to generate solid geometry compatible for building solid and consistent joints.

3. Fabrication-Aware Design Interface

Our investigation is driven by the goal of facilitating the design of rod-joint structures. Rod-joint structures afford a harmonious division of complexity. Complex geometry is delegated to the joints, fabricated by a 3D printer designed for such a task, while rods retain their intrinsic strength and require only perpendicular cuts. Introducing precision 3D-printing into the design task significantly increases the development time: printing the joints for the guitar stand in Fig. 1

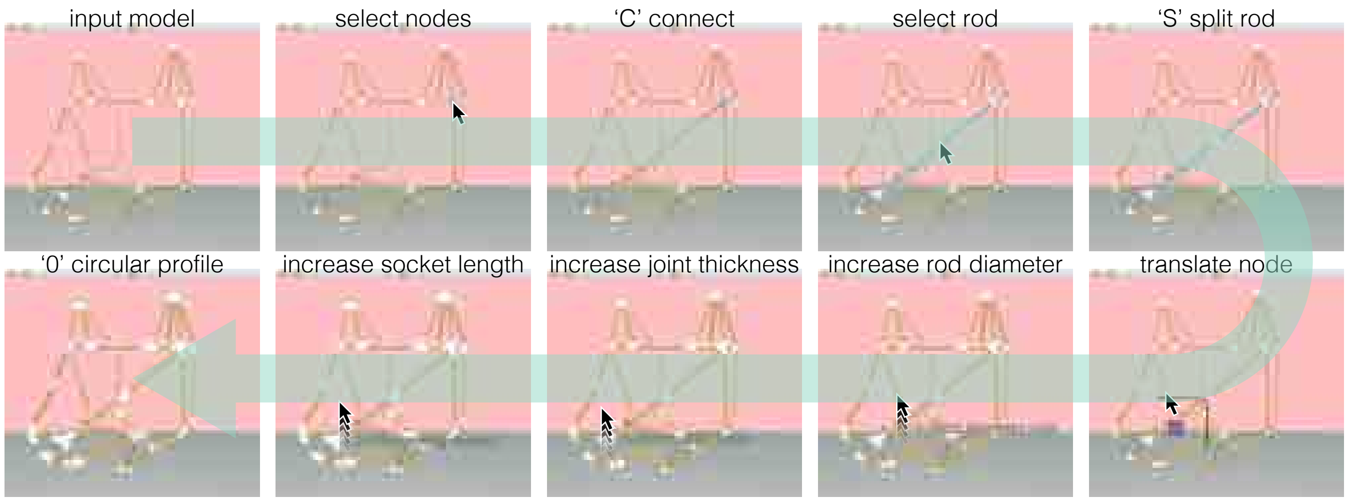


Figure 4: *The user of our design tool may conduct a variety of direct manipulation mouse-based editing operations and hotkey commands. Manipulated values of continuous parameters appear on screen directly next to the draggin cursor.*

required 10 hours and 10 more hours to remove dissolvable supports. We would like to reduce the reliance on time-consuming 3D-printing during design as well as reduce the probability of fabrication, structural, or assembly failure. To this end, we introduce a minimal set of virtual design tools. Discarding potential but unnecessary tools is just as important as retaining the most effective ones. For this reason, we have written our design tool as a stand-alone application rather than a plugin to a monolithic commercial CAD tool. For example, existing CAD tools do not deal with intersections well [GJG16]; some tools will simply crash and others will throw an error.

The invariant we will maintain in our design tool is a 3D rendering of the resulting rod-joint structure (see Fig. 3). Three-dimensional joint geometries are rendered in white (i.e., 3D-printed plastic) and rod geometries in brown (i.e., wood). We expose the following editing operations to the user:

- translate, rotate, and scale selected node positions using a standard 3D manipulation widget,
- connect selected joints with new rods,
- split a selected rod by inserting a joint at the midpoint,
- drag on any rod to directly manipulate rod diameter ($2r$),
- drag on any joint to directly manipulate the joint wall thickness (σ) or the socket length (h), and
- choose the number of sides of the polygonal rod profile (or choosing a circular profile).

See Fig. 4 and our accompanying video for interaction sessions demonstrating each editing interaction. After any edit, the joint and rod geometries are immediately updated. When manipulating the sizing parameters (r,σ,h), the new value is displayed next to the cursor during mouse dragging.

3.1. Detecting and highlighting problems

Not all edge-networks and parameter combinations are fabricable. We introduce a set of tools to help the user detect potential problems

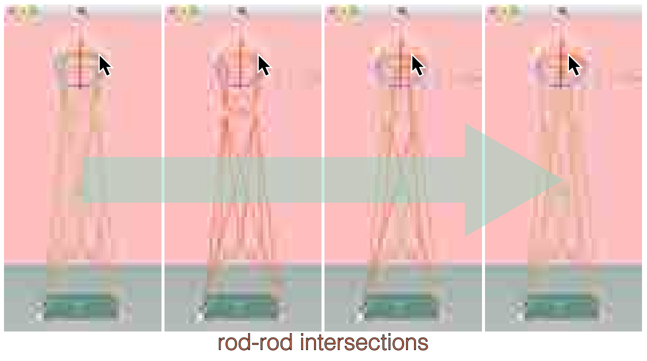


Figure 5: *RSDesigner allows the user to manipulate nodes* through *infeasible designs, highlighting issues (e.g., rod-rod intersections) interactively so the user can creatively explore toward a fabricable design. Real-time interaction is key.*

during virtual design before wasting time trying to fabricate an impossible design. In the physical world, two rods cannot occupy the same space. In Section 4, we will carefully construct joint geometry to prevent such rod-rod intersections from happening *locally* at joints. Rod-rod intersections can also occur *globally* between rods that do not share any joints. We robustly detect rod-rod intersections using the LIBIGL geometry processing library [JP*19] and immediately highlight problematic rods in red. We do not *prevent* the user from making invalid designs. It is often desirable to traverse through invalid states into a new valid state (see Fig. 5). Our real-time feedback allows visual feasibility tracking during any edit.

