MiGumi: Making Tightly Coupled Integral Joints Millable — Supplemental —

ADITYA GANESHAN, Brown University, United States of America
KURT FLEISCHER, Pixar Animation Studios, United States of America
WENZEL JAKOB, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
ARIEL SHAMIR, Reichman University, Israel
DANIEL RITCHIE, Brown University, United States of America
TAKEO IGARASHI, University of Tokyo, Japan
MARIA LARSSON, University of Tokyo, Japan

ACM Reference Format:
Aditya Ganeshan, Kurt Fleischer, Wenzel Jakob, Ariel Shamir, Daniel Ritchie, Takeo Igarashi, and Maria Larsson. 2025. MiGumi: Making Tightly Coupled Integral Joints Millable — Supplemental —. In *Proceedings of The 18th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia (SIGGRAPH Asia '25).* ACM, New York, NY, USA, Article 193, 7 pages. https://doi.org/10.1145/3763304

1 Details: Modeling Millable Geometry

1.1 Cylinder Minkowski Sum Proposition

Our representation, MXG leverages flat extrusions that satisfy the minkowski condition to ensure that each subtraction V_i is millable. Here, we first provide the proposition and its proof that this representation is built on.

We begin by characterizing the class of solids that can be constructed via subtractive milling[1]. A solid $P \subset \mathbb{R}^3$ is millable if it can be expressed as:

$$P = M - \bigcup_i V_i, \tag{1}$$

where M is the material stock and each V_i is a *millable* volume removed by the i-th milling operation. To be millable, each V_i must be a Minkowski sum[2] with respect to the milling tool's rotational swept volume. The rotational sweep of flat-end uniform-radius drill-bit yields a cylinder. Therefore, this can be written as:

$$V_i = X_i \oplus \mathrm{Cyl}_i, \tag{2}$$

where X_i is an arbitrary shape and $\mathrm{Cyl}_i \subset \mathbb{R}^3$ is the rotational sweep of the drill bit for the i-th operation. To enforce accessibility, Cyl_i is modeled as a semi-infinite cylinder extending away from the milling direction, ensuring that each V_i is accessible from outside. This formulation guarantees millability by construction: each subtraction aligns with tool geometry and access constraints, making the resulting solid P physically achievable through sequential milling.

The following proposition characterizes when a solid admits such a representation:

Proposition 1.1. *Let $V \subset \mathbb{R}^3$ be a solid, and let Cyl_r denote the semi-infinite cylinder of radius r along the z-axis. Then $V = X \oplus Cyl^r$ for some solid $X \subset \mathbb{R}^3$ if and only if:*

1. *(**Minkowski Condition**) For every z, the horizontal slice*
$$C(z) = \left\{(x, y) \in \mathbb{R}^2 \;\middle|\; (x, y, z) \in V\right\}$$
satisfies $C(z) = X(z) \oplus B_r$ for some $X(z) \subset \mathbb{R}^2$, where B_r is a circle of radius r.
2. *(**Nesting Condition**) For all $z_1 < z_2$, $C(z_1) \subseteq C(z_2)$.*

Intuitively, this proposition states that a solid is millable with a flat-end bit of radius r along a direction (e.g., the z-axis) if, when sliced perpendicular to that direction, each slice is a Minkowski sum of some base shape with a disk of radius r, and the slices nest monotonically—i.e., the volume forms a heightfield from the milling direction. Although stated here for the z-axis, the same conditions apply for any milling direction $\mathbf{n} \in \mathbb{S}^2$ after rotation.

Proof (if direction). **(⇐)** Suppose $V \subset \mathbb{R}^3$ satisfies the following conditions:

1. For each height z, the horizontal cross-section
$$C(z) = \{(x, y) \in \mathbb{R}^2 \mid (x, y, z) \in V\}$$
satisfies $C(z) = X(z) \oplus B_r$ for some planar shape $X(z) \subset \mathbb{R}^2$.
2. The slices are nested: for all $z_1 < z_2$, $C(z_1) \subseteq C(z_2)$.

We define the solid: $X = \{(x, y, z) \in \mathbb{R}^3 \mid (x, y) \in X(z)\}$, and let $\mathrm{Cyl}_r = B_r \times [0, \infty)$ denote the semi-infinite vertical cylinder of radius r.

We will show that $V = X \oplus \mathrm{Cyl}_r$.

(i) $V \subseteq X \oplus \mathbf{Cyl}_r$**:** Let $(x, y, z) \in V$. Then by definition, $(x, y) \in C(z) = X(z) \oplus B_r$. So there exist $(x', y') \in X(z)$, $(u, v) \in B_r$ such that $(x, y) = (x' + u, y' + v)$. Letting $w = 0$, we have:

$$(x, y, z) = (x', y', z) + (u, v, w),$$

[1] We assume all solids are regular closed subsets of $\mathbb{R}^3$, i.e., equal to the closure of their interior.

[2] Given two sets $A, B \subset \mathbb{R}^n$, their Minkowski sum is defined as $A \oplus B = \{a + b \mid a \in A,\ b \in B\}$.

