Interactive Design and Stability Analysis of Decorative Joinery for Furniture

JIAXIAN YAO
University of California, Berkeley
DANNY M. KAUFMAN
Adobe Research
YOTAM GINGOLD
George Mason University
and
MANEESH AGRAWALA
Stanford University

High-quality hand-made furniture often employs *intrinsic joints* that geometrically interlock along mating surfaces. Such joints increase the structural integrity of the furniture and add to its visual appeal. We present an interactive tool for designing such intrinsic joints. Users draw the visual appearance of the joints on the surface of an input furniture model as groups of two-dimensional (2D) regions that must belong to the same part. Our tool automatically partitions the furniture model into a set of solid 3D parts that conform to the user-specified 2D regions and assemble into the furniture. If the input does not merit assemblable solid 3D parts, then our tool reports the failure and suggests options for redesigning the 2D surface regions so that they are assemblable. Similarly, if any parts in the resulting assembly are unstable, then our tool suggests where additional 2D regions should be drawn to better interlock the parts and improve stability. To perform this stability analysis, we introduce a novel variational static analysis method that addresses shortcomings of the equilibrium method for our task. Specifically, our method correctly detects sliding instabilities and reports the locations and directions of sliding and hinging failures. We show that our tool can be used to generate over 100 joints inspired by traditional woodworking and Japanese joinery. We also design and fabricate nine complete furniture assemblies that are stable and connected using only the intrinsic joints produced by our tool.

Author Gingold was supported in part by the United States National Science Foundation (IIS-1451198 and IIS-1453018) and a Google research award.

Authors' addresses: J. Yao, University of California, Berkeley, CA; email: jiaxian.yao@eecs.berkeley.edu; D. M. Kaufman, Adobe Systems, Seattle, WA; email: kaufman@adobe.com; Y. Gingold, George Mason University, Fairfax, VA; email: ygingold@gmu.edu; M. Agrawala, Stanford University, Stanford, CA; email: maneesh@cs.stanford.edu.

0730-0301/2017/03-ART20 $15.00
DOI: http://dx.doi.org/10.1145/3054740

CCS Concepts: • **Human-centered computing → Graphical user interfaces**; • **Computing methodologies → Volumetric models;**

Additional Key Words and Phrases: Joinery, furniture, geometric design, fabrication

ACM Reference Format:

JiaXian Yao, Danny M. Kaufman, Yotam Gingold, and Maneesh Agrawala. 2017. Interactive design and stability analysis of decorative joinery for furniture. ACM Trans. Graph. 36, 2, Article 20 (March 2017), 16 pages.
DOI: http://dx.doi.org/10.1145/3054740

1. INTRODUCTION

Modern ready-to-assemble furniture made by manufacturers like IKEA is typically composed of a set of parts that are connected together via *external fasteners* such as screws, nails, bolts, and pegs. In contrast, high-quality hand-made furniture often employs *intrinsic fasteners* or *joints* that are formed by carefully crafting the geometries of the parts to interlock along mating surfaces. While such joints are valued for their structural integrity and aesthetic beauty, their three-dimensional (3D) geometries can be difficult to design (Figures 1(a) and 11).

In practice, furniture designers begin by sketching the overall shape of the furniture and delineating the visible 2D surfaces of the parts that are connected at each joint [Postell 2012]. These drawings directly visualize the surface aesthetics of the furniture, and commercial Computer-Aided Design (CAD) software like SketchUp, AutoCAD, and Rhino make it relatively easy to draw these 2D regions on the outer boundary of a solid furniture model. The challenge is to convert such 2D surface regions into a set of solid 3D parts that (1) conform to the specified 2D regions and (2) assemble without collision into the desired furniture shape. Often, even for seemingly simple 2D regions, the solid 3D parts that satisfy these two constraints are geometrically complex and therefore difficult for users to figure out and directly model in CAD software (e.g., in Figure 1(c), the green leg of the table has to slide in at a diagonal angle to the tabletop). Moreover, for the resulting assembly to function as furniture, it must not collapse under its own weight; the parts should remain stable and not move with respect to one another under external forces such as gravity.

We introduce an interactive tool for designing decorative joints such that the resulting parts can be assembled into structurally sound furniture. The input to our tool is a single solid 3D model of the

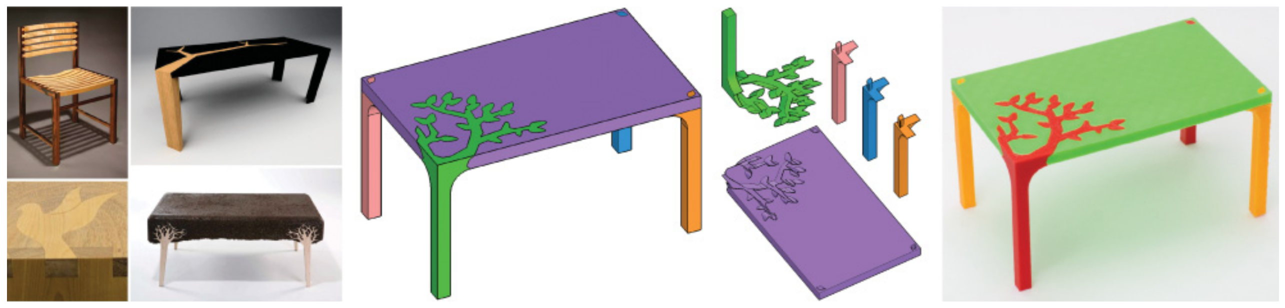

(a) Hand-crafted furniture with decorative joints
(b) Input: Surface 2D parts
(c) Output: Solid 3D parts
(d) Fabricated & assembled Leaf table

Fig. 1. Hand-crafted furniture often includes decorative joinery that is valued for its strength and beauty (a). With our interactive joinery design tool, users draw the visible 2D surface regions corresponding to parts on the surface of a solid 3D furniture model (b). Our tool then constructs solid 3D parts that conform to the user-specified regions and assemble into the solid model ((c) and (d)). (Please zoom in to see joint details in furniture models.)

furniture and a partition of its outer surface into groups of 2D surface regions that must belong to the same part. Our tool automatically partitions the input solid model into a set of solid 3D parts, one per group of 2D regions, or reports that such a partitioning violates one of the structural constraints—conformance to 2D regions, assemblability, or stability. If parts violate the assemblability constraint, then our tool visualizes the collision regions between them and suggests how to redesign the 2D surface regions so the resulting parts are collision free. Similarly, if any parts violate the stability constraint, our tool suggests where additional 2D regions should be drawn to better interlock the parts and improve stability. To analyze the stability of furniture parts, we devise a novel variational static analysis solver, improving on the shortcomings of the widely employed equilibrium method [Ochsendorf 2002] for our analysis tasks.

We demonstrate the generality of our tool by generating over 100 joints inspired by traditional woodworking joints [Rogowski 2002; Noll 2009], geometrically intricate Japanese joinery [Seike et al. 1986], and photographs of joints found on the Web. We also use our tool to design and fabricate nine complete furniture assemblies that are stable and connected using only intrinsic joints. These results, as well as evaluative feedback from amateur woodworkers, professional furniture designers, and novice first-time users, suggest that our tool gives users the creative freedom to prototype new forms of decorative joints and furniture, while focusing on their visual appearance rather than on constructing their 3D solid geometry.

2. RELATED WORK

Developing design tools for fabrication is an active area of research in the computer graphics and Human-computer interaction communities. We focus on the subset of techniques that are most related to our work on designing joints for fabricable furniture.

Furniture Design. Researchers have developed a number of approaches for designing fabricable furniture. The tools of Lau et al. [2011], Umetani et al. [2012], and Schulz et al. [2014] are particularly inspiring, as they let users customize the overall shape of the furniture, while automatically adjusting the alignments and connections between the parts to ensure a fabricable result. Both Umetani et al. and Schulz et al. also check the physical stability of the resulting furniture. However, these methods rely on a standard set of extrinsic fasteners such as nails and screws to join the parts together and do not account for the aesthetics of the joints. Others have developed tools for designing furniture composed of flat boards of material that are connected using intrinsic slotted joints [Oh et al. 2006; Saul et al. 2011; Hildebrand et al. 2012; Schwartzburg and Pauly 2013; Chen et al. 2013; McCrae et al. 2014; Cignoni et al. 2014]. While these techniques can generate a wide variety of furniture shapes, the geometry of the joints between the parts is relatively simple.

Our work is closest to that of Fu et al. [2015] and Yang et al. [2015]. Fu et al. recently introduced an automated technique for generating an interlocking set of intrinsic joints for an input 3D furniture model composed of rectangular boxes. Like earlier work on generating interlocking puzzle parts from input 3D models [Xin et al. 2011; Song et al. 2012], Fu et al. ensure that the resulting jointed parts are assemblable and interlocking, so every part is immobilized except for a single key that is free to move. Yang et al. use a material-aware database to replace parts of an input furniture model that simplify fabrication. They automatically connect adjacent wooden parts with mortise-and-tenon joints. These approaches use a template set of woodworking joints (e.g., dovetail, mortise and tenon, etc.) and do not allow users to specify the design of the joints. In contrast, our work lets users directly design the surface appearance of the joints and computes the underlying 3D joint geometry necessary to ensure assemblability of the parts.

Optimizing Physical Properties for Fabrication. A number of recent efforts have aimed at transforming 3D objects into physically fabricable objects while optimizing various physical properties. These techniques focus on reducing stress [Stava et al. 2012], preserving balance [Prévost et al. 2013], optimizing the moment of inertia [Bächer et al. 2014], maintaining stability of nailed together furniture [Umetani et al. 2012] or assemblies of rigid parts [Whiting et al. 2009, 2012; Shin et al. 2016], increasing aerodynamics [Umetani et al. 2014], and allowing furniture to stack [Li et al. 2012] or fold [Li et al. 2015]. Other techniques aim to reduce the amount of material required to fabricate objects [Saakes et al. 2013; Wang et al. 2013; Lu et al. 2014] and to split large models into smaller pieces for fabrication in small working volume printers [Luo et al. 2012; Chen et al. 2015; Yao et al. 2015] or due to other fabrication constraints [Hu et al. 2014; Herholz et al. 2015]. In contrast, we aim to apportion the internal volume of a model to match user's desired surface appearance, creating fabricable, assemblable parts that stand in a stable arrangement.

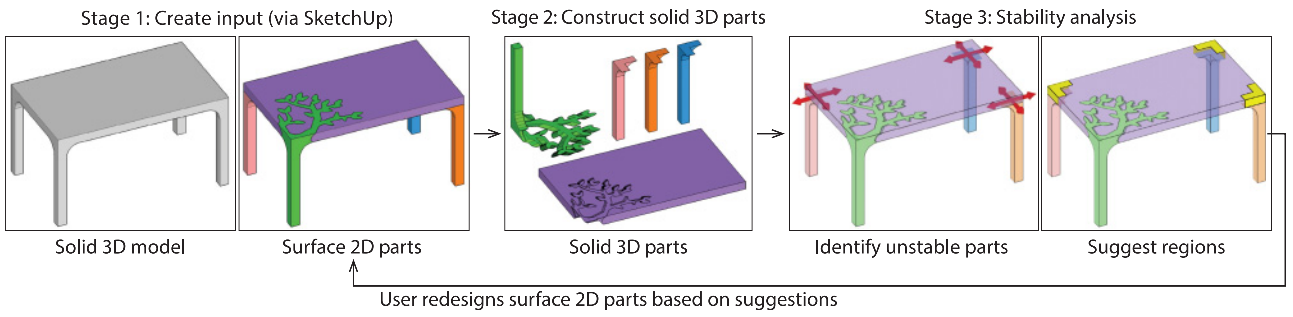

Stage 1: Create input (via SketchUp)
Stage 2: Construct solid 3D parts
Stage 3: Stability analysis
Solid 3D model
Surface 2D parts
Solid 3D parts
Identify unstable parts
Suggest regions
User redesigns surface 2D parts based on suggestions

Fig. 2. Three-stage workflow of our interactive joinery design tool. In stage 1, the user creates a solid 3D model and draws a partition of the model's exterior surface into groups of regions we call ***surface 2D parts*** (each color—red, orange, and green—indicates a different 2D part). In stage 2, our tool either converts each such surface 2D part into a corresponding ***solid 3D part*** that together assemble into the input solid model or reports that such a conversion is infeasible. If the infeasibility is due to collisions between the surface 2D parts, then our tool visualizes the collision regions and allows the user to quickly redesign the parts so they are collision free. In stage 3, our tool uses physical simulation to check the stability of the solid 3D parts (the red arrows indicate that three legs of the Leaf table are unstable and will slide with respect to the table top). If the parts are unstable, then our tool suggest where the user might add 2D regions to a surface part in order to improve stability (highlighted yellow regions). The user can then go back to stage 1 and redesign the surface 2D parts based on the suggestions. Figure 1 shows the redesigned Leaf table with additional joints connecting the legs to the table top.