The angles of rods incident on a joint and the rod/joint thickness parameters determine the ultimate geometry of a joint. If the joints become too large or the rods between them too small, the joint geometries will overlap, *swallowing* a rod (in the notation of Appendix A, if $g_{ij} + g_{ji} + 2h > \ell_{\{i,j\}}$). We immediately highlight such problematic joints in red, alerting the user of an inefficient or undesirable design (see Fig. 6).

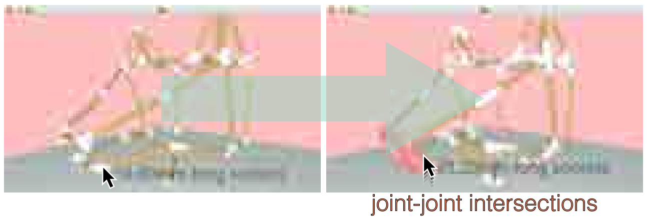


Figure 6: *The user drags on the joint to increase the socket length. If the joints become too large and intersect, RSDesigner highlights them to alert the user of an infeasible design.*

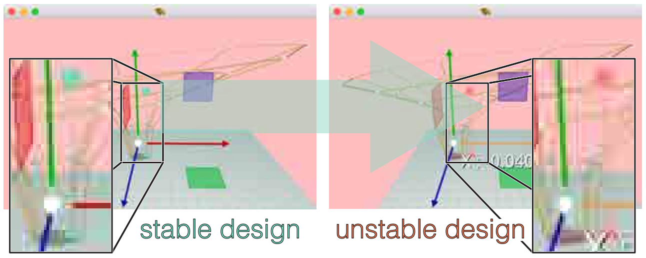


Figure 7: *The user translates the base of this pavilion slightly to the left and RSDesigner highlights that the design will no longer balance, by coloring the center of mass and support polygons red.*

We also help the user determine whether the current design will stand. If the center of mass projected onto the ground falls outside of the support polygon, the design is deemed unstable (see, e.g., [PWLSH13]), and we alert the user by highlighting the center of mass and support polygon red (see Fig. 7).

The effectiveness of our design tool hinges on the ability to efficiently and fully automatically generate general and fabricable joint geometries and rod lengths. We now turn our attention to constructing and then fabricating these geometries.

4. Geometry & Fabrication

The physical realization of designs created with RSDesigner relies on a simple yet novel algorithm to generate 3D-printable joint geometries and precision rod lengths. This algorithm must robustly handle arbitrary joint angles, joint valences, and rod profiles. While only trigonometry is required, a general implementation must avoid a variety of pitfalls. We provide a detailed description in Appendix A.

4.1. 3D-Printed Joints

The resulting joint geometries ($\mathcal{J}_i$ in Appendix A) are 3D printed using heuristic to pick a printing direction that minimizes support material placed *inside* the cavities at each socket (the most difficult to dissolve/remove). We compute the rotation that aligns the average edge-vector ($\bar{\mathbf{w}}_i = \sum_{\{i,j\}} \hat{\mathbf{w}}_{ij}$ in the notation of Appendix A) with the printing extrusion direction (similar to Equation (1)). We use existing software (GrabCADPrint or Simplify3D) to pack the rotated 3D geometries into the smallest number of build volumes.

Each joint is automatically engraved with a two-digit identification number. This is achieved fully automatically. We start by

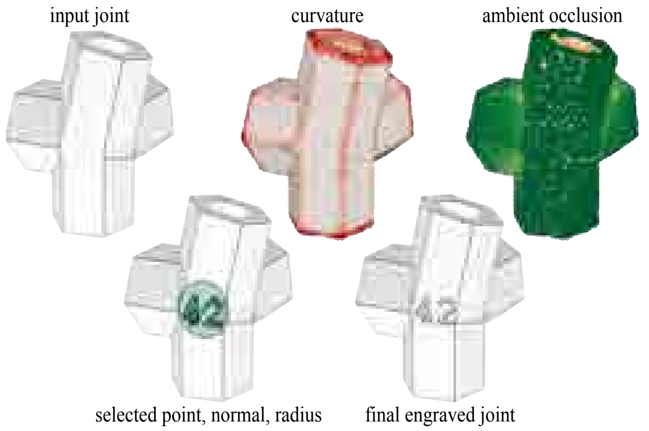


Figure 8: *Our subroutine geometrically engraves ids into each joint part using curvature and occlusion to find a readable location.*

oversampling the joint geometry at 10,000 uniformly random locations. We estimate curvature at each sample by taking the distance-weighted average of dihedral angles between the $k = 200$ nearest neighbors. We compute ambient occlusion at each sample with respect to the original geometry. We select the sample with the smallest sum of these two values and set the engraving's radial extent to the distance to furthest of the sample's k neighbors. Text geometry with a thickness of $\sigma/2$ is placed accordingly and subtracted from the joint geometry via [ZGZJ16] (see Fig. 8). The curvature and occlusion term encourages the engraving to be centered on a flat and visible region, respectively. This simple method was surprisingly effective. The label visibility allows RSAssembler to better guide the assembly described in the next section. It would be interesting to extend our heuristics to consider perceptual preference [ZLP*15].