Authors' Contact Information: Aditya Ganeshan, adityaganeshan@gmail.com, Brown University, Providence, United States of America; Kurt Fleischer, kurt@pixar.com, Pixar Animation Studios, San Francisco, United States of America; Wenzel Jakob, wenzel.jakob@epfl.ch, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland; Ariel Shamir, arik@runi.ac.il, Reichman University, Herzliya, Israel; Daniel Ritchie, daniel_ritchie@brown.edu, Brown University, Providence, United States of America; Takeo Igarashi, takeo.igarashi@gmail.com, University of Tokyo, Tokyo, Japan; Maria Larsson, ma.ka.larsson@gmail.com, University of Tokyo, Tokyo, Japan.

SIGGRAPH Asia '25, Hong Kong, China
© 2025 Copyright held by the owner/author(s).
ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/3763304

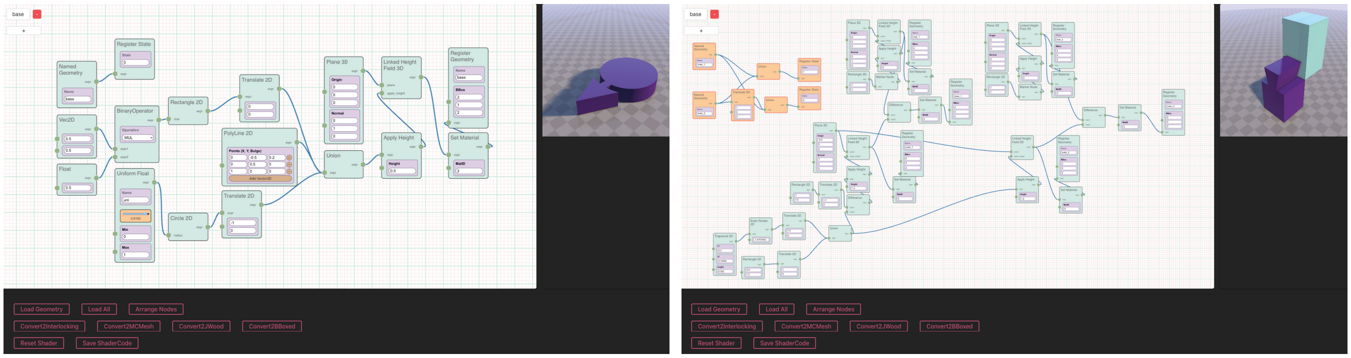


Fig. 1. Our custom in-house node-based visual programming interface. On the left we show a simple program to create a shape. On the right we show the program to create a simple joint "Ari Tsugi". Notice the reuse of sub-graphs for 2D CSG expression between the two parts. Further, we also record the assembly sequence using a state machine. We highlight the nodes which correspond to the Assembly states in the right image (top left of the right image).

where $(x', y', z) \in X$ and $(u, v, w) \in \mathrm{Cyl}_r$. Thus, $(x, y, z) \in X \oplus \mathrm{Cyl}_r$.

(ii) $X \oplus \mathbf{Cyl}_r \subseteq V$**:** Let $(x, y, z) \in X \oplus \mathrm{Cyl}_r$. Then there exist $(x', y', z') \in X$, $(u, v, w) \in \mathrm{Cyl}_r$ such that:

$$(x, y, z) = (x' + u, y' + v, z' + w),$$

with $(x', y') \in X(z')$, $(u, v) \in B_r$, $w \geq 0$. By the nesting condition, $X(z') \subseteq X(z' + w)$. So $(x', y') \in X(z)$, and hence $(x, y) \in X(z) \oplus B_r = C(z)$. Thus, $(x, y, z) \in V$.

Conclusion: We have shown that $V \subseteq X \oplus \mathrm{Cyl}_r$ and $X \oplus \mathrm{Cyl}_r \subseteq V$, hence:

$$V = X \oplus \mathrm{Cyl}_r. \quad \square$$

□

Proof (only if direction). **(⟹)** Assume $V = X \oplus \mathrm{Cyl}_r$, where $\mathrm{Cyl}_r = B_r \times [0, \infty)$ is a semi-infinite cylinder aligned with the z-axis.

We will show that the horizontal slices of V satisfy both the Minkowski and nesting conditions stated in the proposition.

(i) Minkowski Condition. Fix any height $z \in \mathbb{R}$, and define the horizontal slice of V at height z as

$$S(z) = \{(x, y) \in \mathbb{R}^2 \mid (x, y, z) \in V\}.$$

Since $V = X \oplus \mathrm{Cyl}_r$, for any $(x, y, z) \in V$, there exists $(x', y', z') \in X$ and $(u, v, w) \in B_r \times [0, \infty)$ such that

$$(x, y, z) = (x', y', z') + (u, v, w).$$

This implies $z = z' + w$ with $w \geq 0$, so $z' \leq z$. Let us define

$$X(z) = \{(x', y') \in \mathbb{R}^2 \mid \exists z' \leq z \text{ such that } (x', y', z') \in X\}.$$