3. OVERVIEW

As shown in Figure 2, our interactive joinery design tool supports an iterative three stage workflow. In stage 1, users create a watertight solid 3D model and draw the *surface 2D parts*—2D regions on the model surface that must belong to the same part—using SketchUp. In stage 2, our tool constructs a set of *solid 3D parts* that each conform to one of the surface 2D parts and that together assemble into the input solid model. In stage 3, our tool checks the stability of the assembled furniture under the force of gravity and perturbations to gravity. If the parts are unstable, then our tool suggests where the user might add more surface 2D part regions to improve stability. Users can then return to stage 1 and redesign the surface 2D parts to iteratively produce the final furniture model.

4. CONSTRUCT SOLID 3D PARTS

The first stage of our interactive joinery tool tries to construct a solid 3D part for each surface 2D part subject to the following conditions:

(1) ***Conformance to surface 2D part.*** The solid 3D part must be a single solid model (it cannot be disjoint), and its visible surface in the assembled model must exactly coincide with its associated surface 2D part.
(2) ***Assemblability.*** It must be possible to join the solid 3D parts to one another in some sequential order where each subsequent part is attached to the earlier parts using a single collision-free translational motion. We call this *sequential one-push assemblability (SOPA)*.

Prior work in assembly planning has shown that for an object composed of rigid parts, computing a collision-free sequence of motions that brings the parts into their assembled configuration is equivalent to starting with the object in its assembled configuration and computing a sequence of motions that separates (or disassembles) the parts [Wilson 1992; Toussaint 1985]. Therefore, our approach is to first identify all SOPA-compatible disassembly sequences for the surface 2D parts (Section 4.1). We then iterate through each of the SOPA sequences and apply a two-pass algorithm for converting the surface 2D parts into solid 3D parts (Section 4.2). We stop the iteration as soon as we obtain a set of solid 3D parts that satisfy the conformance and assemblability conditions. If all the SOPA sequences fail to meet these conditions, then our tool reports which parts in each sequence failed so the user can further modify those parts and try again (Section 4.3).

For clarity, we illustrate our construction algorithm with figures that show solid 3D models as 2D shapes and surface 2D parts as colored 1D edges (Figures 3–6).

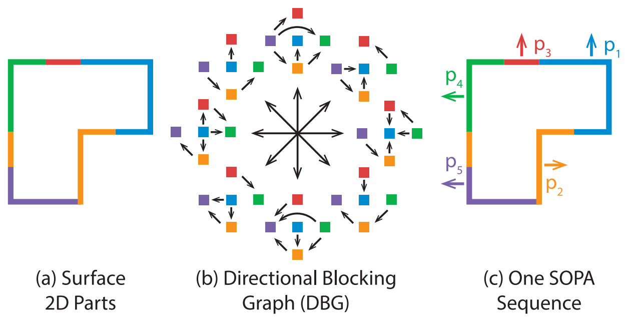


Fig. 3. Given a solid 3D model with a set of surface 2D parts (a), we compute a directional blocking graph (DBG) in a discrete set of directions (8 in this 2D example and 26 in our 3D implementation) (b). For each DBG direction, we compute a blocking graph. For example, in the upward direction graph at the top of the DBG, the arrow from orange to blue means that the orange part is blocked by the blue part in the upward direction. We use the DBG to compute SOPA-compatible disassembly sequences, one of which is shown in (c).

4.1 Find SOPA Compatible Disassembly Sequences

To identify SOPA-compatible disassembly sequences for the surface 2D parts, we first compute low-level blocking relationships between them. Specifically, we adapt the approach of Agrawala et al. [2003] and compute a *directional blocking graph (DBG)* that encodes the set of directions in which each surface 2D part p is blocked by another surface 2D part q from translating arbitrarily far away (Figures 3(a) and 3(b)). Like Agrawala et al., we build the DBG for a discrete set of translation directions—in our case, the 6 main axial directions, the 12 face diagonal directions, and the 8

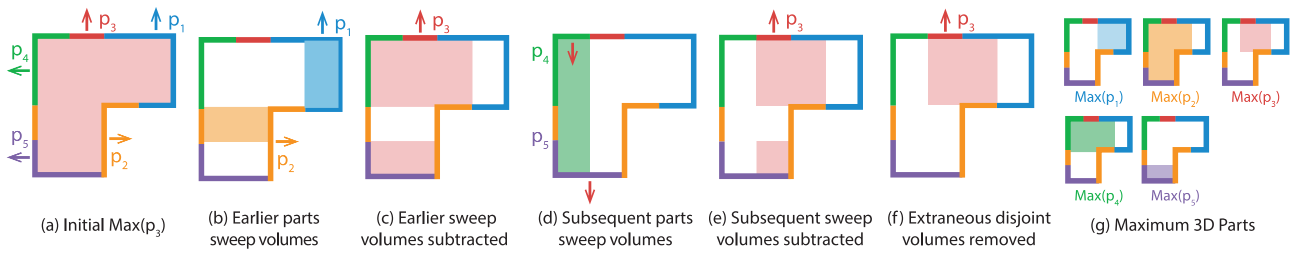

p3
p1
p4
p5
p2
(a) Initial Max(p3)
(b) Earlier parts sweep volumes
(c) Earlier sweep volumes subtracted
(d) Subsequent parts sweep volumes
(e) Subsequent sweep volumes subtracted
(f) Extraneous disjoint volumes removed
Max(p1)
Max(p2)
Max(p3)
Max(p4)
Max(p5)
(g) Maximum 3D Parts

Fig. 4. Given surface 2D parts and a SOPA sequence, we initialize $Max(p_3)$ to include the complete input solid model (a). We then compute the sweep volumes of earlier surface 2D parts in their respective disassembly directions (b) and subtract them from $Max(p_3)$ (c). We also compute the sweep volumes of subsequent surface 2D parts in the negative disassembly direction for p_3 (d) and subtract them from $Max(p_3)$ (e). Finally, we remove any extraneous volumes from $Max(p_3)$ (f). We similarly generate all of the other maximum 3D parts (g).

corner diagonal directions. To identify the blocking relationships between pairs of parts, we first sweep each surface 2D part p in each of the translation directions far enough to escape the bounding box of the input solid model and then check if the resulting sweep volumes intersect any other surface 2D part q. Such an intersection implies that q blocks p in the corresponding translation direction.

Since our tool focuses on generating SOPA sequences, our DBG only considers blocking with respect to single translational motions. Note however, that extension of the DBG to consider multi-step translational motions and rotational motions is possible [Guibas et al. 1995] but would significantly increase the size of the search space for disassembly sequences. We have found that nearly all joints that appear in woodworking books [Seike et al. 1986; Rogowski 2002; Noll 2009] can be assembled using the single translational motion allowed by SOPA, and we leave extensions to other models of assemblability to future work.

We use the DBG to find the set of SOPA-compatible disassembly sequences for the surface 2D parts as follows. For each permutation $p_i, \ldots, p_n$ of the surface 2D parts, we consider each part p_i in order and lookup the set of directions for which p_i is not blocked by the subsequent parts $p_{i+1}, \ldots, p_n$. If every part in the permutation has at least one such unblocked direction, then we add the permutation along with the unblocked directions for each part to our list of SOPA-compatible disassembly sequences. One such SOPA-compatible sequence is shown in Figure 3(c).

4.2 Two-Pass Construction Algorithm

For each SOPA-compatible disassembly sequence we obtain in the first step, we apply a two-pass algorithm that tries to construct a solid 3D part for each surface 2D part subject to the conformance and assemblability conditions. In the first pass, we compute the maximum portion of the input solid model volume that could belong to each surface 2D part p. We call this maximum volume the *maximum 3D part* and denote it $Max(p)$. In most cases, these maximum 3D parts intersect one another, so in the second pass we re-allocate the volume of the maximum 3D parts to remove such intersections, yielding a solid 3D part P for each surface 2D part p. All of our sweep, union, and subtraction operations are computed via constructive solid geometry (CSG) [Zhou et al. 2016].

4.2.1 *Pass 1: Construct Maximum 3D Parts.* Given a SOPA-compatible disassembly sequence of surface 2D parts $p_1, \ldots, p_n$, we compute a corresponding set of maximum 3D parts $Max(p_1), \ldots, Max(p_n)$, where each $Max(p_i)$ is the maximum portion of the solid model volume that can belong to p_i while remaining SOPA disassemblable from the other surface 2D parts (Figure 4). We initialize each $Max(p_i)$ to the complete input solid model volume and then subtract volume to ensure disassemblability. We subtract volume based on two observations.

First, we observe that for each maximum 3D part $Max(p_i)$ to be SOPA disassemblable from the other surface 2D parts, it cannot contain any of the volume swept out by the earlier parts $p_1, \ldots, p_{i-1}$ as they are translated in their respective disassembly directions (Figure 4(b)). Otherwise, some earlier part would collide with $Max(p_i)$ as it was disassembled. Therefore, we subtract all disassembly sweep volumes for earlier surface 2D parts from $Max(p_i)$ (Figure 4(c)).

Second, we observe that if surface 2D part p_i is SOPA disassemblable, we can translate it in its disassembly direction d_i away from the subsequent parts $p_{i+1}, \ldots, p_n$, without collision. Equivalently, we must be able to translate the subsequent parts in the opposite direction $-d_i$ without collision with p_i. Moreover, this translation of the subsequent parts sweeps out a volume that cannot be part of the maximum 3D part for p_i, and we therefore subtract it from $Max(p_i)$ (Figures 4(d) and 4(e)).

After the subtractions, a maximum 3D part may contain disjoint volumes (Figure 4(e)). We cannot add volume to connect these disjoint volumes since the maximum 3D part is, by construction, the largest allowable volume under the SOPA condition. We can, however, remove any extraneous disjoint volume from the maximum 3D part that does not contain a portion of the associated surface 2D part on its surface (Figure 4(f)).

Recall from Section 4 that the conformance condition requires that the visible surface of each solid 3D part P_i must exactly coincide with its associated surface 2D part p_i. Since we will construct the solid 3D part in the second pass of the algorithm as a subset of the maximum 3D part $Max(p_i)$, $Max(p_i)$ must also contain p_i on its visible surface. Our volume subtraction approach guarantees this containment condition. By construction, the sweep volumes we subtract from $Max(p_i)$ cannot contain p_i, because they are swept out based on SOPA disassembly directions. Therefore, p_i must be contained in $Max(p_i)$. Note, however, that $Max(p_i)$ may include other surface 2D parts p_j on its surface. For example, in Figure 4(f), $Max(p_3)$ includes portions of p_1 and p_2 on its surface.

If each resulting maximum 3D part consists of a single (non-disjoint) component, then we proceed to the second pass of the algorithm. Otherwise, some maximum 3D part $Max(p_i)$ remains disjoint and, since we removed extraneous volumes, a portion of the surface 2D part p_i lies on each disjoint piece. In this case, it is impossible to generate a non-disjoint solid 3D part P_i, so we skip the remainder of the two-pass construction and move on to the next SOPA sequence.

ALGORITHM 1: Construct Final Set of Solid 3D Parts $P_1, \ldots, P_n$ by Re-Allocating Volume of Maximum 3D Parts $Max(p_1), \ldots, Max(p_n)$.

```
1  Input: Surface 2D parts in SOPA disassembly order
       $p_1, \ldots, p_n$ and associated maximum 3D parts
       $Max(p_1), \ldots, Max(p_n)$.
2  // Initialize solid 3D parts
3  foreach part $Max(p_i)$ in SOPA order do
4      $P_i \leftarrow Max(p_i)$
5  end
6  foreach part $P_i$ in SOPA order do
7      foreach subsequent part $P_j$, $j > i$ in SOPA order do
8          $I_{ij} \leftarrow P_i \cap P_j$
9          if $I_{ij} \equiv \emptyset$ then continue;
10         // Check if we can remove $I_{ij}$ from $P_i$ and give it to $P_j$
11         if $I_{ij} \cap p_i \neq \emptyset$ then
           $T_i \leftarrow thicken(I_{ij} \cap p_i, P_j, d_i, I_{ij})$;
12         $B_{ij} \leftarrow block(P_i, I_{ij} - T_i)$
13         if $(P_i - I_{ij} - B_{ij} + T_i)$ is not disjoint and contains $p_i$
           then
14             $P_i \leftarrow (P_i - I_{ij} - B_{ij} + T_i)$
15             $P_j \leftarrow P_j - T_i$
16             continue
17         end
18         // Check if we can remove $I_{ij}$ from $P_j$ and give it to $P_i$
19         if $I_{ij} \cap p_j \neq \emptyset$ then
           $T_j \leftarrow thicken(I_{ij} \cap p_j, P_i, d_i, I_{ij})$;
20         $B_{ij} \leftarrow block(P_i, T_j)$
21         if $(P_j - I_{ij} + B_{ij} + T_j)$ is not disjoint and contains $p_j$
           then
22             $P_j \leftarrow (P_j - I_{ij} + B_{ij} + T_j)$
23             $P_i \leftarrow P_i - B_{ij} - T_j$
24             continue
25         end
26         // If neither removal possible terminate
27         terminate
28     end
29 end
```

4.2.2 *Pass 2: Re-Allocate Volume of Maximum 3D Parts.* At this stage, each surface 2D part p_i has anassociated maximum 3D part $Max(p_i)$. While the previous pass ensures that maximum 3D parts are each a single-component solid geometry whose surface includes all of the corresponding surface 2D part p_i, the maximum 3D parts often intersect each other (Figure 4(g)). In this pass, we remove volume from maximum 3D parts until they no longer intersect, resulting in our final set of solid 3D parts P_i that satisfy the conformance and assemblability conditions. Algorithm 1 provides pseudocode for our approach. For a gentler explanation, we initially describe a version of the algorithm where the intersection volume does not contain any surface 2D parts (e.g., we ignore pseudocode operations shaded light gray). The gentle approach is illustrated in Figure 5 with a simpler model and three surface 2D parts. An illustration of the entire run of the algorithm can be found in the supplemental materials.