4.2. Precision-Cut Rods

In theory, any cutting method could be used to cut the m rods (e.g., a traditional hand saw and miter box). However, often each rod has a unique length and the setup/measuring time of each cut starts to dominate for manual methods when the number of rods m is large. Instead, we use a laser cutter to precisely cut all rods (or as many will fit into the machine at once) simultaneously. To maximize the efficiency of this process, we solve a one-dimensional "bin-packing". That is, given a set of m desired lengths to cut and raw factory uncut rods of factory-length b (e.g., $b = 1\text{m}$), we find the assignment of cuts to uncut rods that minimizes waste and uses the fewest uncut rods (see Fig. 2). We implemented a "first-fit" algorithm with multiple random orderings and taking the best packing (the optimal fit, while NP-hard to compute, will correspond to the first-fit result of *some* ordering [Lew09]). We use bin sizes of $b-2p$, where p is a padding amount (e.g., $p = 1\text{cm}$) to account for rough "factory" cuts at rod ends. The optimized cuts drawn as line segments in a `.svg` file sent to the laser cutter to be cut in a single job (see Fig. 2).

We experimented with laser-engraving each rod during cutting in the hopes to help assembly. We found this to be unnecessary and, in fact, confusing. Instead, it was much faster to simply presort all of the cut rods by length and have a ruler/calipers nearby for selection.

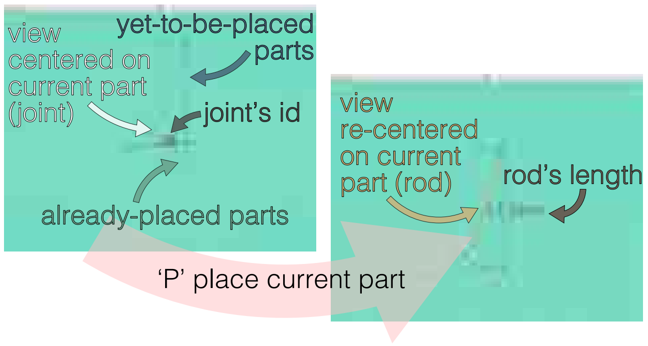


Figure 9: *The RSAssembler interface embraces a focus+context design. The current object is centered and the zoom is initialized to fit the entire object in view in case the user wishes to tumble the camera. (Screenshots are intentionally* not *cropped to the object.)*

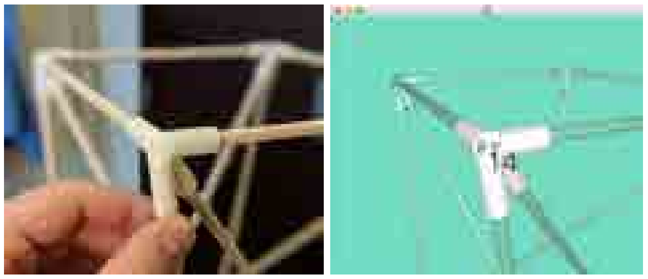


Figure 10: *The joint geometries shown by RSAssembler match the 3D printed. Engraved joint IDs also facilitate orientation matching.*

5. Guided Assembly Interface

After physical fabrication, the design is reduced to a large number of 3D printed of joints and laser-cut rods. For complex designs (e.g., the wolf head in Fig. 3), there are many similar looking joints and similar length rods. Incorrect placement of a rod is often not a problem as this *length* error will diffuse through the design. In stark contrast, incorrect placement of a joint leads to an *angle* error that grows linearly with distance and can cause a Domino-effect of misalignment, infesting the entire design. To help the user avoid such assembly disasters, we propose a guided assembly interface.

The user starts by organizing the 3D-printed joints (e.g., sorted by engraved id) and laser-cut rods (e.g., sorted by length). On a nearby screen (e.g., currently our app runs on a MacBook, but could be ported to a tablet/phone), our guided assembly tool shows a 3D visualization of the design. The user can manipulate the camera parameters to view the design from any direction. After placing a rod or joint into position, the user hits a hotkey and the interface proceeds to the next suggested part to place.

Our guided assembly interface exercises focus+context [CKB08] (see Fig. 9): the current part to be placed is rendered in full color matching the physical counterpart (e.g., white for 3D printed joints and brown for wooden rods). Already-placed parts are shown to provide context but *recessed* out of focus by shifting their color toward the (non-white) background color (e.g., teal). Yet-to-be-placed parts are abstracted as dots (joints) and line segments (rods). When the user signals that the current part has been placed, the 3D camera smoothly transitions to focus on the next part. That part is placed at the center of the screen and viewed at a distance so that all connected parts will fit into view when rotating the camera. For joints, the id number is show in a large font and the geometry (with engraving) is shown in a high-contrast rendering style to assist in matching the correct orientation (see Fig. 10). For symmetric joints, the engraved ID replicated in the RSAssembler display serves as a further registration mark. For rods, the length is displayed. See the accompanying video for a full guided assembly sequence.

Random assembly order would require significant context switching, both visually and physically. In addition, we found that is much more difficult to merge multiple sub parts than to add pieces one-by-one. We experimented with various assembly order heuristics breadth-first, rod length priorities, etc.), and ultimately found that a depth-first traversal of the nodes in the edge-network works best. This strategy is guaranteed to produce a complete ordering, regardless of the size of the input. After placing a joint, the tool suggests each not-yet-placed incident rod and then proceeds to the next not-yet-placed adjacent joint. This ensures that the set of already-placed parts is always connected and that the user often adds a new joint adjacent to the last added joint. In our experiments, structural stability of the partially assembled objects was not an issue, though as a future improvement this could be taken into account in the ordering (or even the possibility of adding temporary assembly-only rods).

6. Results & Discussion

We 3D print the joints in our results using a Stratasys F170 in ABS Ivory plastic with dissolvable QSR support material. We laser cut our rods using a Trotec Speedy400 Flexx, which has a 1m×0.6m build plate. We use a variety of different rod materials purchased from a local hardware store: 6.35mm and 12.7mm diameter round hardwood dowels and 10.5mm^2 square-profile wooden molding. The bottleneck in our end-to-end design-to-assembly system is by far the 3D-printing and support-material dissolution. Our slow 3D-printer took more than 10 hours for most examples (10-20 joints) and automatic support-material-removing bathing took nearly as long. Fortunately, the entire 3D-printing process is fully automatic (aside from moving parts from the printer to the bath). Using a "comb" jig to hold rods in place, cutting takes a few minutes. Assembly itself lasted under 45 minutes for all examples included in this paper: the wolf head in Fig. 3 taking longest with 28 joints and 52 rods.

