Computational Design of Coordinate-Motion Assemblies

YUKUN LU, University of Science and Technology of China, China and Singapore University of Technology and Design, Singapore
KE CHEN, University of Science and Technology of China, China and Singapore University of Technology and Design, Singapore
LIGANG LIU, University of Science and Technology of China, China and Laoshan Laboratory, China
PENG SONG*, Singapore University of Technology and Design, Singapore

Fig. 1. We propose a computational approach for designing coordinate-motion assemblies, which can be disassembled only by using a coordinated motion of component parts specified by users. We demonstrate this with a three-piece Bunny assembly. The columns display the initial, intermediate, and fully disassembled configurations, with the physical prototype (top) matching the virtual design (bottom). Colored arrows indicate the translational directions along which the parts must move *simultaneously* to reach the next configuration.

Coordinate-motion assemblies can only be disassembled by the *simultaneous* motion of multiple parts along distinct paths, providing high structural stability and enabling efficient robotic assembly. Existing examples are largely limited to architectural structures using joint-based connections, or puzzles created through trial-and-error, as the relationship between part geometry and coordinate motion remains poorly understood. Computationally designing such assemblies is challenging because it requires jointly achieving distributed contacts across the entire assembly and a *unique* coordinate motion for disassembly that rules out other feasible motions. We address this challenge by establishing a theoretical connection between part geometry and unique coordinate motion, enabling us to rigorously verify whether a given assembly admits a unique coordinate motion. Building on this theory, we introduce a two-stage algorithm that optimizes contact interfaces for a target motion and constructs physically feasible part geometries that conform to a user-specified global shape. We demonstrate our approach on models with complex geometries and topologies, including assemblies with large part counts, and validate it through physical fabrication and experiments.

CCS Concepts: • **Computing methodologies** → **Shape modeling**.

ACM Reference Format:
Yukun Lu, Ke Chen, Ligang Liu, and Peng Song. 2026. Computational Design of Coordinate-Motion Assemblies. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '26), July 19–23, 2026, Los Angeles, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3799902.3811068

1 Introduction

Most assemblies can be disassembled by removing one part after another in a *sequential* manner. In contrast, *coordinate-motion* assemblies can only be disassembled when all parts move *simultaneously* along different paths; see Fig. 1. Their disassembly plans are therefore non-sequential, and they are also called non-sequential assemblies [Glath et al. 2023b] or *m*-handed assemblies [Schwarzer et al. 1998]. This coordinate-motion property brings advantages unavailable to sequential ones: higher structural stability because disassembly needs multiple parts to move together, and it can facilitate robotic execution since robots can place many parts in one coordinated step. For humans, however, coordinate motion is challenging because two hands are insufficient to manipulate multiple parts at once. These characteristics make coordinate-motion assemblies attractive for structurally demanding architecture [Glath et al. 2023a], robotic assembly [Rossi et al. 2024], and mechanically intriguing puzzles [Coffin 2007].

*Corresponding author.

Authors' Contact Information: Yukun Lu, University of Science and Technology of China, China and and Singapore University of Technology and Design, Singapore, lyrik@mail.ustc.edu.cn; Ke Chen, University of Science and Technology of China, China and and Singapore University of Technology and Design, Singapore, ckck@mail.ustc.edu.cn; Ligang Liu, University of Science and Technology of China, China and and Laoshan Laboratory, China, lgliu@ustc.edu.cn; Peng Song, Singapore University of Technology and Design, Singapore, peng_song@sutd.edu.sg.