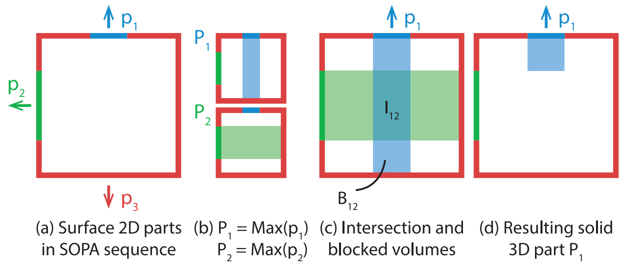

p1
p2
p3
P1
P2
I12
B12
(a) Surface 2D parts in SOPA sequence
(b) P1 = Max(p1) P2 = Max(p2)
(c) Intersection and blocked volumes
(d) Resulting solid 3D part P1

Fig. 5. Surface 2D parts p_1 and p_2 (a) and their corresponding maximum 3D parts (b). The maximum 3D parts are used as the initial solid 3D parts P_1 and P_2. To re-allocate the intersection volume I_{12} between P_1 and P_2 (c), we evaluate whether it can be given to p_2. Such a gift would block a portion of P_1 (labeled B_{12}) from being disassembled upwards with the rest of P_1. Because the solid 3D part P_1 still contains its surface 2D part p_1 even without the intersection and blocked volumes, the gift occurs (d).

We initially set the final solid 3D parts P_i to the corresponding maximum 3D parts (lines 3–5, Figure 5(b)). We then iterate over all pairs of parts P_i and P_j, $j > i$, to resolve their intersections.

If the pair of parts intersect, then we first check whether we can give the entire intersection volume I_{ij} to P_j (lines 10–17). The gift is allowed if removing I_{ij} from P_i still keeps P_i and its surface 2D part p_i connected. However, because P_i is disassembled before P_j, P_j will block any volume in P_i located "in the shadow" of the intersection volume from being disassembled (Figure 5(c), B_{12}). We call volume *blocked* if it cannot be translated in its disassembly direction, because the volume of another part is in the way. To ensure disassemblability, we remove both the intersection volume and the volume blocked by it (lines 12–13, Figure 5(c), B_{12}) when testing whether the intersection volume can be given to P_j. If P_i and its surface 2D part p_i remain connected after the removal, then we re-allocate the intersectionvolume and finish processing the pair of parts (lines 14–16).

If we cannot give the entire intersection volume to P_j, then we check whether we can give it all to P_i (lines 18–25). The operations are nearly symmetric; however, we do not need to consider blocked volume, because P_i is disassembled (removed) from the model before P_j is disassembled.

If neither re-allocation of the intersection volume keeps the solid 3D parts connected to their respective surface 2D parts, then we terminate and proceed to the next SOPA sequence (line 26).

The general case in which the intersection volume contains some or all of P_i's surface 2D part p_i is more complicated. Handling this case requires executing the operations of 1 shaded in light gray. In the example of Figure 6(a), giving all of intersection volume to P_2 would disconnect P_1 from p_1, guaranteeing failure. To account for this situation, when evaluating whether to give P_j the intersection volume (line 13), we *thicken* the portion of p_i that intersects the intersection volume and remove it from the gift (lines 11–15, Figure 6(d)–(f)). The algorithm performs a symmetric modification when considering giving the intersection volume to P_i (lines 19–22).

Thickening. Pseudocode for our *thicken* procedure is provided in Algorithm 2 and illustrated in Figure 6. The procedure takes as input a subset p_{sub} of a surface 2D part p to thicken, a solid 3D part Q for a different surface 2D part q, a disassembly direction d, and the intersection volume I. It outputs the volume T obtained by thickening every face of p_{sub} (lines 2–5). For each

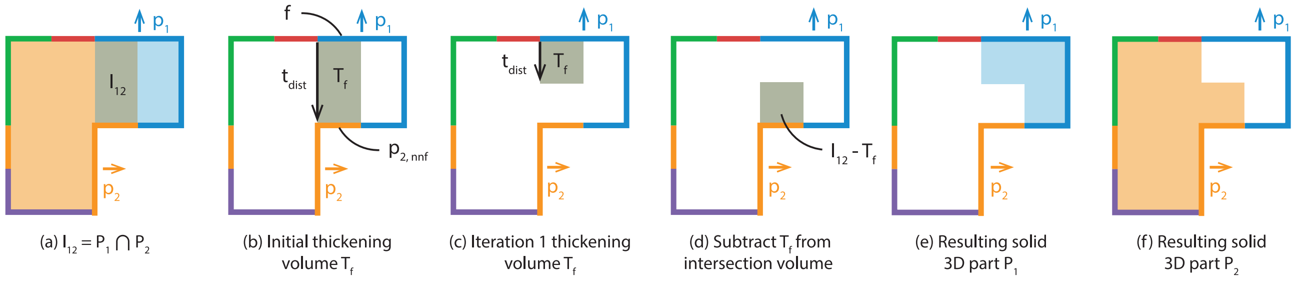

p1
I12
p2
(a) I12 = P1 ∩ P2
f
tdist
Tf
p2, nnf
(b) Initial thickening volume Tf
(c) Iteration 1 thickening volume Tf
I12 - Tf
(d) Subtract Tf from intersection volume
(e) Resulting solid 3D part P1
(f) Resulting solid 3D part P2

Fig. 6. The intersection I_{12} between solid 3D parts P_1 and P_2 touches a portion of surface 2D part p_1 (a). As a result, the intersection volume cannot be given to P_2 without first removing some volume near the intersected portion of p_1. The initial thickening volume T_f for the intersected portion f is obtained by extruding f parallel to p_1's disassembly direction until it touches a non-neighboring face of p_2 (b). This initial thickening volume is iteratively reduced by half (c) until it no longer intersects $p_{2,nnf}$. The intersection volume is reduced by subtracting T_f (d) and can now be given to P_2, resulting in new solid 3D parts ((e) and (f)).

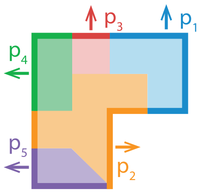
ALGORITHM 2: Thicken a Subset of a Surface 2D Part p by Taking Volume from Another Intermediate Solid 3D Part Q.

```
Input: A subset $p_{sub}$ of a surface 2D part $p$, an intermediate solid 3D
part $Q$ for a different surface 2D part $q$, a disassembly direction $d$,
and an intersection volume $I$
// Initialize thickening volume $T$
$T \leftarrow \emptyset$
// Try to thicken each face of $p_{sub}$ by taking volume from $Q$
foreach face $f$ in $p_{sub}$ do
    // Choose thickening direction $d_{thk}$
    if $f$ is inward facing w.r.t. $d$ then $d_{thk} \leftarrow d$;
    else if $f$ is outward facing w.r.t. $d$ then $d_{thk} \leftarrow -d$;
    else $d_{thk} \leftarrow$ interior facing normal of $f$;
    // Extrude $f$ in direction $d_{thk}$ by distance $t_{dist}$ to build
    thickening volume $T_f$ for face $f$
    $q_{nnf} \leftarrow$ faces of $q$ that do not neighbor $f$
    $t_{dist} \leftarrow$ minimum distance from point on $f$ to any face in $q_{nnf}$
    $T_f \leftarrow extrude(f, d_{thk}, t_{dist}) \cap I$
    while ($T_f \cap q_{nnf} \neq \emptyset$) or ($Q - T_f$ is disjoint) do
        $t_{dist} \leftarrow \frac{1}{2} t_{dist}$
        $T_f \leftarrow extrude(f, d_{thk}, t_{dist}) \cap I$
    end
    // Bisect $T_f$ to prevent intersection with neighboring faces of $q$
    $q_{nf} \leftarrow$ faces of $q$ that neighbor $f$
    foreach face $g$ in $q_{nf}$ do
        if $T_f \cap g \neq \emptyset$ then
            $G \leftarrow$ halfspace formed by plane through edge,
                between $f$ and $g$, bisecting angle between
                them, and containing $g$
            $T_f \leftarrow T_f - G$
        end
    end
    // Add $T_f$ to total thickening volume $T$
    $T \leftarrow T \cup T_f$
end
return $T$
```

such face f, we first determine the thickening direction d_{thk} by orienting the disassembly direction d towards the interior of the solid (lines 6–8). If, however, f is parallel to the disassembly direction, then we thicken along the inward normal of f (line 9).

To thicken f, we seek an extrusion volume T_f that (i) does not intersect faces of q, (ii) is contained within the intersection volume I, and (iii) keeps $Q - T_f$ a connected solid geometry. At first, we are only concerned with intersecting faces of q that are not neighboring f, because neighboring faces of q can intersect the extrusion volume no matter the extrusion distance (lines 10–18). So, we collect the non-neighboring faces of q in a set q_{nnf} (line 12). We then find the minimum distance t_{dist} from f to any face in q_{nnf} along the thickening direction d_{thk} (line 13). If no such face exists, then we set t_{dist} to "infinity"—any distance longer than the shape. To create the candidate thickened volume T_f satisfying (ii), we extrude f along direction d_{thk} and intersect it with I (line 14, Figure 6(b)). This candidate volume initially contacts a face in q_{nnf}, violating (i). To satisfy (iii) and (i) for non-neighboring faces, we repeatedly halve the extrusion distance and regenerate T_f (lines 15–18, Figure 6(c)). If no satisfactory extrusion distance can be found, then we terminate and proceed to the next SOPA sequence.

To ensure that (i) is satisfied even for neighboring faces q_{nf} of q (lines 20–21) we further adjust T_f in a process we call *bisection.* For each face g in q_{nf} that intersects T_f (lines 20–22), we find the plane through the shared edge of f and g that bisects the angle between them. We form the half-space G defined by the side of the plane containing g (lines 23–25). We remove all volume from T_f that lies in G, thereby guaranteeing that T_f contains no faces of g. An illustration of bisection can be found in the supplemental material, where we show the entire processing algorithm for our example shape. Finally, we join T_f with T and proceed to the next face in p_{sub} (line 30). When all faces of p_{sub} have been considered, the algorithm outputs T. The final set of solid 3D parts for our example can be seen in the inset.



4.3 Iterative Redesign for Infeasible Inputs

Since our tool does not put any limitations on the surface 2D parts given as input, users can specify an infeasible set of surface 2D parts for which it is impossible to produce solid 3D parts that satisfy the conformance and assemblability conditions. The most common source of such infeasibility is that the input surface 2D parts are not themselves SOPA disassemblable (Section 4.1). In such cases, our tool visualizes the disassembly sequence that produces the smallest total collision area between the surface 2D parts as a sequence of thumbnails showing each disassembly step (Figure 7). In Figure 7(a) when the green part is removed, the highlighted green

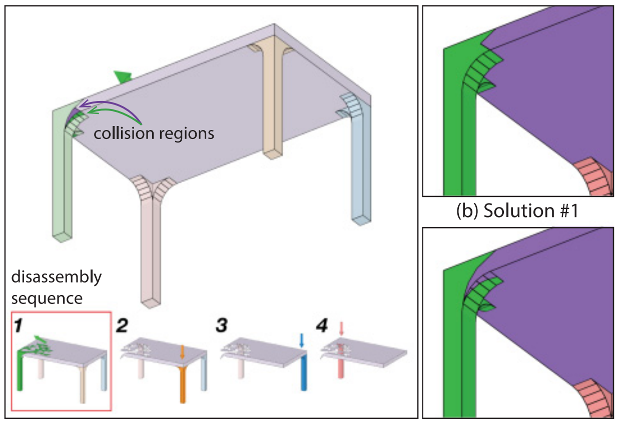
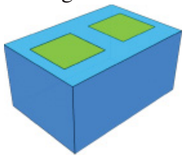


Fig. 7. Disassembly sequence minimizing collisions between surface 2D parts ((a), bottom). Each thumbnail highlights the next part to be removed from the assembly in the disassembly direction indicated by the arrow (e.g., the green part is removed in step 1). A red border indicates that the disassembly step contains collisions. Users can inspect the collision regions within each such step; here the user examines collision regions between the green and purple parts in step 1((a), top). To resolve the collision, the user can choose to give the highlighted collision regions to either of the two regions involved ((b) or (c)).

region collides with the highlighted purple region. Users can resolve such collisions by giving either highlighted collision region to the other part (Figure 7(b) or (c)). The resulting pair of surface 2D parts is guaranteed to be SOPA compatible. Alternatively, users can redesign the surface 2D parts in SketchUp to eliminate the collision.