We printed a small board containing sample sockets of varying engineering tolerances ε to determine good values for each rod radius and profile we used. This way we avoid printing joints that ultimately do not friction fit our rods (due to inaccuracies of either part). Tight friction fit joints allow us to avoid the use of tools or screws during assembly. A locking mechanism would be an interesting alternative.

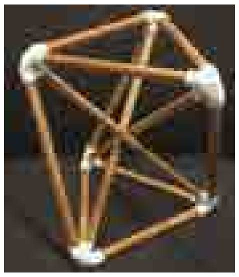


Figure 11: *A close fit.*

The rod-rod intersection testing allows designers to create structures with closely packed but not intersecting rods. In Fig. 11, the

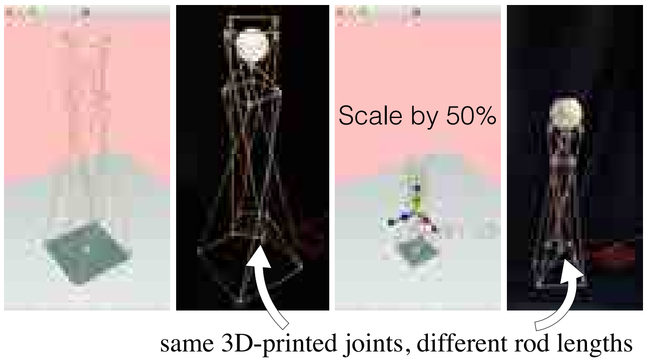


Figure 12: *Isotropic scaling the design will not change the joint geometry, so the same 3D prints may be reused at different scales.*

triangular prism shape was twist just until a collision occurred. Indeed, the assembled rods are nearly touching. Finding designs like this without a virtual real-time feedback interface like RSDesigner would be tedious and difficult.

An interesting design feature that serendipitously emerged from exploring in RSDesigner is that when scaling the node positions isotropically, the joint geometry remains identical: only the rods shrink/grow uniformly. This informs us that the same 3D printed joints can be used to create different scales of the same object: just replace the rods. We realized this feature in the lamp design in Fig. 12. A 1.4m-tall lamp is first constructed, then all rods are cut in half and the lamp is reassembled as a 0.7m-tall lamp.

We do *not* perform finite-element analysis or optimize the design for structural stability beyond balance. Nonetheless the structures we create are sturdy, inheriting the strength of the wooden rods. In Fig. 13, we demonstrate a minimalist coffee-table design. This table is now in regular use at an office environment.

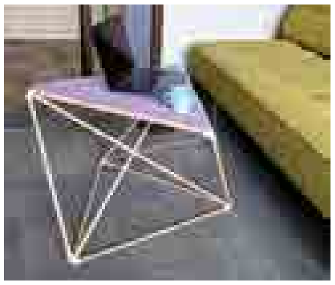


Figure 13: *This tailored coffee table inherits the strength of hardwood dowels to supports not only itself, but also top panel and laptop (table in daily use for six months).*

Disassembled, our structures are compact (see Fig. 1). Rods can be cut from standard-size dowels. Transportation could be as simple as shipping the joints and cutting rods on site. Although circular-profile dowels are the most common and cheapest rods purchased at a hardware store, our entire system supports polygonal profile rods. In Fig. 14, a decorative bowl is designed using square-profile molding for rods.

Unlike cheap, mass-manufactured guitar stand, the unique design shown in Fig. 1 represents an aesthetic discovered by the user during modeling. The interface tools and on-the-fly evaluation of RSDesigner allow a user to fine tune a design in a tight loop.

7. Limitations & Future Work

Our design-to-assembly workflow is not without limitations. We focused on mainly furniture-scale results; although our tools will function correctly at any scale, the size of the structure is bound on the small end by the precision of the 3D printer and on the end by the length of the largest rod. We have also left various incremental improvements to design space as future work: e.g., non-regular convex polygon profiles, varying the rod radius r across edges, varying thickness σ across joints, adding point loads to adjust the center-of-mass. We are intrigued by the idea of adding fine-scale decorative patterns [STG16] or Voronoi duals [MMZ18] to our joint geometries. Very large or volume-filling designs (e.g., trusses) might raise performance concerns and require a dynamic bounding-volume hierarchy (cf. [GJG16]) for intersection testing etc. Our assemblies only required a single person guided by RSAssembler. We forgo a formal user study to confirm that guided assembly is better than having no assembly instructions [KSW*17] or a static plan [TPCS16]. It would be interesting to use crowds or robots to build architectural scale structures *à la* Lafreniere et al. [LGA*16].

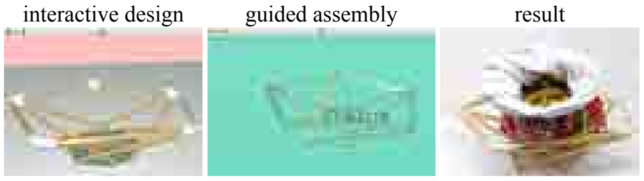


Figure 14: *The square-profile of the rods in this design affect not just the appearance but also the constructed joint geometries.*

If we idealize the joints and rods as perfectly rigid objects, then all our results are mathematically impossible to assemble. We are reliant on compliance. For our sparse, 1D structures reachability was not a major concern (cf. panel-based structures, [UIM12]).

The full power of the laser cutter was not truly leveraged. We use the laser cutter out of availability (we have one), convenience (easier than sawing), and efficiency (faster, too). For the results here, a robotic rod cutter might be more appropriate. In future work, it would be interesting to exploit at least the two-dimensional capabilities of laser cutting in conjunction with our interfaces.