SIGGRAPH Conference Papers '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2554-8/2026/07
https://doi.org/10.1145/3799902.3811068

Existing design practices remain limited. Puzzle artisans rely on trial-and-error combinations of primitives like spheres [Bell 2011] or tetrahedra [Bell 2012], which limits both shape and motion variety. Architectural designs use mortise-and-tenon joints to obtain coordinate motion [Glath et al. 2023a], but adapting these joints to curved global shapes forces them to be short and narrow. This results in weak connections prone to separation, making them impractical for structural applications; see Fig. 13 (right). Prior work has represented coordinate motions via graph models [Glath et al. 2023a] or algebraic formulations [Bell 2024]. But they do not reveal the relation between geometry and coordinate motion, which makes the general design of coordinate-motion assemblies difficult.

Our goal is to theoretically analyze and computationally design coordinate-motion assemblies that are: (i) *contact-based*, utilizing distributed planar contacts rather than local joints; see Fig. 2; (ii) structurally stable, possessing a *unique* disassembly plan; (iii) physically feasible under common fabrication constraints; (iv) versatile in shape, conforming to any user-specified global shape. To the best of our knowledge, we are the first to formulate and solve the research problem of designing contact-based assemblies with unique coordinate motion.

The inputs to our problem are a global shape mesh and a set of 3D vectors in the global frame specifying the target translational velocities of the parts. To make the problem tractable, we assume contact interfaces are composed of planar faces. At the same time, we account for other feasible disassembly motions (e.g., rotations) and design to rule them out. Two main challenges need to be addressed. First, an inequality system $\mathbf{BY} \geq \mathbf{0}$ describing feasible infinitesimal motions of assemblies is known [Wang et al. 2019], but how to use this inequality system to test whether an assembly admits unique coordinate motion has been unclear. We observe that the sliding contacts (those aligned with the target relative disassembly motion) are the key to constraining the coordinate motion to be unique, and then we derive an equivalence between contact geometry and unique coordinate motion (Section 3). Second, finding a suitable contact-based geometry representation for the assemblies and searching for a configuration to achieve all the constraints is difficult. Our approach models contact interfaces (Section 4), and uses a two-stage design approach that first optimizes the interfaces to achieve the target motion, then constructs full watertight parts that respect fabrication and shape constraints (Section 5).

In summary, our contributions are:

- We establish a connection between geometry and unique coordinate motion, and propose a test algorithm to verify whether an assembly admits a unique coordinate motion.
- We propose a computational framework that designs contact-based coordinate-motion assemblies with prescribed shapes and motions while guaranteeing disassembly uniqueness and physical feasibility.

Our results demonstrate the effectiveness of the approach using diverse examples with different shapes, part counts, and topologies. We validate our designs with fabricated prototypes, opening new possibilities for applications in architecture, robotics, and puzzle design. Related code and data for this paper are publicly available at https://github.com/zLyrikz/Coordinate-MotionAssembly.

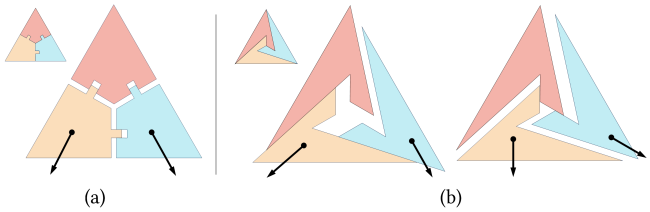

Fig. 2. Comparison between (a) *joint-based* and (b) *contact-based* coordinate-motion assemblies. The same contact-based coordinate-motion assembly has multiple disassembly motions. This raises the question of how to design contact-based assemblies while guaranteeing *unique* coordinate motion.

2 Related Work

Assembly planning. Assembly planning is the process of creating detailed plans to combine separate parts into the final structure. Assembly planning typically involves finding a sequence of operations to assemble the parts (assembly sequencing [Jiménez 2013]), and determining the motions that bring each part to its target pose (assembly path planning [Ghandi and Masehian 2015]).