It is also possible to specify surface 2D parts that are SOPA disassemblable but for which conforming and assemblable solid 3D parts provably do not exist. Consider a blue box with two disjoint green squares lying on its top face (inset). Although the blue and green surface 2D parts are SOPA disassemblable, there is no way to connect the green squares together and still disassemble them with a single translation. Our tool detects such infeasibility by checking the connectedness of maximum 3D parts (Section 4.2.1) and reports it to the user for redesign in SketchUp. In practice, we have found that this type of infeasibility is far less common than specifying non-SOPA-compatible 2D parts.

5. STABILITY ANALYSIS

Our construction algorithm generates a set of conforming, assemblable solid 3D parts. For this assembly to function as furniture, these parts must be stable under external forces. We call such an assembly *fully stable*. If an assembly is not fully stable, yet its parts nevertheless move together rigidly, then we call it *part-stable*. Finally, if an unstable assembly has parts that move with respect to one another, we call it *part-unstable*. Well-designed joints form interlocking geometries that rigidly lock an assembled structure together to provide part stability under gravity. If the assembled structure is part-stable, then the structure's shape and material densities determine whether its center of mass lies over the structure's support base. If so, then it is fully stable and thus in static equilibrium; if not, it is part-stable but not fully stable. User-designed surface 2D parts do not necessarily generate solid 3D parts that, once assembled, are part-stable, as they may lack constraining joints between contacting parts (Figure 10(a)). If we know which parts are unstable and *in what directions* they can move, then new joint geometries can then be designed to improve part stability.

To perform part-stability analysis for our furniture models, we require an analysis method that can (1) accurately predict whether an assembly is fully stable, part-stable, or part-unstable and (2) if the assembly is part-unstable, the method must report the locations and directions of sliding (translational) and hinging (rotational) accelerations that are generated by the part instabilities sothese failures can be fixed with modifications to the joints.

The equilibrium method (EM) [Huerta 2001; Ochsendorf 2002; Whiting et al. 2009, 2012; Shin et al. 2016] is the current state of the art for assembly stability analysis in graphics and architecture (see Shin et al. [2016] for a thorough survey). Unfortunately, EM is not suitable for furniture part-stability analysis, as it does not satisfy either of our requirements. EM can erroneously report assemblies as stable even when they are not.

EM is a constraint satisfaction method that formulates a set of physically based constraints on admissible contact forces that can equilibrate a part assembly. If any feasible set of equilibrating forces are found, then EM declares the assembly stable and returns a set of constraint-satisfying forces as a certificate. If not, then the constraint system is declared infeasible. In the latter case, EM declares the assembly unstable. However, as a constraint-satisfaction problem, EM does not have a model of the unstable assembly forces and accelerations and therefore cannot determine whether the assembly is part-unstable, nor identify the locations and directions of sliding and hinging failures between parts if the assembly is part-unstable.

Moreover, EM does not model sliding [Ochsendorf 2002] and, in practice, as we will show, incorrectly reports stability in the presence of sliding failures. Although the masonry analysis literature assumes sliding failures do not occur due to high friction between blocks, this assumption is not valid for furniture assemblies where inadequately jointed furniture parts often suffer from translational sliding instabilities (Figure 9).

The failure to satisfy our two requirements makes EM unsuitable for our part-stability analysis. Instead, we propose a *variational static analysis* method that addresses EM's failures. Our variational analysis correctly identifies instabilities when both hinging *and* sliding instabilities are present; it determines whether the assembly is part-unstable; and, if part instabilities are found, it provides the necessary per-part accelerations to report the locations and directions of sliding and hinging failures. This analysis lets us suggest additional joint regions to improve part stability.

5.1 Static Analysis for Rigid Assemblies

For stability analysis, we treat our parts P_b as rigid, each equipped with rotational $\boldsymbol{R}_b \in SO(3)$ and translational $\boldsymbol{t}_b \in \mathbb{R}^3$ degrees of freedom. Per body, we choose coordinates so $\boldsymbol{R}_b$ rotates from P_b's principal-axis-aligned body frame to world frame and $\boldsymbol{t}_b$ gives the location of P_b's center of mass in the world frame. The corresponding angular and linear accelerations $\dot{\boldsymbol{\omega}}_b$, $\ddot{\boldsymbol{t}}_b \in \mathbb{R}^3$ are concatenated into a single, systemwide acceleration vector,

$$\ddot{\mathsf{q}} = \left(\ddot{\boldsymbol{t}}_1^T,\ \dot{\boldsymbol{\omega}}_1^T, \ddot{\boldsymbol{t}}_2^T,\ \dot{\boldsymbol{\omega}}_2^T, \dots\right)^T. \tag{1}$$

The corresponding, systemwide, block-diagonal mass matrix is M (as in Kaufman et al. [2008]). Each material point $\boldsymbol{x} \in \mathbb{R}^3$ belonging to part P_b has corresponding constant body-frame coordinates $\hat{\boldsymbol{x}}$ so its world frame acceleration is $\ddot{\boldsymbol{x}}(\ddot{\mathsf{q}}) = \ddot{\boldsymbol{t}}_b - \boldsymbol{R}_b\hat{\boldsymbol{x}} \times \dot{\boldsymbol{\omega}}_b$.

Contacts. Contacts are between parts or between parts and a fixed boundary such as the ground. To simplify the following discussion, for each such contact $k \in \mathcal{C}$, the relative acceleration between two contacting points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j \in \mathbb{R}^3$ (at contact k) can be expressed via the linear map $\Gamma_k : \ddot{\mathsf{q}} \to \ddot{\boldsymbol{x}}_i - \ddot{\boldsymbol{x}}_j$. If $\boldsymbol{f} \in \mathbb{R}^3$ is a force applied to point $\boldsymbol{x}_i$, then and an equal but opposite force is applied to point $\boldsymbol{x}_j$, then $\Gamma_k^T \boldsymbol{f}$ is the resulting generalized force applied to the contacting system.

In turn, points in contact apply an equal and opposite force along their shared, unit-length normal $\boldsymbol{n}_k$. In global coordinates, this is equivalent to applying a force of magnitude $\alpha_k \in \mathbb{R}$ along a generalized normal,

$$\mathsf{n}_k = \Gamma_k^T \boldsymbol{n}_k, \tag{2}$$

to the system of parts. The subspace of generalized normal directions

$$\mathsf{N} = (\mathsf{n}_1 \ldots \mathsf{n}_{|\mathcal{C}|}) \tag{3}$$

then forms a basis for contact forces. Concatenating the corresponding force magnitudes in $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{|\mathcal{C}|})^T$, the total contact force applied in the system is then $\mathsf{N}\boldsymbol{\alpha}$.

Friction Forces. A friction force, applied at a contact point, lies in the tangent plane orthogonal to the contact normal. At each contact k, we sample an orthogonal pair of unit length vectors from the tangent plane. The 3×2 matrix composed columnwise of these samples is given by $\boldsymbol{T}_k$. A friction force, $\boldsymbol{f}_k \in \mathbb{R}^3$, applied at a contact k, lies in the span of $\boldsymbol{T}_k$ so $\boldsymbol{f}_k = \boldsymbol{T}_k\boldsymbol{\beta}_k$, where each $\boldsymbol{\beta}_k = (\beta_1, \beta_2)_k \in \mathbb{R}^2$ gives the frictional response coefficients at contact k.

The total friction force applied to the system at each contact k must be equal and opposite and is $\mathsf{f}_k = \Gamma_k^T \boldsymbol{T}_k\boldsymbol{\beta}_k$.

The generalized basis for a friction force at contact k is then

$$\mathsf{D}_k = \Gamma_k^T \boldsymbol{T}_k. \tag{4}$$

We build the corresponding subspace of generalized tangent directions,

$$\mathsf{D} = (\mathsf{D}_1 \ldots \mathsf{D}_{|\mathcal{C}|}). \tag{5}$$

and form the corresponding vector of frictional force coefficients as $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_m)^T = ((\beta_1, \beta_2)_1, \ldots, (\beta_1, \beta_2)_{|\mathcal{C}|})^T$, so the total friction force on the system is $\mathsf{D}\beta$.

With gravity and any additional forces acting on the system summed per system DoF and stored in the system force vector g, the total resultant force on the system is $\mathsf{r} = \mathsf{N}\boldsymbol{\alpha} + \mathsf{D}\boldsymbol{\beta} + \mathsf{g}$.

5.2 The Equilibrium Method

The EM seeks a set of feasible, equilibrating forces where the net torques and forces acting on all parts sum to zero. Concisely, it seeks a set of forces satisfying the static equilibrium condition

$$\mathsf{N}\boldsymbol{\alpha} + \mathsf{D}\boldsymbol{\beta} + \mathsf{g} = 0, \tag{6}$$

subject to the following two force feasibility conditions: first, that contact forces are compressive or, equivalently, that their magnitude along normals are non-negative so

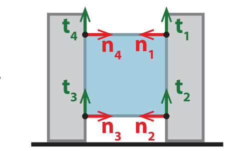
$$\boldsymbol{\alpha} \geq 0, \tag{7}$$

and, second, that the isotropic Coulomb constraint is enforced. This restricts, per contact, the magnitude of the friction force by the inequality

$$\| \boldsymbol{f}_k \| = \| \boldsymbol{T}_k\boldsymbol{\beta}_k \| = \| \boldsymbol{\beta}_k \| \leq \mu_k\alpha_k, \ \forall k \in \mathcal{C}, \tag{8}$$

where μ_k is the coefficient of friction and α_k is the normal (contact) force magnitude at k. This restricts the friction force to lie within a disk in the tangent plane with a radius of $\mu_k\alpha_k$. Applications [Huerta 2001; Whiting et al. 2009, 2012; Shin et al. 2016] often further linearize this inequality with a polyhedral approximation and apply a Quadratic Program (QP) or Linear Program objective to seek these forces. No matter the method of finding it, if a feasible equilibrating set of forces $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ satisfying Equations (6), (7), and (8) exists, then EM *claims* the structure as stable.

As Oschendorf [2002] notes, EM cannot accurately predict when parts will slide against one another. It is widely deployed for masonry analysis tasks under the assumption that strong friction forces between blocks prevent such sliding failures. This assumption is not valid for furniture assemblies, where inadequately jointed furniture parts often suffer from sliding instabilities (Figure 9).



To illustrate EM's inability to accurately predict sliding instabilities, consider the following didactic example. A single free block is placed in contact between two *fixed* vertical walls (inset), with gravity pointing downwards along the y-axis. Their contacting surfaces are geometrically flat, and there is no compression between the block and two walls. This system is symmetric, so we may equivalently consider it in the plane. With a linear force distribution on contacting faces, we sample just the contact points at the four corners of the block (now a square). Contact normals point *inward* from the wall towards the block, along the x-axis, while in 2D we require just a single tangent direction per contact, $\boldsymbol{t}_k$, pointing upwards along the y-axis.

The static equilibrium conditions for force and torque balance in this simple case are then, respectively,

$$\begin{aligned} &\sum_{k\in[1,4]} \left(\boldsymbol{n}_k\alpha_k + \boldsymbol{t}_k\beta_k\right) + \mathsf{g} = 0, \quad \text{and} \\ &\sum_{k\in[1,4]} \left(\hat{\boldsymbol{x}}_k \times \boldsymbol{n}_k\alpha_k + \hat{\boldsymbol{x}}_k \times \mathsf{t}_k\beta_k\right) = 0. \end{aligned} \tag{9}$$

Notice in Equation (9) that *any* set of feasible, uniform contact forces $\alpha_k = a \in \mathbb{R}$, $\forall k \in [1, 4]$, $a > 0$ entirely cancels out to a net torque and force equal to zero. This means that EM's feasibility constraints allow us to apply an arbitrarily large set of uniform contact forces. When substituted into Coulomb's constraint (8), arbitrarily large contact forces allow us to apply arbitrarily large friction forces. If we similarly make friction forces uniform and positive, that is, $\beta_k = b \in \mathbb{R}$, $\forall k \in [1, 4]$ with $b > 0$, then their net *torque* contributions cancel out, leaving us able to apply an arbitrarily large, feasible net upward force along the y-axis to balance the downward acceleration of gravity irrespective of how heavy we make the block.