While our lamps and coffee tables can withstand light external loads, larger furniture or structural shapes require consideration of their use and environment [UIM12, WOM*17]. In future work, we would like to consider structural properties of joints [MIJ14] and directly incorporate loads from non-trivial contact and friction with external objects into our design tool (i.e., beyond the point loads of [UIM12]). We have already entertained interest from architects to adapt our joint generation for architectural-scale objects (see, e.g., Fig. 15). We are excited by the recent work of Yoshida et al. [YLI19], who build structures out of found tree branches. Curved or arbitrarily shaped rods could be another direction for future research.

Figure 15: *RodSteward could extend design-to-assembly beyond furniture: a rendered pavilion.*

Acknowledgements

This research is funded in part by NSERC Discovery (RG-PIN2017–05235, RGPAS–2017–507938), New Frontiers of Research Fund (NFRFE–201), the Ontario Early Research Award program, the Canada Research Chairs Program, the Fields Centre for Quantitative Analysis and Modelling and gifts by Adobe Systems, Autodesk and MESH Inc. We are grateful to Tovi Grossman, Fanny Chevalier, Zhicong Lu, Sarah Kushner, Leo Sacht and Eitan Grinspun for feedback on drafts of this paper. We thank David Levin for illuminating discussions and John Hancock for infrastructural support.

References

[AGWF15] ANNETT M., GROSSMAN T., WIGDOR D. J., FITZMAURICE G. W.: Moveablemaker: Facilitating the design, generation, and assembly of moveable papercraft. In *Proc. UIST* (2015).

[APH*03] AGRAWALA M., PHAN D., HEISER J., HAYMAKER J., KLINGNER J., HANRAHAN P., TVERSKY B.: Designing effective step-by-step assembly instructions. *ACM TOG* (2003).

[AUK*15] AGRAWAL H., UMAPATHI U., KOVACS R., FROHNHOFEN J., CHEN H.-T., MUELLER S., BAUDISCH P.: Protopiper: Physically sketching room-sized objects at actual scale. In *Proc. UIST* (2015).

[BKLZ17] BENES B., KASIK D. J., LI W., ZHANG H.: Computational design and fabrication. *IEEE CG&A* (2017).

[Ceg14] CEGAR I.: 3d printed joints, 2014. URL: www.iaacblog.com/programs/3d-printed-joints-2/.

[CKB08] COCKBURN A., KARLSON A. K., BEDERSON B. B.: A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.* (2008).

[CZS*19] CHIDAMBARAM S., ZHANG Y., SUNDARARAJAN V., ELMQVIST N., , RAMANI K.: Shape structuralizer: Design, fabrication, and user-driven iterative refinement of 3d mesh models. In *Proc. CHI* (2019).

[DCS17] DRITSAS S., CHEN L., SASS L.: Small 3d printers / large scale artifacts: Computation for automated spatial lattice design-to-fabrication with low cost linear elements and 3d printed nodes. *Int. Conf. of the Ass. for Comp.-Aided Arch. Design Research in Asia* (2017).

[Fri16] FRIED C.: Node generator, 2016. URL: www.grasshopper3d.com/forum/topics/node-genetor.

[Gel15] GELLÉRT O.: Print to build: Joint collection for variable space structures, 2015. URL: www.behance.net/gallery/27812109/Print-To-Build-3D-printed-joint-collection.

[GJG16] GARG A., JACOBSON A., GRINSPUN E.: Computational design of reconfigurables. *ACM TOG* (2016).

[Har06] HART G. W.: Sculptural forms from hyperbolic tessellations. In *IEEE Int. Conf. on Shape Modeling and Applications* (2006).

[HAW16] HARQUAIL N., ALLEN M., WHITING E.: Foldlings: A tool for interactive pop-up card design. In *Proc. GraDiFab* (2016).

[HGM18] HUANG Y., GARRETT C. R., MUELLER C. T.: Automated sequence and motion planning for robotic spatial extrusion of 3d trusses. *Construction Robotics* (2018).

[HZH*16] HUANG Y., ZHANG J., HU X., SONG G., LIU Z., YU L., LIU L.: Framefab: Robotic fabrication of frame shapes. *ACM TOG* (2016).

[JP*19] JACOBSON A., PANOZZO D., ET AL.: libigl: A simple C++ geometry processing library, 2019. libigl.github.io/libigl/.

[KIL*18] KOVACS R., ION A., LOPES P., OESTERREICH T., FILTER J., OTTO P., ARNDT T., RING N., WITTE M., SYNYTSIA A., BAUDISCH P.: Trussformer: 3d printing large kinetic structures. In *Proc. CHI* (2018).

[KSW*17] KOVACS R., SEUFERT A., WALL L., CHEN H.-T., MEINEL F., MÜLLER W., YOU S., BREHM M., STRIEBEL J., KOMMANA Y., POPIAK A., BLÄSIUS T., BAUDISCH P.: Trussfab: Fabricating sturdy large-scale structures on desktop 3d printers. In *Proc. CHI* (2017).

[LEM*17] LIVESU M., ELLERO S., MARTÍNEZ J., LEFEBVRE S., ATTENE M.: From 3d models to 3d prints: An overview of the processing pipeline. *Comp. Graph. Forum* (2017).

[Lew09] LEWIS R.: A general-purpose hill-climbing method for order independent minimum grouping problems: A case study in graph colouring and bin packing. *Computers & OR* (2009).

[LGA*16] LAFRENIERE B., GROSSMAN T., ANDERSON F., MATEJKA J., KERRICK H., NAGY D., VASEY L., ATHERTON E., BEIRNE N., COELHO M. H., COTE N., LI S., NOGUEIRA A., NGUYEN L., SCHWINN T., STODDART J., THOMASSON D., WANG R., WHITE T., BENJAMIN D., CONTI M., MENGES A., FITZMAURICE G.: Crowd-sourced fabrication. In *Proc. UIST* (2016).