In practice, the most widely used assembly plan is *linear*, *monotone*, and *coherent* due to its simplicity [Ghandi and Masehian 2015], where *linear* means each assembly operation involves the insertion of a single part, *monotone* means assembling parts does not require intermediate placement of subassemblies, and *coherent* means all parts inserted into the assembly must touch some previously-assembled part(s). Computational methods have been developed to find such simple plans for various tasks, including constructing architectural structures [Deuss et al. 2014], assembling mechanical parts [Tian et al. 2022], and solving puzzles [Markaki and Panagiotakis 2023]. Please refer to [Wang et al. 2021b] for a review.

Recently, there has been growing interest in studying complex assemblies with non-trivial assembly plans. In particular, non-linear plans have been computed for illustrating the disassembly of 3D complex objects [Guo et al. 2013; Kerbl et al. 2015]. Non-monotone plans have been computed for disassembling complex mechanical assemblies [Masehian and Ghandi 2020] and designing high-level interlocking puzzles [Chen et al. 2022]. Non-coherent plans have been computed for solving 2-piece disentanglement puzzles [Zhang et al. 2020]. Following this line of research, we study 3D assemblies with non-sequential plans that require coordinate motion to assemble all the parts in one shot.

Coordinate-motion assemblies. Prior work on coordinate-motion assemblies primarily focuses on computing assembly or disassembly plans. Schwarzer [Schwarzer and Schweikard 2002] showed that finding non-sequential plans for 2D polygonal assemblies is NP-hard. For polyhedral assemblies, early studies restricted motions to translations [Schwarzer et al. 1998; Schweikard and Schwarzer 1997], while recent work incorporates rotations [Wang et al. 2019]. Yet, verifying whether an assembly possesses a *unique coordinate motion* while accounting for rotations remains unknown.

Coordinate-motion assemblies have been explored across diverse applications, including the design of mechanical puzzles [Coffin 2007], decomposition for 3D printing [Araújo et al. 2019], theoretical investigations [Houle and Toussaint 2015, 2017; Snoeyink and Stolfi

1994], and the construction of architectural structures [Glath et al. 2023a; Rossi et al. 2024].

Notable design methods exist in the domains of puzzles and architecture. Puzzle artisans typically employ an exhaustive search approach, decomposing polyhedra into elemental shapes such as spheres [Bell 2011] or tetrahedra [Bell 2012, 2014], and recombining them to form puzzles. In architecture, Glath et al. utilized hodographs to represent translational coordinate motions, realizing them via mortise-and-tenon joints [Glath et al. 2023b]. However, adapting these joints to curved global shapes forces them to be short and narrow, resulting in weak connections; see Fig. 13 (right). In contrast, we investigate the relationship between geometry and unique coordinate motion to design assemblies with global planar contacts rather than local joints; see Fig. 2. Unlike prior designs where multiple parts may move as a rigid subassembly during disassembly, our approach guarantees a unique disassembly motion.

Interlocking assemblies. An assembly is called an interlocking assembly if disassembling it requires first moving a single key part, with all others fixed [Wang et al. 2018]. Because of the stability brought by interlocking, many research works have designed interlocking assemblies for different applications, including mechanical puzzles [Song et al. 2012; Xin et al. 2011], 3D printed objects [Song et al. 2015; Yao et al. 2017a], furniture [Fu et al. 2015; Song et al. 2017], architecture [Wang et al. 2019], and robotic assembly [Zhang and Balkcom 2016]. Other works [Ganeshan et al. 2025; Yao et al. 2017b] designed integral joints for making interlocking assemblies.

Interlocking assemblies are stable because fixing the key part and any other part leads to deadlock of the entire assembly. Similarly, coordinate-motion assemblies are deadlocked when *any* two parts are fixed. Consequently, coordinate-motion assemblies offer high stability without relying on a specific, potentially looser, key part. Like interlocking assemblies, coordinate-motion assemblies are applicable to furniture [Glath et al. 2020] and architectural structures [Glath et al. 2023a], as well as puzzles [Coffin 2007].