No matter what coefficient of friction is used in this example, the correct solution for this configuration is that the block should always fall under gravity with no resistance. But the sliding instability is entirely missed here by EM. Instead, EM incorrectly declares the single sliding block configuration to be stable. Moreover, if we turn to an example where both the vertical walls and the block are *free*, then infinite forces are no longer permitted by EM. However, the EM code [Shin 2016] still continues to incorrectly claim equilibrium for this system of three free parts that have no joints to hold them together; see Figure 8, rightmost column. Here all parts are free and the assembly should be unstable with the middle block sliding.

		Infeasible	Feasible	3-legged π	H
	# blocks	36	36	4	3
stability at 0°	Analytic	U	S	S	U
	Experiment	-	S	S	U
	Variational	U	S	S	U
	EM	U	S	S	S
tilt test θ°	Analytic	N/A	8.2	-	N/A
	Experiment	N/A	4.7 ± 0.2	14.3 ± 0.2	N/A
	Variational	N/A	5.3	16.4	N/A
	EM	N/A	8.2	19.1	N/A

Fig. 8. Results of stability and tilt tests comparing analytic and experimental results with analyses from our variational static solver and the EM implementation analyzed by Shin et al. [2016]. To match with prior results [Shin et al. 2016], we follow the reported material density, 1.5g/cm^3, and friction angle, 43°. "S" and "U" indicate stable and unstable. For stability determination on a horizontal ground plane, our variational solver matches with the analytic and experimental solutions for all assemblies. EM differs by missing the sliding instability for the H assembly. For the tilt tests, our variational solver predicts a critical tilt angle between the angles experimentally determined and those predicted by EM. Note: Mesh colors indicate block number and do not depict stress.

More broadly, EM is missing adequate models of constraint and friction forces to properly predict and analyze sliding instabilities.

Independent of accuracy, when EM reports an instability, it does not and cannot report whether there are part instabilities or not. By construction, if an assembly is determined unstable, EM is simply an oracle and does not provide any additional information on which parts of the assembly are unstable or in which direction parts will move relative to each other. The solid 3D parts generated by user-designed surface 2D parts can suffer from multiple stability failures (Figures 9 and 10). Without information on which parts are unstable, we cannot offer meaningful suggestions to users on how to improve part stabilities for furniture assembly.

Prior work in masonry analysis has similarly needed to confront EM's inability to analyze instability. In order to optimize structures to satisfy EM constraints, Whiting et al. [2009] remove EM's non-negativity constraint in Equation (7). This temporarily allows for inadmissible, tension forces along normal directions at contacts. They then minimize the squared norm of these tensile forces to find a design that best satisfies EM's material compression constraint. While this process seeks a new design satisfying EM feasibility, the defined metric does identifies neither the locations nor the directions of sliding and hinging failures between parts.

5.3 Variational Static Analysis

To build our static analysis solver, we begin with the observation that EM makes well-founded physical assumptions, yet it misses sliding instabilities by finding physically invalid or *unrealizable* forces—something must be missing. As illustrated in the single block with two fixed walls example above, EM is underconstrained and can greedily find physically infeasible contact forces leading to incorrect predictions. To prevent such unrealizable forces, our solver invokes a pair of variational principles from classical mechanics, Gauss's Least Constraint [Moreau 1966] and the Principle of Maximal Dissipation [Goyal et al. 1991]. These two principles restrict the static analysis model to a predictive set of feasible frictional-contact forces.

We start with Gauss's Least Constraint which, geometrically interpreted, tells us that realizable contact-constraint forces should be *minimal*. We then first require a constraint. We start with contact constraints: *points in contact should not interpenetrate*. To prevent interpenetration at a contact k, the relative acceleration between the two contacting points along their normal $\boldsymbol{n}_k$, given by $\boldsymbol{n}_k^T \Gamma_k \ddot{\mathsf{q}}$, must be non-negative. This is equivalent to enforcing the contact constraint $\boldsymbol{n}_k^T \Gamma_k \ddot{\mathsf{q}} \geq 0$. The equivalent non-penetration inequality constraint, for all points of contact simultaneously, is then

$$\mathsf{N}^T \ddot{\mathsf{q}} \geq 0. \tag{10}$$

Gauss's Least Constraint for contacts requires that realizable constraint forces $\boldsymbol{\alpha}^*$ be minimal so

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \ \tfrac{1}{2} \|\mathsf{N}\boldsymbol{\alpha}\|^2_{\mathsf{M}^{-1}} \ \text{ s.t. } \ \mathsf{N}^T \ddot{\mathsf{q}} \geq 0. \tag{11}$$

Maximal Dissipation Principle. Letting $\boldsymbol{f}_k$ denote a frictional force applied at a contact point, $\boldsymbol{x}_k$, the *Maximal Dissipation Principle* requires friction to maximize the rate of negative work done at the contact. Adopting an instantaneous view, we maximize $-\boldsymbol{f}_k^T \ddot{\boldsymbol{x}}$. Combined with the Coulomb friction constraint, maximal dissipation provides an acceleration-level interpretation of the familiar Coulomb friction law.

We then find friction forces for a system in contact, with no velocity, by enforcing Maximal Dissipation simultaneously at *all* contact points to obtain an instantaneous, global minimization,

$$\max_{\boldsymbol{f}_k} \sum_{k \in \mathcal{C}} \left(-\boldsymbol{f}_k^T \ddot{\boldsymbol{x}}_k\right) = \min_{\boldsymbol{f}_k} \left[\sum_{k \in \mathcal{C}} \boldsymbol{f}_k^T \Gamma_k \right] \ddot{\mathsf{q}} = \min_{\mathsf{f}} \mathsf{f}^T \ddot{\mathsf{q}}. \tag{12}$$

Equilibrium Testing. Substituting, $\ddot{\mathsf{q}} = \mathsf{M}^{-1}(\mathsf{N}\boldsymbol{\alpha} + \mathsf{D}\boldsymbol{\beta} + \mathsf{g})$, Gauss's Least Constraint is equivalently the convex QP,

$$\min_{\boldsymbol{\alpha}} \tfrac{1}{2} \boldsymbol{\alpha}^T \mathsf{N}^T \mathsf{M}^{-1} \mathsf{N} \boldsymbol{\alpha} \ \ s.t. \ \ \mathsf{N}^T \mathsf{M}^{-1} (\mathsf{N}\boldsymbol{\alpha} + \mathsf{D}\boldsymbol{\beta} + \mathsf{g}) \geq 0. \tag{13}$$

Similarly, substituting acceleration into Equation (12), we must simultaneously satisfy Maximal Dissipation as a Quadratically Constrained QP,

$$\begin{aligned} \min_{\boldsymbol{\beta}} \ & \boldsymbol{\beta}^T \mathsf{D}^T \mathsf{M}^{-1} \mathsf{D} \boldsymbol{\beta} + \boldsymbol{\beta}^T \mathsf{D}^T \mathsf{M}^{-1} (\mathsf{N}\boldsymbol{\alpha} + \mathsf{g}) \\ s.t. \ & \| \beta_k \| \leq \mu_k \alpha_k, \quad \forall k \in \mathcal{C}. \end{aligned} \tag{14}$$

As the global system's forces must satisfy both Equations (13) and (14) at the same time, we solve for unknown contact and friction forces with a Staggered Projections [Kaufman et al. 2008] sequence. We initialize Equation (13) with a starting $\boldsymbol{\beta}$ and then iterate between the solutions of the two minimizations to convergence with a relative tolerance of 10^{-4}.

The solution is the total force on the system $\mathsf{r}^* = \mathsf{N}\alpha^* + \mathsf{D}\beta^* + \mathsf{g}$. We have force balance and thus equilibrium if $\|\mathsf{r}^*\| = 0$. Force balance guarantees that all system parts are at relative rest and that the system is not moving. If we are not at equilibrium, then our analysis models the total out of balance forces $\mathsf{r}^* > 0$ on the assembly. This allows us to compute the relative accelerations between parts in order to determine which parts are unstable and in what direction they are moving. We compute the relative accelerations between part contact pairs i and j of each contact k as

$$\boldsymbol{a}_k = \ddot{\boldsymbol{x}}_i - \ddot{\boldsymbol{x}}_j = \Gamma_k \mathsf{M}^{-1} \mathsf{r}^* \in \mathbb{R}^3 \tag{15}$$

and extract the individual rigid part accelerations from the system-wide acceleration vector $\ddot{\mathsf{q}} = \mathsf{M}^{-1}\mathsf{r}^*$.

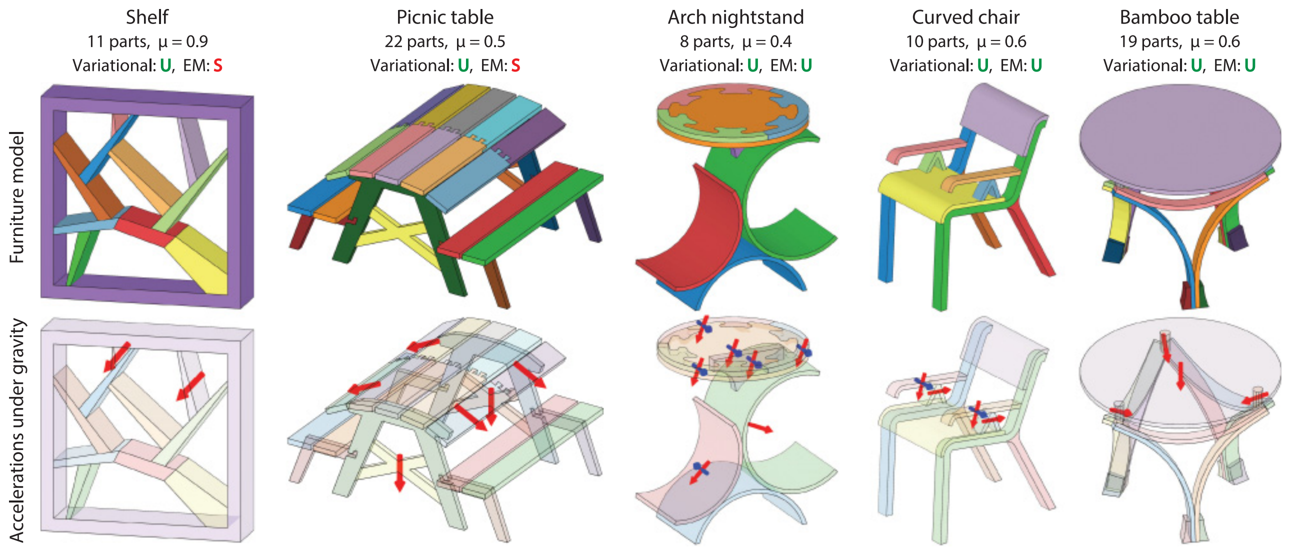
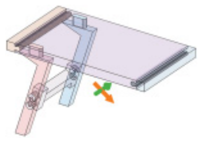


Fig. 9. Results of stability analyses from our variational solver and the EM on five inadequately jointed furniture models under gravity. "S" and "U" indicate stable and unstable assembly results per method, respectively. All examples should be determined unstable in the analysis. Note that EM incorrectly determines the two leftmost assemblies stable. In addition to analyzing whether the furniture assembly is stable, for furniture design we require an analysis of where and how parts are sliding or hinging against each other if the system is found part-unstable. This is information that EM cannot provide. Our variational solver identifies the location and direction of sliding and hinging part instabilities between furniture parts (bottom row). This allows our tool to suggest additional joint regions to improve part stabilities. For the bamboo table, the six small braces near the ground are modeled with density $8.0\text{g}/\text{cm}^3$ and all other parts are modeled with density $0.3\text{g}/\text{cm}^3$, while the four remaining models use a density of $0.5\text{g}/\text{cm}^3$. The corresponding friction coefficients μ used for each model are summarized above. Red and blue arrows, if present, give the local linear and angular acceleration of unstable parts (i.e., the locations and directions of the sliding and hinging instabilities, respectively).

After each execution of stability analysis, our system reports one of three possible stability states:

Fully stable. If all relative accelerations, $\boldsymbol{a}_k$, and individual accelerations, $\ddot{\boldsymbol{x}}_i, \ddot{\boldsymbol{x}}_j$, are zero for all contacts, then we have a *fully stable* assembly.



Part-stable (but not fully stable). If all relative accelerations, $\boldsymbol{a}_k$, are zero *but* there are individual non-zero accelerations, $\ddot{\boldsymbol{x}}_i, \ddot{\boldsymbol{x}}_j$, then the system is part-stable but not at equilibrium. All joints are holding parts together *but* the entire furniture assembly is out of balance and needs reshaping. The parts of the inset table are interlocked with joints, yet, because it is missing two legs, the table is unbalanced and will fall as a rigid unit. Our interface shows the *global* linear and angular accelerations with orange and green arrows, respectively.