Then for any $(x, y) \in S(z)$, we have $(x, y) \in X(z) \oplus B_r$. Conversely, for any $(x', y') \in X(z)$ and $(u, v) \in B_r$, the point $(x' + u, y' + v, z) \in V$. Therefore,

$$S(z) = X(z) \oplus B_r.$$

(ii) Nesting Condition. Let $z_1 < z_2$. Then by construction, $X(z_1) \subseteq X(z_2)$, since $\{z' \leq z_1\} \subseteq \{z' \leq z_2\}$. Taking Minkowski sums with B_r, we get:

$$S(z_1) = X(z_1) \oplus B_r \subseteq X(z_2) \oplus B_r = S(z_2),$$

as required. □

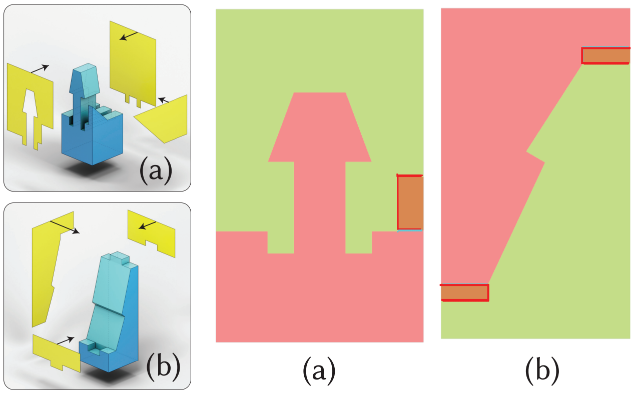


Fig. 2. Fixed interface constraints during slice-wise optimization. We show two example joints with the interface with non-aligned fixed extrusions highlighted in red. These fixed interfaces are preserved by enforcing a strong boundary loss, allowing the optimization to proceed on a compact set of slice-aligned contours without hampering tight coupling between non-aligned extrusions.

A constructive way to satisfy these conditions is to extrude a 2D region $C_i \subset \mathbb{R}^2$, embedded in a plane orthogonal to the milling direction $\mathbf{n}_i$, over a semi-infinite interval $(-\infty, h_i)$. This ensures that the volume forms a heightfield along $\mathbf{n}_i$, satisfying the nesting condition. If C_i additionally satisfies the Minkowski condition (i.e., $C_i = X_i \oplus B_r$ for some $X_i \in \mathbb{R}^2$), then the resulting solid is millable with a flat drill bit of radius r. This leads naturally to our core primitive: the subtractive extrusion, which forms the foundation of the MXG representation.

1.2 Valid Dilation and Erosion Operations

As described in the main paper, each extrusion field $\mathcal{E}_i$ is defined by a 2D signed distance function (SDF) f_i, constructed using symbolic expressions composed of primitives and Boolean operations such as union, intersection, difference, and complement. While this representation offers high expressivity, the general protocol for evaluating such expressions using min/max functions yields only pseudo-SDFs—functions that preserve the sign of the distance but do not provide accurate Euclidean values [1].

This approximation is detrimental for morphological operations such as dilation and erosion, which rely on exact distance values to compute correct offsets. To address this, we convert each symbolic expression into a PolyCurve representation: a closed non-intersecting polygonal chain composed of straight-line segments and circular arcs. We ensure that all Boolean operations (e.g., intersections or differences) are performed such that no two primitives have intersecting boundaries. This guarantees that min/max-based SDF evaluation produce exact SDFs.

Beyond improving accuracy for morphological operations, PolyCurve conversion also facilitates optimization. The original symbolic representation may lack sufficient degrees of freedom to reduce loss terms effectively. In contrast, PolyCurves provide direct, parameterized control over geometric elements, improving convergence during optimization.

1.3 Authoring Interface

We construct MXG_0 programs for each joint using a custom in-house node-based visual programming interface, shown in Figure 2. The interface exposes symbolic construction of extrusion profiles via basic geometric primitives and 2D CSG operations. Though minimal in design, it supports parametric control and real-time feedback.

Internally, each expression is compiled into GLSL shader code and rendered using ray marching—enabling direct visualization without conversion to triangle meshes. Our renderer builds on Inigo Quilez's ShaderToy pipeline, allowing fast previews of complex joint geometries.

While the current interface is targeted at expert users, we view it as a foundation for broader tooling. Making this system usable by non-experts remains important future work.

2 Details: Restoring Tight Coupling

2.1 Fixed Interface Constraint in Equivalence Slice Sets

Our optimization process operates slice-by-slice, but many slices are redundant in their contribution to the coupling loss. To reduce computational cost, we identify a compact set of representative slices that together cover the lateral surface of the joint. We refer to this set as the *equivalence slice set.* Each slice is selected from a set of planar extrusions that share tool direction and overlapping coupling regions.

However, not all interfaces are optimized simultaneously. Interfaces with extrusions from non-aligned directions—i.e., those not belonging to the current slicing direction—are treated as fixed. Figure 2 (a,b) shows examples of such non-aligned interfaces (highlighted with lines) which are held constant during a particular slice-aligned optimization stage.

To preserve surface continuity at these boundaries, we enforce a strong *boundary loss* that penalizes deviation from the current surface geometry. This effectively treats the extrusions from other directions as fixed walls, guiding the optimization toward solutions that remain compatible with them.