The primary distinction lies in the disassembly sequence: interlocking assemblies typically follow a strictly sequential removal process, whereas coordinate-motion assemblies are non-sequential. Due to the non-sequential nature, coordinate-motion assemblies are well suited to robotic assembly [Rossi et al. 2024], since multiple robots can work simultaneously to assemble different parts, improving assembly efficiency.

3 Theory of Coordinate-Motion Assemblies

This section presents a theoretical analysis of coordinate-motion assemblies. We analyze the relation between contact geometry and unique coordinate motion, and propose an algorithm for testing unique coordinate motion based on this analysis.

3.1 Definitions

Unique disassembly motion. We study assemblies of polyhedral parts $\mathcal{A} = \{P_i\}_{i=1}^{K}$. Denote the infinitesimal rigid motion of a part P_i as $\mathbf{Y}_i = [\mathbf{t}_i^\intercal, \boldsymbol{\omega}_i^\intercal]^\intercal$, where $\mathbf{t}_i$ is the translational velocity and $\boldsymbol{\omega}_i$ is the angular velocity. The collision-free constraint at point-plane contact c with position $\mathbf{r}_c$ and normal $\mathbf{n}_c$ is described by the inequality $\mathbf{n}_c \cdot [(\mathbf{t}_i - \mathbf{t}_j) + (\boldsymbol{\omega}_i - \boldsymbol{\omega}_j) \times \mathbf{r}_c] \geq 0$; see Fig. 3. A contact

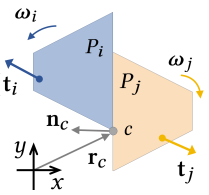

Fig. 3. Parts P_i and P_j have a point-plane contact c.

between two planar polygonal faces can be equivalently represented by point-plane constraints at the vertices of the contact region's convex hull [Wilson and Matsui 1992]. Setting $\mathbf{Y}_K = \mathbf{0}$ to remove the rigid-body motion of the whole assembly, all the contacts give a system of linear inequalities [Wang et al. 2019]:

$$\mathbf{B} \cdot \mathbf{Y} \geq \mathbf{0}, \quad \text{s.t. } \mathbf{Y} \neq \mathbf{0}, \tag{1}$$

where $\mathbf{Y} = [\mathbf{Y}_1^\intercal, \ldots, \mathbf{Y}_{K-1}^\intercal]^\intercal$ is the stacked generalized velocity of the assembly, and $\mathbf{B}$ is the contact matrix of m rows and n columns. Denote $W = \{\mathbf{Y} \in \mathbb{R}^n | \mathbf{B} \cdot \mathbf{Y} \geq \mathbf{0},\ \mathbf{Y} \neq \mathbf{0}\}$ as the set of feasible generalized velocities. We say that the assembly has *unique disassembly motion* if there exists $\mathbf{Y}^* \in \mathbb{R}^n$ s.t. $W = \{\lambda \mathbf{Y}^* | \lambda > 0\}$; see Fig. 2 (a) for an example on a 2D assembly.

Contact-based coordinate-motion assembly. The coordinate-motion assembly we study is defined as:

Definition 3.1 (Coordinate-motion assembly). An assembly is a coordinate-motion assembly if it satisfies:

(1) $W \neq \emptyset$;
(2) if $\mathbf{Y} \in W$, then $\mathbf{Y}_i \neq \mathbf{Y}_j$, $\forall\ i \neq j$.

We study contact-based assemblies in which the connections between parts are formed by planar contacts rather than standard joints. The relative motion space between two parts can be a motion cone rather than a single line. Such coordinate-motion assemblies tend to have multiple disassembly solutions; see Fig. 2 (b). This raises the question of what contact geometry the assembly should have to ensure unique coordinate motion.