Part-unstable. If there exist nonzero relative accelerations, $\boldsymbol{a}_k \neq 0$, then parts in contact are moving relative to one another, and the system is part-unstable. The relative accelerations allow us to determine which parts of the assembly are unstable, as well as the locations and directions of their relative sliding and hinging instabilities. We then offer suggestions, detailed in Section 5.5, on where additional joint regions may be added to improve part stability.

5.4 Evaluation

To evaluate our variational static solver, we validate it with a suite of benchmark examples (Figure 8), a collection of inadequately jointed furniture models with a range of sliding and hinging instabilities (Figure 9), and all of our fabricated furniture examples (Figure 1 and 12).

In Figure 8, we compare the analysis of our variational static solver in determining stability to the EM implementation of Shin et al. [2016]. We compute the stability of each assembly for the horizontal ground plane at 0° (top four rows). If an assembly is stable at horizontal, then we additionally determine the critical ground-plane tilt angle where the assembly becomes unstable (bottom four rows). Under the assumption of no sliding failures, *circular masonry arches* have analytic solutions for their stability on horizontal and tilted grounds given by the thickness-to-radius ratio t/r; arches are increasingly stable as t/r grows [Ochsendorf 2002]. Here we consider an *infeasible* arch with $t/r = 0.08$ and a *feasible* arch with $t/r = 0.15$. Our variational solver correctly determines the infeasible arch unstable and the feasible arch stable. As we tilt the ground plane, the variational solver predicts the critical tilt angle beyond which the feasible arch is unstable at 5.3°. This angle is between the experimentally determined angle of 4.7° and the 8.2° determined by EM [Shin et al. 2016]. While the analytic solution for the critical tilt angle is given as 8.2°, both the analytic and EM solutions neglect treatment of finite friction with sliding, opening an interesting question on the degree to which results differ due to treatment of friction and sliding.

The *three-legged* π assembly evaluates equilibrium analysis with static indeterminacy. Our variational solver determines the π assembly stable on a horizontal ground plane and computes the critical ground plane tilt angle to be 16.4°. As in the masonry arch examples, this angle is again between the experimentally determined angle of 14.3° and the 19.1° determined by EM [Shin et al. 2016].

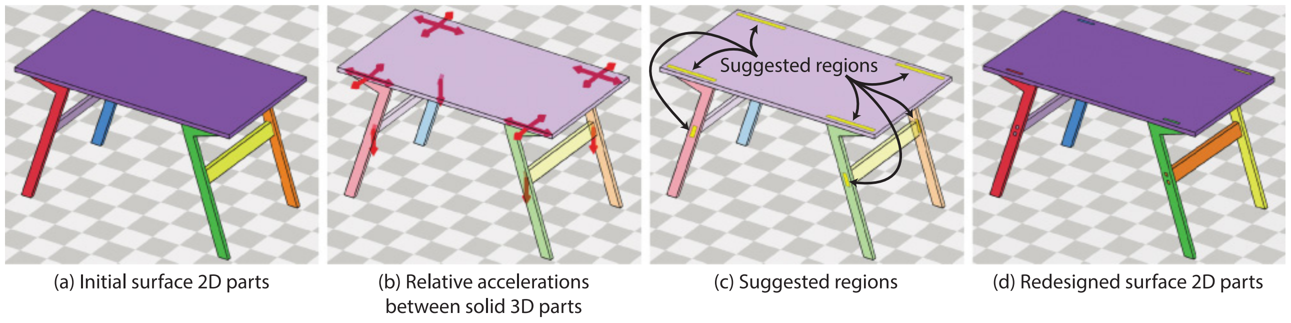

Suggested regions
(a) Initial surface 2D parts
(b) Relative accelerations between solid 3D parts
(c) Suggested regions
(d) Redesigned surface 2D parts

Fig. 10. An initial set of surface 2D parts (a) generates unstable solid 3D parts (b). Our tool reports the part instabilities and presents the relative linear accelerations between unstable parts as red arrows. Our tool highlights suggested regions in yellow on the model surface where additional 2D regions can be drawn to improve part stability (c). Re-design with additional 2D regions leads to a stable 3D part assembly (d).

We also introduce the *H* assembly example to evaluate sliding instability. For this example, we place a horizontal block between two vertical free-standing blocks *without* connecting joints. All parts are *free* and without connections so the middle block falls towards the ground. Our variational solver captures the expected sliding instability with a predicted downward acceleration while the EM code incorrectly declares stability for the unsupported middle block.

In addition to these benchmark examples, we compare our variational solver with a state-of-the-art EM code [Shin et al. 2016] on five inadequately jointed furniture assembly models with complex geometries and non-trivial instability modes (Figure 9). Gravity points downwards along the y-axis for all examples, and the same material densities and friction coefficients are used for both methods, as detailed in the figure. Red and blue arrows indicate translational and rotational accelerations of unstable parts respectively. Both our solver and the EM code agree and report instability for three of five models. However, only our solver correctly captures the sliding instabilities in the *shelf* and the *picnic table* examples. The EM code incorrectly declares these examples stable. The pink and light purple parts at the top of the shelf slide due to the significant slope of the contacting blue and light green parts below them. The cross bars of the picnic table fall freely due to lack of support, while the dovetail-jointed outer slabs of the table top slide due to insufficient friction.

Unlike EM, our solver not only determines if an assembly is stable but also finds the parts that are unstable and presents the location and direction of each sliding and hinging instability (Figure 9, bottom row). For the *arch nightstand*, our solver determines that the red and green arches are unstable, as well as the purple support part just below the table top and the four lap-jointed table-top parts that rest on it. The *curved chair*'s triangular arm-rest supports do not balance the arm rests. Our solver correctly captures the hinging and sliding failures. As the arm rests pivot and fall, the triangular supports slide along the chair seat. This is also captured by our solver. Note that triangular supports do not exhibit hinging failure due to their triangular shape. In the *bamboo table* example, the six small braces near the ground have a higher density of metal (8.0g/cm^3), while the nine arches, the table top, and three supporting pegs underneath the table top have the (significantly lighter) density of bamboo (0.3g/cm^3). Our solver reports that the nine contacting arches are stable, due to the heavy braces near the ground. However, the three pegs supporting the table top slide toward the center. As a result, the table top falls downward. Finally, all of our *fabricated furniture examples* (Figure 1 and 12) are determined stable by our variational solver and then validated as stable on assembly of the final fabricated parts.

5.5 Visualizing Instability

In our joinery design interface, we analyze the part stability of the solid 3D furniture parts generated by our construction algorithm, using our variational solver 5 times. First, we run our solver with gravity alone (pointing downwards along the y-axis). We then run our solver four additional times with perturbational forces added to gravity along the positive and negative x- and z-axes covering all forces in the downward hemisphere of directions, including horizontal forces. These perturbations allow us to suggest new regions for unattached parts such as table tops (Figure 10) or table legs that should be attached but would be found part-stable under perfect conditions. While we focus on stability under gravity in this work, our solver supports analysis with arbitrary external forces; our interface can be extended to model additional specified loads and forces, such as the weight of a load on a seat or shelf.

If part instability is reported, then our tool guides the user through all pairs of contacting parts that are not part-stable, visualizing all predicted motions with arrows. Relative linear and angular accelerations $\boldsymbol{a}_k$ computed by Equation (15) are shown with red and blue arrows, respectively (Figure 10(b)). Global linear and angular accelerations are shown with orange and green arrows, respectively (inset figure near the end of Section 5.3). For every unstable pair of parts, we highlight suggested faces on the model surface where additional 2D regions can be drawn to oppose the reported part instabilities (Figure 10(c)). To find such faces, we observe that each contact patch between a pair of unstable parts represents a site on which joints can potentially be added to prevent relative motion. We assume that added joints must lie inside the extrusion volume of the contact patch in either of its normal directions; this is standard practice in the woodworking literature for joint placement. So, for each contact patch, we find the extrusion volume that lies in either part. The suggested faces are the visible faces of this volume that lie on one of the two parts (Figures 10(c) and (d))—the part that would be less covered or "erased" by the suggestion.

6. RESULTS

We have used our interactive joinery design tool to generate over 100 different decorative joints (Figure 11 shows representative examples, and supplemental materials contain all of the joints we

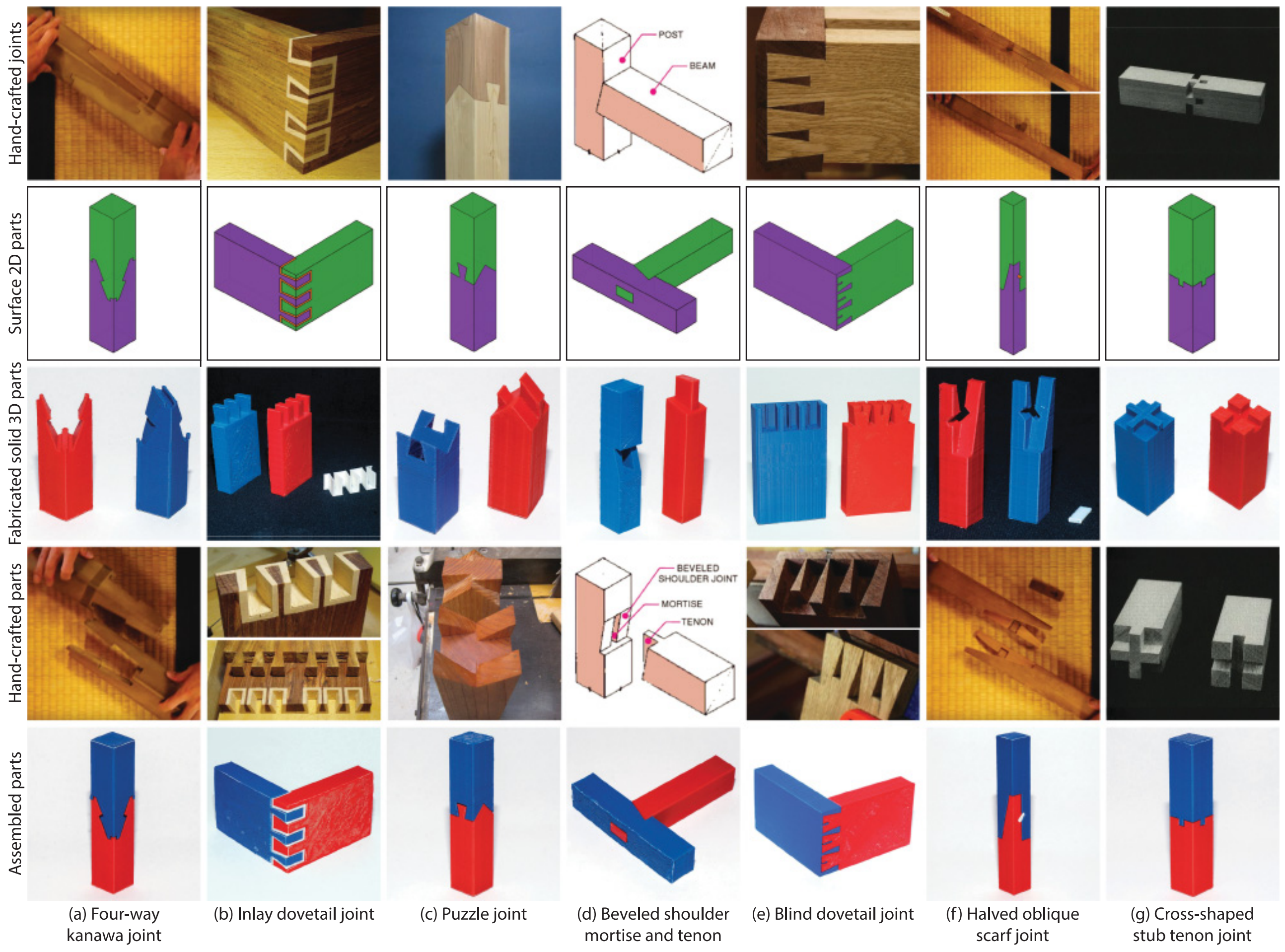


Fig. 11. A representative sample of joints designed and fabricated using our interactive tool. Some are based on traditional woodworking joints ((b), (e), and (g)) and others on Japanese joinery ((a), (c), (d), and (f)). For all of these designs, we drew the surface 2D parts using a photo of the assembled hand-crafted joint (top row) as a guide. The resulting solid 3D parts generated by our tool (third row) is very similar to the hand-crafted part geometry (fourth row), suggesting that the internal geometry of such joints is inherently complex. Please zoom in to see the geometric details. Supplemental materials contain over 100 decorative joints we created using our tool.

designed) as well as nine complete furniture assemblies containing multiple parts and joints (Figures 1 and 12). We have also evaluated our system with amateur woodworkers, professional furniture designers, and novice first-time users.