Algorithm 1 Optimization from MXG_0 Program to MXG_r Program

```
Require: MXG_0 Program $\mathcal{P}_0$, target drill radius $r$
 1: for each direction $\mathbf{n}$ do
 2:     Identify extrusion fields aligned with $\mathbf{n}$
 3:     Sample slicing planes orthogonal to $\mathbf{n}$
 4:     Group extrusions into equivalence sets $\{\mathcal{S}_1, \ldots, \mathcal{S}_K\}$
 5:     for each equivalence set $\mathcal{S}_k$ do
 6:         Freeze misaligned boundary constraints
 7:         for $t = 1$ to $T$ do
 8:             Increase dilation rate $r_d \leftarrow \mathrm{schedule}(t)$
 9:             Sample points on $\partial g$ & $\partial C_i$ on each slice
10:             Compute total loss $\mathcal{L}_{\mathrm{total}}$
11:             Update $g_i$ parameters via gradient descent
12: return Final MXG_r Program $\mathcal{P}_r$
```

2.2 Optimization

Algorithm 1 summarizes the full pipeline used to optimize MXG_0 program to its fabricable MXG_r form.

None of the algorithm's steps require manual input. Extrusion fields aligned with a sampled direction $\mathbf{n}$ are detected via their dot product with $\mathbf{n}$ (step 2). Slices are then generated at regular intervals along this direction as intersections between the fields and planes at the sampled positions (step 3). Finally, to form equivalence sets, we assign each slice a signature based on the extrusions it intersects with (each slice's signature is the set of signatures of the extrusions it intersects with) and group slices that share the same signature to form the equivalence sets (step 4).

Sampling points on Part contours ∂C_i. : Let $\mathcal{E}_i$ be an extrusion aligned with the slicing plane z_k, defined via a 2D signed distance function $g_i : \mathbb{R}^2 \to \mathbb{R}$. Its dilated contour ∂C_i is defined by the r-level set: $\partial C_i = \{x \mid f_i(x) = r\}$. Now, f_i is constructed from boolean compositions over *PolyCurve* primitives—closed curves composed of line and arc segments. Therefore, we can sample points $x_\ell \in \partial C_i$ in a differentiable way using a three-step approach:

1. **Offset curve primitives**: Sample points on each line and arc segment of the polycurve and offset them outward along the normal by distance r.
2. **Handle corners**: At each vertex v_i of the polycurve, generate a circular arc of radius r and sample along it.
3. **Contour filtering**: Reject any point x for which $f_i(x) \neq r$, ensuring that retained samples lie on the true offset contour.

This sampling strategy avoids constructing a closed-form expression for the offset curve and remains fully differentiable with respect to the parameters of f_i.

Per-part 2D Signed Distance Field. Given a sample $x_\ell \in \partial C_i(z_k)$, we compute its signed distance from the 2D projection of another part P^b in the same slice plane. This part is represented as $P^b(z_k) = M(z_k) - \bigcup_j \overline{\mathcal{E}_j(z_k)}$. Since all components—$M(z_k)$ and $\overline{\mathcal{E}_j(z_k)}$—are represented as signed distance functions (SDFs), we construct a pseudo-SDF for $P^b(z_k)$ using min/max composition. The distance from part $\partial P^b(z_k)$ can then be evaluated as: $\mathrm{SDF}_{P^b}(x_\ell)$. Although this pseudo-SDF may deviate from the true Euclidean SDF away from the boundary, it matches closely near the zero-contour, where

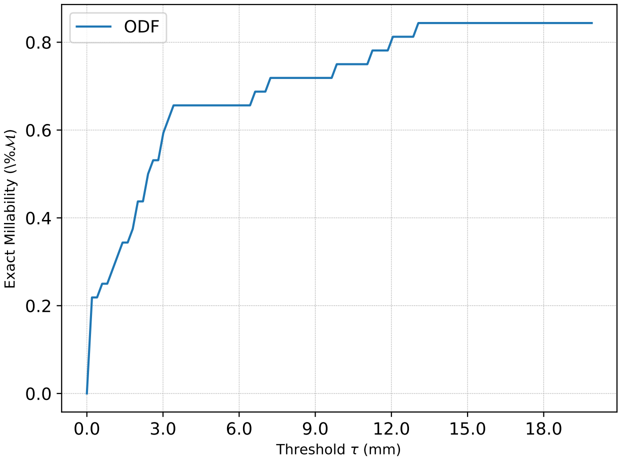


Fig. 3. Threshold sensitivity analysis. Left: *Exact Millability* (%$\mathcal{M}$) of the ODF method across varying distance thresholds. Trends remain consistent across a wide range, confirming that the results in the main paper are not sensitive to specific parameter choices.

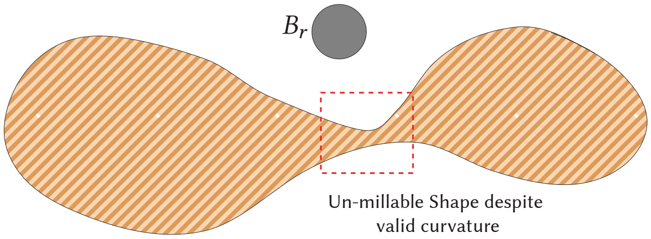


Fig. 4. For millability of a given shape with a drill bit of radius r, constraining the boundary curvature to be less than $1/r$ is insufficient.

our optimization is concentrated. We provide additional discussion and analysis in the supplementary.