3.2 Relation Between Contact Geometry and Unique Coordinate Motion

This section studies the inverse problem: given a desired unique coordinate motion $\mathbf{Y}^*$, what assembly geometry achieves it? We observe that contacts fall into three categories during disassembly: those that remain in contact, those that separate instantaneously, and those that block undesired motions; see Fig. 4. We formalize this classification as follows:

Definition 3.2 (Contact classification). Let $\mathbf{Y}^* \in \mathbb{R}^n$ and matrix $\mathbf{B} = (\mathbf{b}_1^\intercal, \ldots, \mathbf{b}_m^\intercal)^\intercal \in M_{m \times n}(\mathbb{R})$. Classify the row indices of $\mathbf{B}$ as:

- $I_{\text{sld}} = \{i | \mathbf{b}_i \mathbf{Y}^* = 0\}$ (sliding contacts);
- $I_{\text{spt}} = \{i | \mathbf{b}_i \mathbf{Y}^* > 0\}$ (separating contacts);
- $I_{\text{blk}} = \{i | \mathbf{b}_i \mathbf{Y}^* < 0\}$ (blocking contacts).

We call $\{I_{\text{sld}}, I_{\text{spt}}, I_{\text{blk}}\}$ the *contact classification* of velocity $\mathbf{Y}^*$.

The classification is well-defined because I_{sld}, I_{spt}, and I_{blk} are pairwise disjoint, and their union is $\{1, 2, \ldots, m\}$. The sliding contacts in I_{sld} confine the disassembly motion to a lower-dimensional

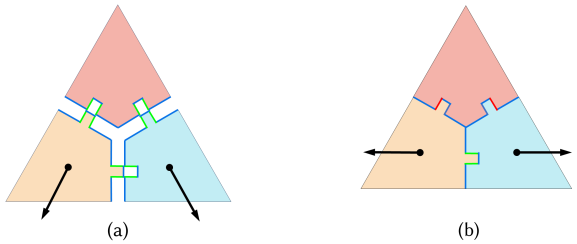

Fig. 4. Different intended disassembly motions give different contact classifications. With the red part fixed, the remaining parts are pulled along two directions (a&b). The same contact may behave differently under different motions. We classify contacts as sliding (in green), separating (in blue), or blocking (in red).

subspace. If they completely block all motions orthogonal to $\mathbf{Y}^*$, the feasible motions are confined to $\{\lambda\mathbf{Y}^*|\lambda \in \mathbb{R}\}$, yielding a unique disassembly motion. This observation leads to the following equivalent condition for a unique disassembly motion:

Proposition 3.3 (Unique disassembly conditions). *Let vector* $\mathbf{Y}^* \in \mathbb{R}^n$, *matrix* $\mathbf{B} = (\mathbf{b}_1^\intercal, \ldots, \mathbf{b}_m^\intercal)^\intercal \in M_{m\times n}(\mathbb{R})$, *set* $W = \{\mathbf{Y} \in \mathbb{R}^n | \mathbf{BY} \geq \mathbf{0}, \mathbf{Y} \neq \mathbf{0}\}$, *and* $\{I_{sld}, I_{spt}, I_{blk}\}$ *is the contact classification of* $\mathbf{Y}^*$. *The following two statements are equivalent:*

(1) $W = \{\lambda\mathbf{Y}^*|\lambda > 0\}$;
(2) The following conditions hold:
  (a) $\forall\, \mathbf{v} \in \mathbf{Y}^{*\perp} \setminus \{\mathbf{0}\}, \exists j \in I_{sld}, \mathbf{b}_j\mathbf{v} < 0$ *(sliding condition);*
  (b) $I_{spt} \neq \emptyset$ *(separating condition);*
  (c) $I_{blk} = \emptyset$ *(blocking condition).*

Please refer to the supplementary material for a general version of Proposition 3.3 and its proof. We provide a geometric interpretation of this proposition. W is a polyhedral cone, and unique disassembly motion means that this cone consists of a single extreme ray. Let $\text{span}(W)$ denote the minimal linear subspace containing W. We interpret $\dim(\text{span}(W))$ as the *disassembly degree of freedom* (DOF). In the case of unique coordinate motion, $\dim(\text{span}(W)) = 1$. The sliding condition implies that the sliding contacts in I_{sld} must completely block all motions within the orthogonal complement space $W^\perp$. By doing so, they reduce the disassembly DOFs by exactly $\dim(W^\perp)$. The separating condition ensures that contacts in I_{spt} block valid motions in the reverse direction (i.e., motions in $\text{span}(W) \setminus W$). Finally, the blocking condition requires I_{blk} be empty; otherwise, these contacts would prevent the intended motion $\mathbf{Y}^*$.