[LRL17] LEEN D., RAMAKERS R., LUYTEN K.: Strutmodeling: A low-fidelity construction kit to iteratively model, test, and adapt 3d objects. In *Proc. UIST* (2017).

[MBM*15] MUELLER S., BEYER D., MOHR T., GUREVICH S., TEIBRICH A., PFISTERE L., GUENTHER K., FROHNHOFEN J., CHEN H.-T., BAUDISCH P., IM S., GUIMBRETIÈRE F.: Low-fidelity fabrication: Speeding up design iteration of 3d objects. In *Proc. CHI* (2015).

[MIG*14] MUELLER S., IM S., GUREVICH S., TEIBRICH A., PFISTERER L., GUIMBRETIÈRE F., BAUDISCH P.: Wireprint: 3d printed previews for fast prototyping. In *Proc. UIST* (2014).

[MIJ14] MUELLER C. T., IRANI A., JENETT B. E.: Additive manufacturing of structural prototypes for conceptual design. In *Proc. Int. Ass. of Shell and Spatial Structures* (2014).

[MMZ18] MAGRISSO S., MIZRAHI M., ZORAN A.: Digital joinery for hybrid carpentry. In *Proc. CHI* (2018).

[PBW*18] PENG H., BRIGGS J., WANG C.-Y., GUO K., KIDER J., MUELLER S., BAUDISCH P., GUIMBRETIÈRE F.: Roma: Interactive fabrication with augmented reality and a robotic 3d printer. In *Proc. CHI* (2018).

[PRW*19] PANOTOPOULOU A., ROSS E., WELKER K., HUBERT E., MORIN G.: Scaffolding a skeleton. *Association for Women in Mathematics* (2019).

[PWLSH13] PRÉVOST R., WHITING E., LEFEBVRE S., SORKINE-HORNUNG O.: Make It Stand: Balancing shapes for 3D fabrication. *ACM TOG* (2013).

[PWMG16] PENG H., WU R., MARSCHNER S., GUIMBRETIÈRE F.: On-the-fly print: Incremental printing while modelling. In *Proc. CHI* (2016).

[Rut18] RUTILO L.: New family of hybrid design products created with 3d printed joints by italian startup designlibero, 2018.

[SLR*16] SHAO T., LI D., RONG Y., ZHENG C., ZHOU K.: Dynamic furniture modeling through assembly instructions. *ACM TOG* (2016).

[STG16] SCHUMACHER C., THOMASZEWSKI B., GROSS M. H.: Stenciling: Designing structurally-sound surfaces with decorative patterns. *Comp. Graph. Forum* (2016).

[TPCS16] TONELLI D., PIETRONI N., CIGNONI P., SCOPIGNO R.: Design and fabrication of grid-shells mockups. In *Smart Tools and Apps for Graphics* (2016).

[TSC*18] TIAN R., STERMAN S., CHIOU E., WARNER J., PAULOS E.: Matchsticks: Woodworking through improvisational digital fabrication. In *Proc. CHI* (2018).

[UBM15] UMETANI N., BICKEL B., MATUSIK W.: Computational tools for 3d printing. In *ACM SIGGRAPH Courses* (2015).

[UIM12] UMETANI N., IGARASHI T., MITRA N. J.: Guided exploration of physically valid shapes for furniture design. *ACM TOG* (2012).

[WOM*17] WHITING E., OUF N., MAKATURA L., MOUSAS C., SHU Z., KAVAN L.: Environment-scale fabrication: Replicating outdoor climbing experiences. In *Proc. CHI* (2017).

[WPGM16] WU R., PENG H., GUIMBRETIÈRE F., MARSCHNER S.: Printing arbitrary meshes with a 5dof wireframe printer. *ACM TOG* (2016).

[YIO*15] YOSHIDA H., IGARASHI T., OBUCHI Y., TAKAMI Y., SATO J., ARAKI M., MIKI M., NAGATA K., SAKAI K., IGARASHI S.: Architecture-scale human-assisted additive manufacturing. *ACM TOG* (2015).

[YLI19] YOSHIDA H., LARSSON M., IGARASHI T.: Upcycling tree branches as architectural elements through collaborative design and fabrication. In *Proc. TEI* (2019).

[ZGZJ16] ZHOU Q., GRINSPUN E., ZORIN D., JACOBSON A.: Mesh arrangements for solid geometry. *ACM TOG* (2016).

[Zho19] ZHOU Q.: Pymesh, 2019. URL: pymesh.readthedocs.io.

[ZLP*15] ZHANG X., LE X., PANOTOPOULOU A., WHITING E., WANG C. C. L.: Perceptual models of preference in 3d printing direction. *ACM TOG* (2015).

Appendix A: 3D-Printable Solid Joint Geometry

The input to our geometry construction algorithm is a 3D edge-network, i.e., a graph embedded in $\mathbb{R}^3$, composed of a list of n node positions $\mathbf{p}_i \in \mathbb{R}^3$ for $i \in \{1,\ldots,n\}$ and a list of m *undirected* edges as pairs $\{i,j\} \in \{1,\ldots,n\}^2$ where we use the equivalence relation $\{i,j\} = \{j,i\}$.

In this technical section, we use subscript notation such that $a_{\{i,j\}} = a_{\{j,i\}}$ refers to a quantity associated with the undirected edge $\{i,j\}$, whereas b_{ij} refers to a quantity associated with end-point i on the edge $\{i,j\}$ and in general $b_{ij} \neq b_{ji}$. In most cases, the difference will also be clear from context.