Occupancy Preservation Loss. We define occupancy as a continuous field estimated using the signed distance function SDF_P for each part P. Let $O(x)$ be the binary occupancy of the original shape (pre-optimization), and let $\hat{O}_\theta(x)$ denote the soft occupancy field of the optimized part at point x, derived from its SDF using a sigmoid activation:

$$\hat{O}_\theta(x) = \sigma(-\alpha \cdot \mathcal{D}_P(x)), \tag{3}$$

where α controls the sharpness of the transition. We then define the occupancy loss as:

$$\mathcal{L}_{\text{occ}} = \sum_{x \in \mathcal{G}} \left(\hat{O}_\theta(x) - O(x) \right)^2, \tag{4}$$

where $\mathcal{G}$ is a uniform grid of sample points within the coupling volume. This loss encourages the optimized parts to remain close to their original geometry in both shape and extent, acting as a regularizer on the optimization.

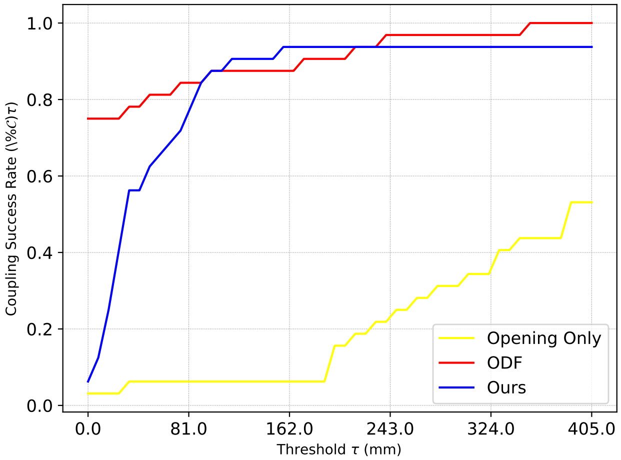


Fig. 5. Threshold sensitivity analysis. Right: *Coupling Success Rate* (C_τ) across the same thresholds. Trends remain consistent across a wide range, confirming that the results in the main paper are not sensitive to specific parameter choices.

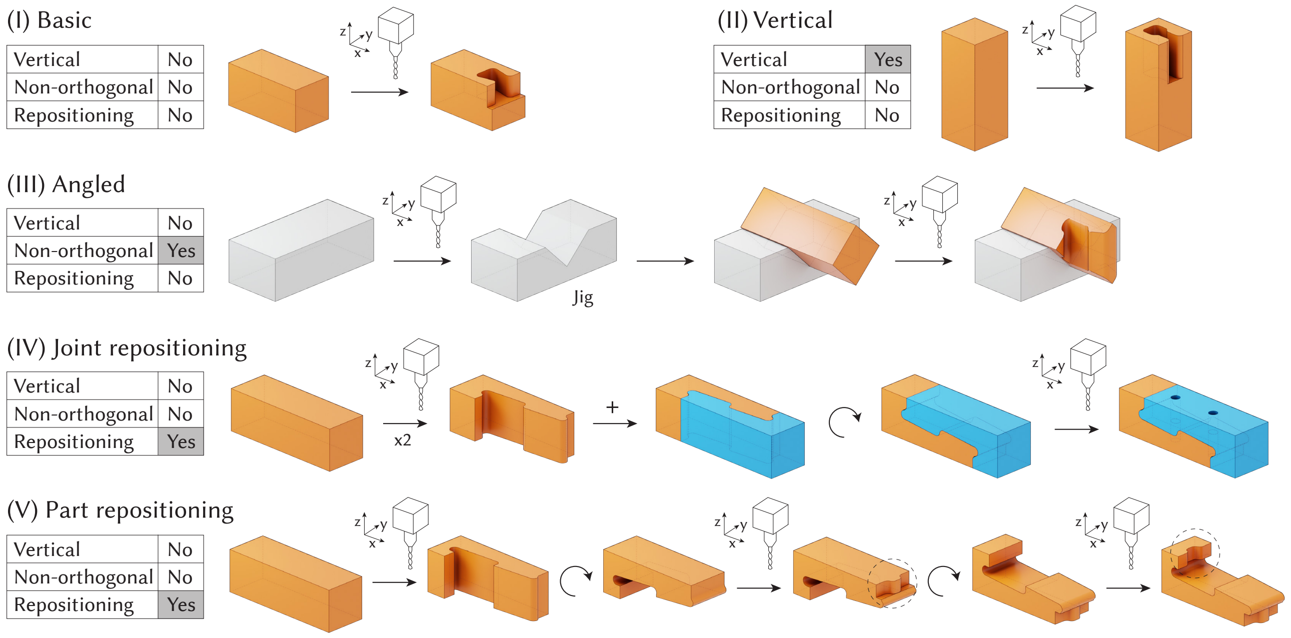
2.3 Alternate Approaches

We considered two alternate strategies for generating millable, tightly coupled geometries, but found them either insufficient or impractical in our setting.

Iterative ODF. One possible approach is to alternate morphological opening with Diff-Flip updates, progressively restoring coupling after each step. However, in practice, this method encounters major hurdles. First, most 2D boolean libraries lack robust support for polycurve representations involving both lines and arcs, making repeated set operations error-prone. Even with rasterized occupancy grids and morphological operations, the procedure becomes unstable after just a few iterations. Moreover, convergence is not guaranteed, and thin features are often lost due to over-erosion. While simple in principle, this approach fails to yield consistent or controllable results.