To create the joints (Figure 11 and supplemental materials), we drew the surface 2D parts based on photographs we found in traditional woodworking guides [Rogowski 2002; Noll 2009] (e.g., inlay dovetail, blind dovetail), Japanese joinery books [Seike et al. 1986] (e.g., four-way kanawa, puzzle joint), and examples from the Web (e.g., arrow in supplemental materials). We replicated 52 of the 62 joints described at two popular woodworking websites,[1,2] as well as 38 of the 40 Japanese joints of Seike [1986]. All of the remaining joints (10 from websites and 2 from Japanese joinery book) require rotations or multi-step translational disassembly and are not SOPA compatible. The photographs often served as inspiration, and we adapted them to test different appearances. For example, we varied the thickness of the inlay in the inlay dovetail or modified the shapes of the arrows. The 3D geometry generated by our tool was often surprisingly complex despite relatively simple surface 2D parts. Yet, for the joints inspired by traditional woodworking and Japanese joinery, our tool produced similar, if not identical, 3D geometry, suggesting that the complexity is inherent in certain joint designs.

We designed the complete furniture models (Figures 1 and 12) to contain a variety of different joint types. For example:

- **Mortise-and-tenon joints:** joining purple Duffy table top to legs; Duffy table legs to yellow and light purple leg braces; and sides of Nightstand to green top.
- **Straight and curved finger joints:** joining sides of Nightstand to back; green side of Bench to top; and red and green legs of Arch chair.
- **Lap and half-lap joints:** joining light purple top seat back to side frames of Branca chair; dark purple and orange seat parts of Branca chair to sides.

[1]http://wwideas.com/2015/11/the-most-impressive-wood-joints/.
[2]http://www.flexiblestream.org/Digital-Wood-Joints-001.php.

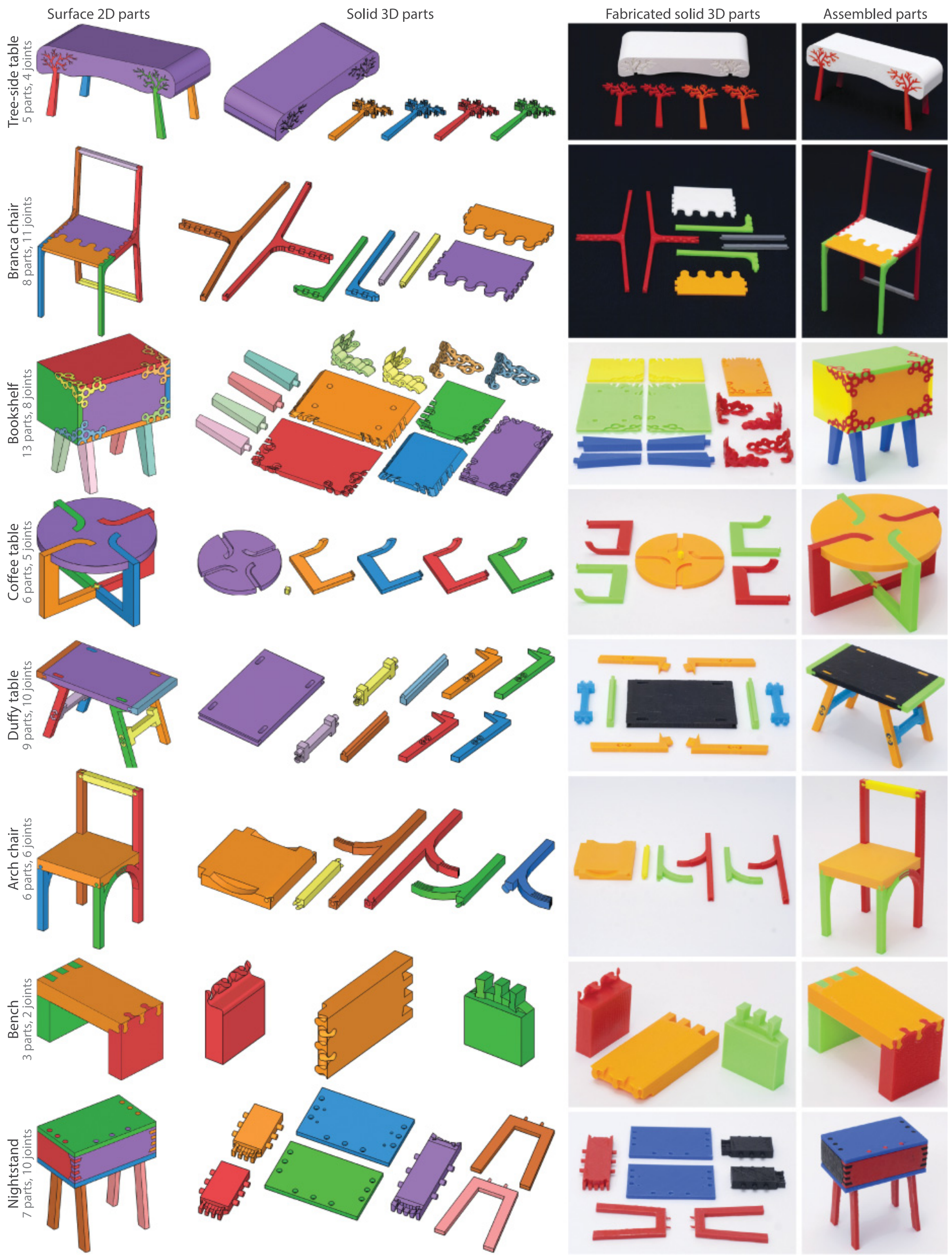


Fig. 12. Complete furniture assemblies we designed using our interactive tool. Please zoom in to see the geometry of the joints.

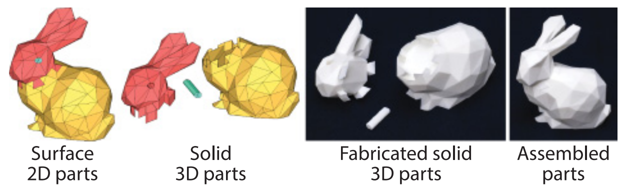


Fig. 13. Our tool can generate decorative joints for free-form geometric surfaces like the Stanford bunny.

Tongue-and-groove joints: joining left/right ends of Duffy table to purple table top.

Other custom joint designs: tree-shaped legs on Leaf table and Tree side table; multi-circular joints at corners of Bookshelf; curved variants of puzzle joint between red side and top of Bench and between yellow top of seat back and sides frames of Arch chair; hooked joints on Coffee table top.

We adapted several of these designs from Web photographs of finished hand-crafted furniture.

As we iterated the joint designs, we regularly generated SOPA-incompatible surface 2D parts and used the collision feedback from our interface (Section 4.3) to modify the parts—often directly using one of the solutions provided by our tool. The furniture examples also contain enough parts that our stability analysis often identified places where we forgot to add joints to prevent sliding between parts. In such cases, we added joints in the regions suggested by our tool to eliminate such part instabilities. Figure 10 shows an early design for the Duffy table in which we identified instabilities between the table top and legs as well as the legs and the leg braces. Despite the complexity of some of the resulting part geometries, all of the resulting furniture models are assemblable and stable.

Our tool can also generate decorative joints for more free-form geometric surfaces like the Stanford bunny (Figure 13). Together these examples demonstrate that our tool is flexible enough to explore a wide space of decorative joint designs.

Our implementation takes between 3 and 15min to construct the solid 3D parts for the joint models. Most of the furniture assemblies take between 30min (e.g., the 8-part Branca chair) and 60min (e.g., the 7-part Nightstand). Two assemblies take much longer (about 24h for the 13-part Bookshelf and the 3-part Bench). In all cases, computing a SOPA-compatible assembly sequence for the surface 2D parts takes 1–4min. If there are collisions, then they are presented to the user at this point. The remainder of the time is spent constructing the solid 3D parts. Our running time is dominated by the performance of mesh-based CSG. We use `libigl` [Zhou et al. 2016], a performant and freely available CSG implementation. The running time depends primarily on the tessellation of the surface 2D parts—when the edges of these parts are curved, they require finer tessellation, which in turn requires more work when applying the sweep, union, and subtraction operations of our algorithm. The Bench and the Bookshelf assemblies contain intricately curved parts which require very high tessellation. Since our algorithm considers pairwise interactions between parts, its running time depends quadratically on the number of parts.

3D Fabrication. We used a PrintrBot and MakerBot with PLA or ABS plastic material to fabricate physical parts for many of our joints and furniture assemblies. Due to the resolution of these printers and expansion of 3D printed parts, we have observed that the fabricated parts often do not fit each other when their geometries are directly supplied to the 3D printer. To ensure they fit and can be assembled, we assume uniform material expansion and offset the 3D part geometries before supplying them to the 3D printer. We translate each vertex of a part by ϵ in the average of the normal directions of its incident faces. For both the PrintrBot and MakerBot, we find that $\epsilon = 0.25$mm ensures that the fabricated parts can be assembled and that their fit is tight. As a proof of concept, we commissioned a woodworker to fabricate the Branca chair at full size using the solid 3D parts generated by our tool (Figure 14).

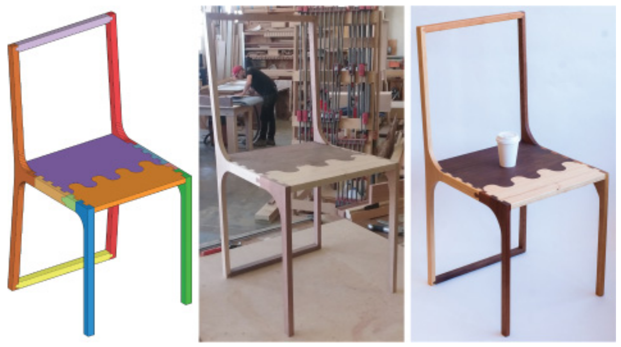


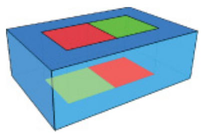
Fig. 14. We commissioned a woodworker to fabricate a version of the Branca chair at full size using the solid 3D parts generated by our tool. (Coffee cup added for scale.).

User Feedback. We have shown our tool and results to seven hobbyist (amateur) woodworkers and two professional furniture designers from local studios. All of them thought that our tools made it easier to design and generate novel, decorative furniture joints. They especially appreciated the creative freedom with which they could focus on sketching the surface aesthetics of joints while letting our tool determine their complex internal geometries. A hobbyist woodworker said *"I avoided complex 3D geometries in my joint designs"* in the past and explained that with our tool he could focus on designing more decorative joints. They liked that our tool gave suggestions for fixing SOPA incompatibilities and for correcting unstable part configurations. One of the professional furniture designers noted, *"This is a super cool system! I could see people creating some amazing pieces using this technique. It would be really powerful in the hands of an artist."*

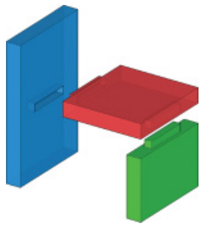
We also asked three novice designers to generate joints for the Bench furniture model using our tool. Some of them drew surface 2D parts based on inspirational images of joints found online while others incorporated free-form shapes, such as letter-forms and icons. They generally did not think about constraints on assemblability of parts and were focused on surface appearance. In a few cases, our system could not find a SOPA sequence for the surface 2D parts and instead suggested how the parts might be redesigned to produce a SOPA sequence. The users went back and iteratively redesigned the joints based on the suggestions; most often using one of the automatically suggested redesigns. In a few cases, the redesign involved going back into SketchUp to re-sketch the surface 2D parts. While using our tool, the novice designers did raise concerns over the rate at which our tool constructed the solid 3D parts and believed improving the construction time would further facilitate the joinery design process. However, they still believed our tool was much easier to use than directly creating the 3D geometry in SketchUp and were surprised at the complexity of 3D geometry that was necessary to produce conforming, assemblable joints.

Limitations. While our tool can successfully generate solid 3D parts for a wide variety of user-specified joint designs, our algorithm does have some limitations. First, since we only search over a fixed set of 26 disassembly directions for parts, our tool may deem some input surface 2D parts as not assemblable when in fact they are. One way to mitigate this problem is to increase the search space over additional disassembly directions, but each additional direction adds a dimension to the search space. Combining our approach with an analytical model of the directional blocking graph [Romney et al. 1995] is a direction for future work.

Second, since our algorithm focuses on generating SOPA-compatible parts, it can only generate joints that assemble via single translational motions. Our approach cannot generate joints that assemble via rotational motions, multi-step translations, or deformation-based snap-fit motions. Handling such joints would likely require a much higher-dimensional search for identifying collision-free assembly sequences and more sophisticated algorithms for distributing volume between the parts.