We give an example to illustrate the above proposition.

Example 3.4. Two parts P_1 and P_2 are in $\mathbb{R}^2$ with three flat contacts with normals $\mathbf{n}_1$, $-\mathbf{n}_1$, and $\mathbf{n}_2$, where $\mathbf{n}_2 \perp \mathbf{n}_1$; see Fig. 5. For simplicity, we ignore rotational motion. Fix P_2 (i.e., $\mathbf{Y}_2 = \mathbf{0}$), and Eq. (1) yields:

$$\mathbf{BY} = \begin{bmatrix} \mathbf{n}_1^\intercal \\ -\mathbf{n}_1^\intercal \\ \mathbf{n}_2^\intercal \end{bmatrix} \mathbf{Y}_1 \geq \mathbf{0}, \quad \text{s.t. } \mathbf{Y}_1 \neq \mathbf{0}. \tag{2}$$

Let $\mathbf{Y}^* = \mathbf{n}_2$. Then $\{\lambda\mathbf{Y}^* \mid \lambda > 0\}$ is the complete solution set. The contact classification satisfies the conditions in Proposition 3.3:

(a) $I_{\text{sld}} = \{1, 2\}$: For any $\mathbf{v} \in \mathbf{Y}^{*\perp} \setminus \{\mathbf{0}\}$, we have $\mathbf{v} \parallel \mathbf{n}_1$, so $\mathbf{n}_1\mathbf{v} < 0$ or $-\mathbf{n}_1\mathbf{v} < 0$;

Algorithm 1: Algorithm to verify if an assembly $\mathcal{A}$ possesses a unique coordinate motion.

Input: assembly $\mathcal{A}$
Output: "True" iff $\mathcal{A}$ has a unique coordinate motion

```
B ← ComputeContactMatrix(A)                    // Eq. (1)
Y* ← DeadlockTest(B)                           // Linear program (4)
if Y* = 0 then
    return False                               // Assembly is in deadlock
if ∃ i ≠ j s.t. Y*_i = Y*_j then
    return False                               // Not coordinate motion
(I_sld, I_spt) ← ClassifyContacts(B, Y*)       // Def. 3.2
B̃ ← ProjectContacts(I_sld, B, Y*^⊥)            // Eq. (3)
ṽ ← DeadlockTest(B̃)
if ṽ ≠ 0 then
    return False                               // Sliding condition fails
if I_spt = ∅ then
    return False                               // Separating condition fails
return True
```

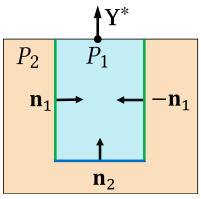

Fig. 5. Illustration of Example 3.4. When fixing part P_2, part P_1 can only move along the direction of $\mathbf{Y}^*$.

(b) $I_{\text{spt}} = \{3\}$;
(c) $I_{\text{blk}} = \emptyset$.

The two sliding contacts block all motions orthogonal to $\mathbf{Y}^*$, reducing the disassembly degree of freedom to one. The separating contact blocks motions opposite to $\mathbf{Y}^*$, i.e., $\{\lambda\mathbf{Y}^*|\lambda < 0\}$.

This proposition provides a design principle: given a desired unique coordinate motion $\mathbf{Y}^*$, the assembly geometry must satisfy the three conditions in Proposition 3.3. Our design algorithm searches for contact geometries that satisfy these conditions.