Curvature Constraint. Another approach can be to constrain the curvature of 2D contours used in subtractive profiles. Since flat-end tools cannot produce sharp turns, one might attempt to ensure that the curvature of all profile paths remains below $1/r$, where r is the milling tool radius. When a contour is at the interface of two joints, we can then constraint the absolute curvature to be less than $1/r$. While this condition is necessary, it is not sufficient: a curve may have bounded curvature but still violate millability if opposing path segments come too close. Figure 4 shows a counterexample where the curvature constraint holds, yet the tool cannot pass through due to local underclearance. Millability imposes global constraints on offset distance, not just local curvature bounds.

One might attempt to strengthen the curvature condition by also enforcing a minimum local thickness, requiring that opposing segments of a contour remain at least one tool diameter apart. However, implementing such a rule is non-trivial: it demands reliable detection of thin regions and careful reshaping to increase clearance without



(I) Basic

Vertical	No
Non-orthogonal	No
Repositioning	No

(II) Vertical

Vertical	Yes
Non-orthogonal	No
Repositioning	No

(III) Angled

Vertical	No
Non-orthogonal	Yes
Repositioning	No

(IV) Joint repositioning

Vertical	No
Non-orthogonal	No
Repositioning	Yes

(V) Part repositioning

Vertical	No
Non-orthogonal	No
Repositioning	Yes

Fig. 6. The five different types of milling procedures used to fabricate the eight physical outputs in Fig. 15 of the main paper.

altering intended contact geometry. Moreover, for joints involving more than two parts, it is unclear how curvature and thickness bounds should be formulated across several interacting boundaries. These challenges limit the practicality of a curvature-plus-thickness strategy and motivate the global offset-distance formulation adopted in our method.

2.4 Optimization and Implementation Details

We implement our optimization pipeline in PyTorch, using the AdamW optimizer with a learning rate of 0.003. Each slice-level optimization is run for 250 iterations, and we select the iteration with the lowest boundary loss as the final result. We ran all our experiments on an Alienware workstation equipped with a Intel i9 11900K CPU, a Geforce RTX 3090 GPU and 64 GBs of DDR4 RAM.

Prior to optimization, we convert all input extrusion profiles to non-intersecting polycurve representations. Due to occasional instability in boolean operations over polycurves (e.g., unresolved self-intersections or missing arcs), we manually corrected a small number of cases. These corrections apply only to the initial geometry and do not affect the optimization procedure. Additionally, for two joints in our dataset, we adjust the threshold used for detecting coupling boundaries due to noisy overlaps in the manually authored input geometry.

For joint assemblies involving three or more parts, we observe instability in the $\mathcal{M}_{\text{P}}$ gradient when a point on one milling path lies near multiple opposing paths. To mitigate this, we introduce a weighting scheme that downweights gradient contributions from such ambiguous regions. Specifically, we compute an entropy-based score over the top-k nearest opposing points and suppress gradients where the entropy exceeds a threshold, indicating poor localization of a unique coupled path. This improves convergence and avoids incorrect updates due to competing path associations.

3 Evaluation

Figure 3 plots *Exact Millability* (%$\mathcal{M}$) for ODF as a function of the distance threshold used to verify the Minkowski condition. Figure 5 shows the corresponding *Coupling Success Rate* (C_τ) across thresholds. Together, these plots demonstrate that the trends reported in the main paper are robust and not the result of hand-tuned thresholds.

4 Dataset

Our dataset consists of 30 traditional integral joint designs, each modeled parametrically using our MXG$_0$ representation. These parametric programs allow for continuous variation in dimensions, proportions, and milling configurations—enabling the synthesis of a much larger family of design variants from each base joint.

We plan to expand the dataset to include additional designs from historical catalogs and contemporary applications. Figure 8 provides an overview of all joint designs currently included in the dataset.

5 Details of the Physical Fabrication Process

We fabricated eight joints using a 3-axis CNC machine. The references (a)-(h) that follows refers to these physically fabricated joints (refer to in fig. 15 in the main paper). Joints (a) *Koshikake Ari Tsugi* and (b) *Kime Kata Tsugi* were fabricated in the most straight-forward manner, without repositioning and in flat orthogonal positions (fig. 6-I). For joints (c) *Kiguchi Ari* and (d) *Shimigiri Daimochi Tsugi*, one part each were fabricated in a vertical orthogonal position (fig. 6-II). Joint (e) *Sumi Niho Kama Tsugi* was positioned at a 45 degree angle to achieve the diagonal cut. To reliably position the material at the 45 degree angle, we first milled out a jig, and then positioned material there before milling (fig. 6-III). Joint (f) *Okkake Daisen Tsugi* was repositioned to accommodate the milling of holes facing in a different direction from the main geometries (fig. 6-IV). For this joint, we first fabricated the two main parts. Then we assembled them, and cut out the holes for the plugs in the assembled state. In this way, there is no loss in fabrication precision in terms of the tightness of the coupling despite the repositioning—even if the repositioning is slightly inexact, the holes in the two different parts will still be perfectly aligned because there are cut together. Joint (g) *Kanawa Tsugi* requires a different type of repositioning, namely, millings from three directions per part (fig. 6-V). This process is more error-prone, so fabrication with a 4-or-more axis CNC machine would have been preferred if we had had access to such a machine. Nonetheless, we managed to fabricate a functional joint with our 3-axis machine setup by careful repositioning.