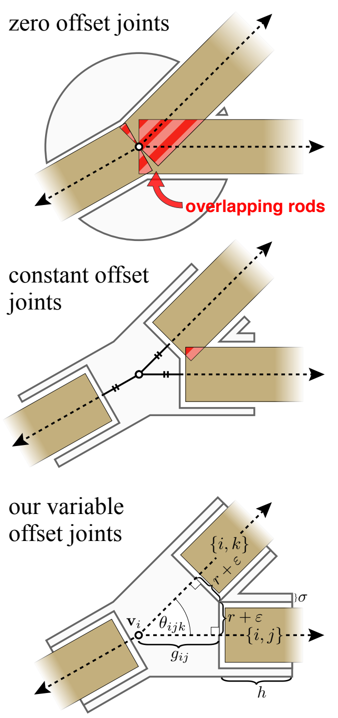
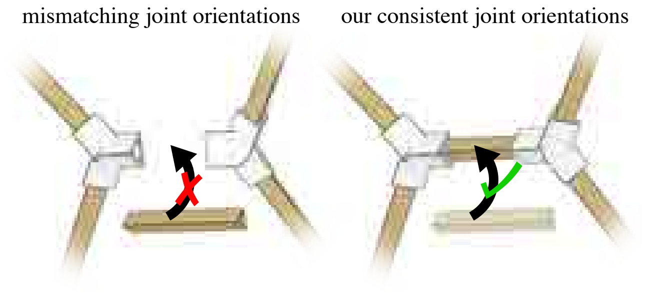


Figure 16: *Previous joint geometry generation methods do not consider the possibility of rod intersections at joints. A per-edge offset is necessary and can be minimized on a case-by-case basis.*

The algorithm is controlled by a number of user-defined parameters (see Fig. 16, right): r the radius of the rods, p the number of sides on the polygonal cross-sectional profile of the rods (without loss of generality, we will assume these polygons are regularly shaped), σ the thickness of the 3D-printed *joints* encasing each node, h the amount that joints overhang along incident rods, and ε the "engineering tolerance" (possibly negative for friction fitting) between the joints and rods.

The output of our method includes n solid meshes representing the surfaces of the 3D-printable joint geometry at each node and m precise rod lengths to cut. As seen in Fig. 16, the physically realizable length of the rod of each edge $\{i,j\}$ will generally be less than the raw edge length ($\|\mathbf{p}_i - \mathbf{p}_j\|$). Instead, the precise lengths will be implicitly determined by the geometry of the joints at either node.



mismatching joint orientations
our consistent joint orientations

Figure 17: *Direct extensions of wiremeshing algorithms (e.g., [PRW*19, Har06]) may result in* twisting *along edges, leading to inconsistent rod-orientations at either end. Our joint-construction algorithm avoids this.*

The geometry of the joint at each node will be an independent solid object, but we require that the outlets at either joint incident on an edge to be *consistent* so that polygonal-cross-section rod geometry can be rotated to fit either end Fig. 17.

A useful subroutine is to generate the primitive geometry of a solid mesh of a generalized cylinder with the profile of a p-sided polygon. This is accomplished by extruding a regular p-gon inscribed in the unit circle of the xy plane along the z-axis for one unit.

This unit-cylinder mesh geometry can then be transformed to lie along any given edge. For each edge $\{i,j\}$, we compute a 3D rotation $\mathbf{R}_{\{i,j\}} \in SO(3)$ aligning the z-axis vector $\mathbf{e}_z = (0,0,1)^\top$ to its unit edge vector $\hat{\mathbf{w}}_{ij} = (\mathbf{p}_j - \mathbf{p}_i)^\top / \|\mathbf{p}_j - \mathbf{p}_i\|$:

$$\mathbf{R}_{\{i,j\}} = \mathbf{I} + [\mathbf{e}_z \times \hat{\mathbf{w}}_{ij}]_\times + \frac{1}{1 + \mathbf{e}_z \cdot \hat{\mathbf{w}}_{ij}} [\mathbf{e}_z \times \hat{\mathbf{w}}_{ij}]_\times^2 \tag{1}$$

where $\mathbf{I} \in \mathbb{R}^{3\times3}$ is the identity matrix and $[\mathbf{x}]_\times$ is the matrix representing cross-product by the vector $\mathbf{x}$:

$$[\mathbf{x}]_\times = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix}. \tag{2}$$

We *could* place rod geometry along each edge $\{i,j\}$ by composing this per-edge rotation with anisotropic pre-scaling along the z-axis by the edge length $\|\mathbf{p}_j - \mathbf{p}_i\|$ and radially in the xy-plane by the desired radius r and a post-translation to the edge tip position $\mathbf{p}_i$. As a per-edge affine transformation:

$$\underbrace{\begin{pmatrix} \mathbf{I} & \mathbf{p}_i \\ 0\,0\,0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}_{\{i,j\}} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0\,0\,0 & 1 \end{pmatrix}}_{\mathbf{T}_{\{i,j\}}} \begin{pmatrix} r & 0 & 0 & 0 \\ 0 & r & 0 & 0 \\ 0 & 0 & \|\mathbf{p}_j - \mathbf{p}_i\| & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3}$$

where $\mathbf{T}_{\{i,j\}} \in \mathbb{R}^{4\times4}$ is a rigid transformation *placing* the rod into the edge-network. However, this naïve rod geometry will result in messy intersections at each node (see Fig. 16, left). In the physical world, we can not allow the rods of multiple edges incident on a node to "share" the space at that node. Offsetting by a uniform amount g (cf. [Har06]) works when g is large relative to r and the angles between incident rods are not very acute. The 3D-printed joints of Tonelli et al. [TPCS16] use a fixed offset, but their results are limited to surface edge-networks with modest angles. For arbitrary edge-networks, acute angles are common. If g is too small relative to r, overlaps will occur even for obtuse angles (see Fig. 16, center). If g is too large, joints become bulky (everywhere) and may even (unnecessarily) envelope small edges.

We *could* remove the rod-intersection volumes (e.g., the red regions in Fig. 16) from the rod geometries, but this would require non-trivial shaving or whittling of the rods. Instead, we move all complex geometry to the 3D-printed joints and use simple straight perpendicular cuts on off-the-shelf rods.