Third, we know of one algorithmic limitation. In the inset figure, red and green 2D parts have identical maximum 3D parts. Our algorithm cannot generate 3D geometries for the red and green squares, because neither reallocation attempted by our intersection resolution leaves both parts connected. However, 3D joint geometry similar to a cross lap joint is a valid solution. Note that this is the only algorithmic limitation we have discovered and, after creating over 100 joints covering a large design space, it never arose in practice.



Fourth, as our tool is focused on interactively designing decorative joinery, users must manually design joints to correct all part instabilities. As an alternative, we have experimented with automatically adding hidden joints. These are joints on the mating surfaces between two unstable parts that interlock them yet cannot be seen on any surface of the assembled furniture. In our experiments, we automatically add mortise-and-tenon joints aligned with the disassembly direction to the mating surfaces. This appears to work well in many cases, but more work is needed to generate hidden joints when the disassembly direction is parallel to the mating surface.

Fifth, our tool requires the user to draw surface 2D parts as input, that is, a complete partition of the solid 3D model surface. For more complex models or joinery designs, creating a complete partition can sometimes be time-consuming. We believe exploring methods to facilitate or partly automate the partitioning process provides a fruitful direction for future work.

7. CONCLUSION

We have described an interactive joinery design tool. Users draw the visual appearance of joints on the surface of an input model, and our tool automatically generates an assemblable set of solid 3D parts that conform to the drawings. Our tool reduces by a dimension the amount of user effort needed to create assemblable solid 3D parts. In addition, we provide a stability analysis technique for analyzing the resulting assembly of solid 3D parts to identify part instabilities. When our tool finds unstable parts that can move with respect to one another, it visualizes the instability and suggests where the user might draw additional surface regions to fix the problem. Thus, our approach can save the user from unnecessary costly and lengthy fabrication. As personalized fabrication devices become more and more ubiquitous, we believe that design tools such as ours are needed to make fabrication more accessible.

ACKNOWLEDGMENTS

We are grateful to Hijung V. Shin and Etienne Vouga for helpful discussions and sharing their EM code. We thank Dufficy Fine Carpentry and Design for fabricating the wooden Branca chair, Greg Niemeyer and Ravi Ramamoorthi for graciously lending MakerBot 3D printers, and William Walsh for taking photographs of the 3D printed decorative joints.

REFERENCES

Maneesh Agrawala, Doantam Phan, Julie Heiser, John Haymaker, Jeff Klingner, Pat Hanrahan, and Barbara Tversky. 2003. Designing effective step-by-step assembly instructions. *ACM Trans. Graph.* 22, 3 (July 2003), 828–837.

Moritz Bächer, Emily Whiting, Bernd Bickel, and Olga Sorkine-Hornung. 2014. Spin-it: Optimizing moment of inertia for spinnable objects. *ACM Trans. Graph.* 33, 4, Article 96 (July 2014), 96:1–96:10.

Desai Chen, Pitchaya Sitthi-amorn, Justin T. Lan, and Wojciech Matusik. 2013. Computing and fabricating multiplanar models. *Comput. Graph. Forum* 32, 2pt3 (2013), 305–315.

Xuelin Chen, Hao Zhang, Jinjie Lin, Ruizhen Hu, Lin Lu, Qixing Huang, Bedrich Benes, Daniel Cohen-Or, and Baoquan Chen. 2015. Dapper: Decompose-and-pack for 3d printing. *ACM Trans. Graph.* 34, 6, Article 213 (Oct. 2015), 213:1–213:12.

Paolo Cignoni, Nico Pietroni, Luigi Malomo, and Roberto Scopigno. 2014. Field-aligned mesh joinery. *ACM Trans. Graph.* 33, 1, Article 11 (Feb. 2014), 11:1–11:12.

Chi-Wing Fu, Peng Song, Xiaoqi Yan, Lee Wei Yang, Pradeep Kumar Jayaraman, and Daniel Cohen-Or. 2015. Computational interlocking furniture assembly. *ACM Trans. Graph.* 34, 4, Article 91 (July 2015), 91:1–91:11.

Suresh Goyal, Andy Ruina, and Jim Papadopoulos. 1991. Planar sliding with dry friction, Part 1. Limit surface and moment function. *Wear* 143 (1991), 307–330.

Leonidas J. Guibas, Dan Halperin, Hirohisa Hirukawa, Jean-Claude Latombe, and Randall H. Wilson. 1995. A simple and efficient procedure for polyhedral assembly partitioning under infinitesimal motions. In *IEEE Robotics and Automation*, Vol. 3. IEEE, 2553–2560.

Philipp Herholz, Wojciech Matusik, and Marc Alexa. 2015. Approximating free-form geometry with height fields for manufacturing. *Comput. Graph. Forum* 34, 2 (2015), 239–251.

Kristian Hildebrand, Bernd Bickel, and Marc Alexa. 2012. crdbrd: Shape fabrication by sliding planar slices. *Comp. Graph. Forum* 31, 2pt3 (May 2012), 583–592.

Ruizhen Hu, Honghua Li, Hao Zhang, and Daniel Cohen-Or. 2014. Approximate pyramidal shape decomposition. *ACM Trans. Graph.* 33, 6, Article 213 (Nov. 2014), 213:1–213:12.

S. Huerta. 2001. *Mechanics of Masonry Vaults: The Equilibrium Approach*. Universidade do Minho (2001).

Danny M. Kaufman, Shinjiro Sueda, Doug L. James, and Dinesh K. Pai. 2008. Staggered projections for frictional contact in multibody systems. *ACM TOG (SIGGRAPH Asia)* 27, 5 (2008), 1–11.

Manfred Lau, Akira Ohgawara, Jun Mitani, and Takeo Igarashi. 2011. Converting 3D furniture models to fabricatable parts and connectors. *ACM Trans. Graph.* 30, 4, Article 85 (July 2011), 85:1–85:6.

Honghua Li, Ibraheem Alhashim, Hao Zhang, Ariel Shamir, and Daniel Cohen-Or. 2012. Stackabilization. *ACM Trans. Graph.* 31, 6, Article 158 (Nov. 2012), 158:1–158:9.

Honghua Li, Ruizhen Hu, Ibraheem Alhashim, and Hao Zhang. 2015. Foldabilizing furniture. *ACM Trans. Graph.* 34, 4, Article 90 (July 2015), 90:1–90:12.

Lin Lu, Andrei Sharf, Haisen Zhao, Yuan Wei, Qingnan Fan, Xuelin Chen, Yann Savoye, Changhe Tu, Daniel Cohen-Or, and Baoquan Chen. 2014. Build-to-last: Strength to weight 3D printed objects. *ACM Trans. Graph.* 33, 4, Article 97 (July 2014), 97:1–97:10.

Linjie Luo, Ilya Baran, Szymon Rusinkiewicz, and Wojciech Matusik. 2012. Chopper: Partitioning models into 3D-printable parts. *ACM Trans. Graph.* 31, 6, Article 129 (Nov. 2012), 129:1–129:9.

James McCrae, Nobuyuki Umetani, and Karan Singh. 2014. FlatFitFab: Interactive modeling with planar sections. In *User Interface Software and Technology (UIST)*. 13–22.

Jean Jaques Moreau. 1966. Quadratic programming in mechanics: One-sided constraints. *J. SIAM Contr.* 4, 1 (1966), 153–158.

T. Noll. 2009. *Joint Book: The Complete Guide to Wood Joinery*. Chartwell Books.

J A Ochsendorf. 2002. *Collapse of Masonry Structures. University of Cambridge*. Ph.D. Dissertation. University of Cambridge.

Yeonjoo Oh, Gaberial Johnson, Mark Gross, and Ellenyi-Luen Do. 2006. The designosaur and the furniture factory. In *Design Computing and Cognition*, Gero Johns (Ed.). Springer, Amsterdam, 123–140.

J. Postell. 2012. *Furniture Design*. Wiley.

Romain Prévost, Emily Whiting, Sylvain Lefebvre, and Olga Sorkine-Hornung. 2013. Make it stand: Balancing shapes for 3D fabrication. *ACM Trans. Graph.* 32, 4, Article 81 (July 2013), 81:1–81:10.

Gary Rogowski. 2002. *The Complete Illustrated Guide To Joinery. Complete Illustrated Guide*. Taunton Press.

Bruce Romney, Cyprien Godard, Michael Goldwasser, G. Ramkumar, and others. 1995. An efficient system for geometric assembly sequence generation and evaluation. *Computers in Engineering* (1995), 699–712.

Daniel Saakes, Thomas Cambazard, Jun Mitani, and Takeo Igarashi. 2013. PacCAM: Material capture and interactive 2D packing for efficient material usage on CNC cutting machines. In *User Interface Software and Technology (UIST)*. 441–446.

Greg Saul, Manfred Lau, Jun Mitani, and Takeo Igarashi. 2011. SketchChair: An all-in-one chair design system for end users. In *Tangible, Embedded, and Embodied Interaction (TEI)*. 73–80.

Adriana Schulz, Ariel Shamir, David I. W. Levin, Pitchaya Sitthi-amorn, and Wojciech Matusik. 2014. Design and fabrication by example. *ACM Trans. Graph.* 33, 4, Article 62 (July 2014), 62:1–62:11.

Yuliy Schwartzburg and Mark Pauly. 2013. Fabrication-aware design with intersecting planar pieces. *Comput. Graph. Forum* 32, 2 (2013), 317–326.

K. Seike, Y. Yobuko, and R.M. Davis. 1986. *The Art of Japanese Joinery*. Weatherhill.

Hijung V. Shin. 2016. Personal communication.

Hijung V. Shin, Christopher F. Porst, Etienne Vouga, John Ochsendorf, and Frédo Durand. 2016. Reconciling elastic and equilibrium methods for static analysis. *ACM Trans. Graph.* 35, 2 (2016), 13–16.

Peng Song, Chi-Wing Fu, and Daniel Cohen-Or. 2012. Recursive interlocking puzzles. *ACM Trans. Graph.* 31, 6, Article 128 (Nov. 2012), 128:1–128:10.

Ondrej Stava, Juraj Vanek, Bedrich Benes, Nathan Carr, and Radomír Měch. 2012. Stress relief: Improving structural strength of 3D printable objects. *ACM Trans. Graph.* 31, 4, Article 48 (July 2012), 48:1–48:11.

Godfried Toussaint. 1985. Movable separability of sets. In *Computational Geometry*. Citeseer.

Nobuyuki Umetani, Takeo Igarashi, and Niloy J. Mitra. 2012. Guided exploration of physically valid shapes for furniture design. *ACM Trans. Graph.* 31, 4, Article 86 (July 2012), 86:1–86:11.

Nobuyuki Umetani, Yuki Koyama, Ryan Schmidt, and Takeo Igarashi. 2014. Pteromys: Interactive design and optimization of free-formed free-flight model airplanes. *ACM Trans. Graph.* 33, 4, Article 65 (July 2014), 65:1–65:10.

Weiming Wang, Tuanfeng Y. Wang, Zhouwang Yang, Ligang Liu, Xin Tong, Weihua Tong, Jiansong Deng, Falai Chen, and Xiuping Liu. 2013. Cost-effective printing of 3D objects with skin-frame structures. *ACM Trans. Graph.* 32, 6, Article 177 (Nov. 2013), 177:1–177:10.

Emily Whiting, John Ochsendorf, and Frédo Durand. 2009. Procedural modeling of structurally-sound masonry buildings. *ACM Trans. Graph.* 28, 5, Article 112 (Dec. 2009), 112:1–112:9.

Emily Whiting, Hijung Shin, Robert Wang, John Ochsendorf, and Frédo Durand. 2012. Structural optimization of 3D masonry buildings. *ACM Trans. Graph.* 31, 6, Article 159 (Nov. 2012), 159:1–159:11.

Randall H. Wilson. 1992. *On Geometric Assembly Planning*. Ph.D. Dissertation. Stanford University.

Shiqing Xin, Chi-Fu Lai, Chi-Wing Fu, Tien-Tsin Wong, Ying He, and Daniel Cohen-Or. 2011. Making burr puzzles from 3D models. *ACM Trans. Graph.* 30, 4, Article 97 (July 2011), 97:1–97:8.

Yong-Liang Yang, Jun Wang, and Niloy J. Mitra. 2015. Reforming shapes for material-aware fabrication. *Comput. Graph. Forum* 34, 5 (Aug. 2015), 53–64.

Miaojun Yao, Zhili Chen, Linjie Luo, Rui Wang, and Huamin Wang. 2015. Level-set-based partitioning and packing optimization of a printable model. *ACM Trans. Graph.* 34, 6, Article 214 (Oct. 2015), 214:1–214:11.

Qingnan Zhou, Eitan Grinspun, Denis Zorin, and Alec Jacobson. 2016. Mesh arrangements for solid geometry. *ACM Trans. Graph.* 35, 4, Article 39 (July 2016), 39:1–39:15.

Received October 2016; revised December 2016; accepted December 2016