The CNC machine we used are of the brand *Shopbot* and model *Desktop MAX with Aluminum T-Slot Deck*. The milling paths were created by importing meshes (*.stl files) of our system outputs to the software *VCarve*, and then generating roughing tool paths with clearance set to 0.0. To create a small tolerance of the joint, we set the milling bit size to slightly (about 0.05 inches) smaller than the actual mill bit size (1/4 inch). This will result in a slight over-cut, and can the amount can be adjusted depending on the desired tightness of the joint.

6 Additional Results

Expanding the Design Space. Our method accommodates complex joint configurations that lie outside the scope of prior systems. The use of general CSG-like 2D expressions enable us to cover a wider spread of designs. Furthermore, for certain joints, multi-directional milling is critical to ensure that tight coupling is feasible. Figure 7 shows such a joint. While fabrication of individual parts with a single subtractive operation is feasible, it results in shapes that cannot be coupled, despite optimization. In contrast, with two oblique subtractions, we can generate a design that is millable and tightly coupled (after optimization).

7 Frequently Asked Questions (FAQs)

During review, several thoughtful questions were raised about our approach; we summarize them here for clarity.

Does the method require substantial manual modeling? Some manual input is needed because no dataset of CNC-adapted traditional joints exists, making data-driven automation infeasible. Moreover, joint design involves subjective choices—such as where to allow small milling artifacts—that benefit from designer control. Our interactive editor streamlines this process by enforcing millability while preserving flexibility. Future work will explore partial automation of common edits and a user-study evaluation.

Is the approach limited to simple geometry or assemblies? Traditional joinery was historically shaped with planar tools such as chisels and saws, and rarely involves free-form 3D surfaces or elaborate multi-stage assemblies. Our method targets this regime and does not attempt to cover joints requiring sculpted surfaces or complex assembly choreography.