To this end, we compute per-node-edge offsets, where g_{ij} is the offset at node i along the incident edge $\{i,j\}$. The offsets at either end of an edge $\{i,j\}$ will in general be different (i.e., $g_{ij} \neq g_{ji}$). We would like g_{ij} values that: 1) are as small as possible and 2) guarantee that rods will not overlap. We can compute a safe offset g_{ij} by considering the minimum angle formed by edge $\{i,j\}$ and all other edges $\{i,k\}$ with $k \neq j$:

$$\theta_{ij} = \min_k \cos^{-1}\left(\hat{\mathbf{w}}_{ij} \cdot \hat{\mathbf{w}}_{ik}\right) \tag{4}$$

or equivalently the largest dot-product

$$c_{ij} = \cos\theta_{ij} = \max_k \ \hat{\mathbf{w}}_{ij} \cdot \hat{\mathbf{w}}_{ik}. \tag{5}$$

In general, for a node i the smallest angles along different incident edges will not be the same (i.e., $\theta_{ij} \neq \theta_{ik}$). Given the rod radius r and engineering tolerance ε, a safe offset g_{ij} is the solution to a trigonometry problem solved using the tangent half-angle formula:

$$g_{ij} = (r+\varepsilon)\sqrt{\frac{1+c_{ij}}{1-c_{ij}}}. \tag{6}$$

As this formula confirms, the offset tends toward infinity as the angle tends toward zero (and c_{ij} tends toward one).

Armed with offsets that guarantee the absence of rod intersections at joints, we can now generate solid joint geometry. We start by considering every edge $\{i,j\}$. We generate unit-cylinder mesh and scale it radially by $r+\sigma$ and axially by $h+\sigma$. We then place two copies offset axially by $g_{ij}-\sigma$ and $\|\mathbf{p}_j - \mathbf{p}_i\| - h - g_{ji}$, respectively. Both are finally transformed into place by $\mathbf{T}_{\{i,j\}}$. All together, the *tip* and *tail* pieces are transformed, respectively, by:

$$\mathbf{H}_{ij} = \mathbf{T}_{\{i,j\}} \begin{pmatrix} r+\sigma & 0 & 0 & 0 \\ 0 & r+\sigma & 0 & 0 \\ 0 & 0 & h+\sigma & g_{ij}-\sigma \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

and

$$\mathbf{H}_{ji} = \mathbf{T}_{\{i,j\}} \begin{pmatrix} r+\sigma & 0 & 0 & 0 \\ 0 & r+\sigma & 0 & 0 \\ 0 & 0 & h+\sigma & \|\mathbf{p}_j - \mathbf{p}_i\| - h - g_{ji} \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

We denote the transformed solid models as $\mathcal{C}_{ij}$ and $\mathcal{C}_{ji}$ respectively. Though strictly not necessary to generate a solid joint, the $+\sigma$ in the axial scaling ensures a σ-thick "cap" at each end of a rod. For each pair of transformed cylinders, we keep track of the mesh vertices of the cylinder model that end up at either end of the edge. That is, those with projected distance g_{ij} and g_{ji} to the tip and tail nodes, respectively. We call these vertex-sets $\mathcal{V}_{ij}$ and $\mathcal{V}_{ji}$, respectively. Due to the procedural generation and transformation of the cylinder model, this bookkeeping is purely combinatorial and does not require measuring distances *after* transforming each cylinder model.

Now we consider each node of the input edge-network. Like Hart's method [Har06], we compute the convex hull $\mathcal{H}_i$ of the node position $\mathbf{p}_i$ and all vertex-sets $\mathcal{V}_{ij}$ from incident edges $\{i,j\}$. While this convex hull is guaranteed to have *at least* a two-dimensional intersection with each of the incident cylinder models, it is not true (cf. [Har06]) that faces of the cylinder meshes will always appear as faces of the convex hull: some vertices of $\mathcal{V}_{ij}$ may be strictly *inside* the convex hull. We merge the hull model $\mathcal{H}_i$ and the transformed cylinders $\mathcal{C}_{ij}$ of each incident edge $\{i,j\}$ by computing their exact mesh union via [ZGZJ16]. We denote the solid result of this union $\mathcal{U}_i$.

As a side effect of this process, we have determined that the precise length ℓ_{ij} to cut the rod at each edge $\{i,j\}$ is the full edge-length minus the safe offsets computed at either end:

$$\ell_{ij} = \|\mathbf{p}_j - \mathbf{p}_i\| - g_{ij} - g_{ji}. \tag{7}$$

For each edge $\{i,j\}$, we generate unit edge-cylinder mesh and scale it radially by $r+\varepsilon$ and axially by $\ell_{ij}+2\varepsilon$. We then align this geometry with the offsets and transform it into place by by $\mathbf{T}_{\{i,j\}}$. All together, the transformation of the unit-cylinder per-edge is:

$$\mathbf{T}_{\{i,j\}} \begin{pmatrix} r+\varepsilon & 0 & 0 & 0 \\ 0 & r+\varepsilon & 0 & 0 \\ 0 & 0 & \ell_{ij}+2\varepsilon & g_{ij}-\varepsilon \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{8}$$

We denote the transformed model at each edge by $\mathcal{E}_{\{i,j\}}$.

To complete our joint geometry, we consider each node again. We compute the exact solid difference of the joint geometry $\mathcal{U}_i$ and the geometry $\mathcal{E}_{\{i,j\}}$ of all incident edges $\{i,j\}$. The result of this Boolean operation is the final solid joint geometry

$$\mathcal{J}_i = \mathcal{U}_i \setminus \bigcup_{\{i,j\}} \mathcal{E}_{ij}. \tag{9}$$

The mesh boolean operations resulting in $\mathcal{U}_i$ and then $\mathcal{J}_i$ are necessary to create a solid mesh. During interaction with our fabrication-aware interface, the user is unaware that boolean operations are happily skipped as they do not affect the visual appearance.