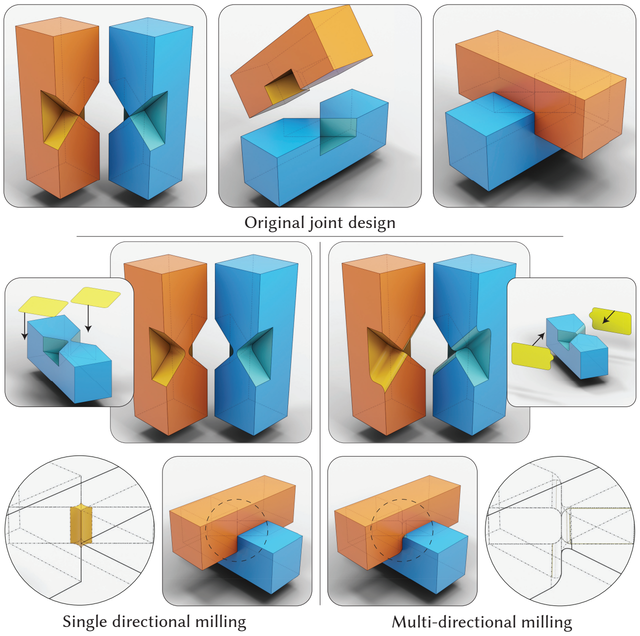


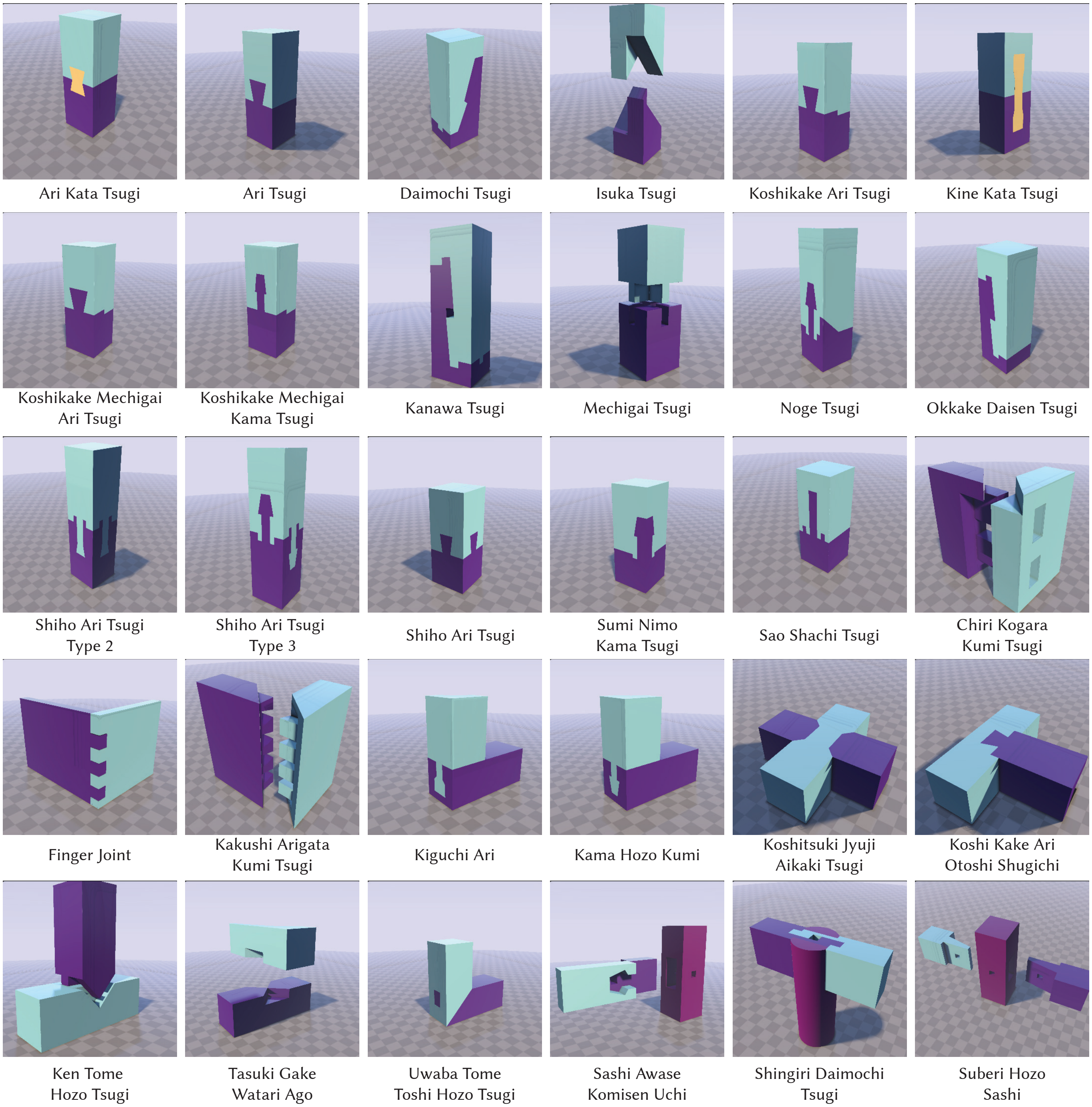
Fig. 7. For certain joints, multi-directional milling is critical to ensure that tight coupling is feasible. While fabrication of individual parts with a single subtractive operation is feasible, it results in shapes that cannot be coupled, despite optimization. In contrast, with two oblique subtractions, we can generate a design that is millable and tightly coupled (after optimization).

In Fig. 4(c), why are some joint boundaries rounded while others remain sharp? We apply a morphological opening with radius r to the subtraction region. This guarantees millability but differs from uniform filleting: only concave corners inside the removal region (yellow) are rounded, while other edges remain unchanged.

Is the method specific to 3-axis CNC milling? The geometry produced by our algorithm is compatible with 4- and 5-axis mills. We emphasise 3-axis setups because they are widely available in standard woodworking shops, while higher-axis machines can execute the same extrusions with fewer reorientations.

Why assume a single cutter radius? Could multiple tools improve efficiency? Our formulation adapts geometry for the smallest finishing radius, ensuring tight coupling and clearance. Standard practice—roughing with larger cutters and finishing with a small tool—is fully compatible with this geometry and supported by existing CAM software; our focus is on geometric correctness, not minimising machining time.

Does the approach guarantee global accessibility for the cutter? Every subtraction is constructed as a semi-infinite extrusion along a fixed direction (Supplementary Section 1), which guarantees a straight-line tool path along that direction, assuming feasible fixturing.



Ari Kata Tsugi

Ari Tsugi

Daimochi Tsugi

Isuka Tsugi

Koshikake Ari Tsugi

Kine Kata Tsugi

Koshikake Mechigai Ari Tsugi

Koshikake Mechigai Kama Tsugi

Kanawa Tsugi

Mechigai Tsugi

Noge Tsugi

Okkake Daisen Tsugi

Shiho Ari Tsugi Type 2

Shiho Ari Tsugi Type 3

Shiho Ari Tsugi

Sumi Nimo Kama Tsugi

Sao Shachi Tsugi

Chiri Kogara Kumi Tsugi

Finger Joint

Kakushi Arigata Kumi Tsugi

Kiguchi Ari

Kama Hozo Kumi

Koshitsuki Jyuji Aikaki Tsugi

Koshi Kake Ari Otoshi Shugichi

Ken Tome Hozo Tsugi

Tasuki Gake Watari Ago

Uwaba Tome Toshi Hozo Tsugi

Sashi Awase Komisen Uchi

Shingiri Daimochi Tsugi

Suberi Hozo Sashi

Fig. 8. Our Full dataset of 30 Joints. This dataset supports evaluation, benchmarking, and further research into CNC-fabricable joinery.

How are 2D mill path contours represented? They are stored as a sequence of straight segments and circular arcs, using a vertex–bulge format. For implementation details, please refer to the released code.

References

[1] Zoë Marschner, Silvia Sellán, Hsueh-Ti Derek Liu, and Alec Jacobson. 2023. Constructive Solid Geometry on Neural Signed Distance Fields. In *SIGGRAPH Asia 2023 Conference Papers* (Sydney, NSW, Australia) *(SA '23)*. Association for Computing Machinery, New York, NY, USA, Article 121, 12 pages. doi:10.1145/3610548.3618170