3.3 Unique Coordinate Motion Test

An assembly has unique coordinate motion if and only if it has a unique disassembly solution with distinct part motions. Algorithm 1 tests whether a given assembly has a unique coordinate motion. It proceeds in three steps. First, we compute one disassembly motion $\mathbf{Y}^*$ by solving Eq. (1) using the linear program (4) (discussed later). Second, we verify that $\mathbf{Y}^*$ represents coordinate motion by checking that all parts move distinctly: $\mathbf{Y}_i^* \neq \mathbf{Y}_j^*$ for all $i \neq j$. Third, we verify uniqueness by enforcing the three conditions from Proposition 3.3: the separating condition is checked explicitly, and the blocking condition is satisfied by definition since $\mathbf{Y}^*$ is a valid motion. The

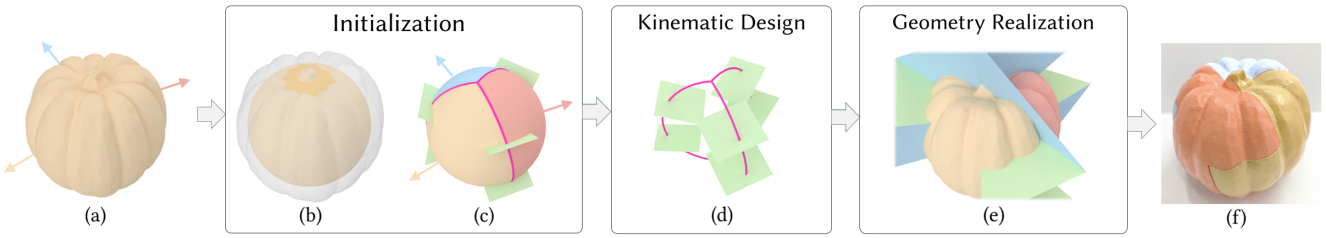

Fig. 6. Overview of our approach. (a) Given a target global shape and a desired coordinate motion, (b) we place a reference sphere inside the shape, and (c) partition it according to the prescribed coordinate motion to initialize an abstract contact model (Sec. 5.1). (d) Next, we optimize the abstract contact model to enforce unique coordinate motion (Sec. 5.2). (e) Finally, we construct full part geometries by clipping with the global shape (Sec. 4.2) and enforcing fabrication constraints (Sec. 5.3). (f) A 3D printed prototype that validates the physical feasibility and motion correctness of the designed assembly.

sliding condition, however, requires verifying deadlock within a specific subspace, which we formulate next.

Sliding condition test. We project the sliding contact constraints into the subspace $\mathbf{Y}^{*\perp}$ as follows. Let $\mathbf{E} \in M_{n\times(n-1)}(\mathbb{R})$ be a basis matrix of $\mathbf{Y}^{*\perp}$, so that any deviation motion $\mathbf{v} \in \mathbf{Y}^{*\perp}$ is parameterized as $\mathbf{v} = \mathbf{E}\tilde{\mathbf{v}}$. Let $\mathbf{B}_{\text{sld}}$ denote the submatrix of $\mathbf{B}$ containing only the sliding contact rows. We define the *projected contact matrix* as $\tilde{\mathbf{B}} = \mathbf{B}_{\text{sld}}\mathbf{E}$. The sliding condition holds if and only if the following reduced system has no non-trivial solution:

$$\tilde{\mathbf{B}}\tilde{\mathbf{v}} \geq \mathbf{0}, \quad \text{s.t. } \tilde{\mathbf{v}} \neq \mathbf{0}. \tag{3}$$

This reduces the sliding condition verification to a standard deadlock test on $\tilde{\mathbf{B}}$.

Deadlock test. Following [Wang et al. 2019], the deadlock test determines whether Eq. (1) admits a non-zero solution by solving:

$$\max \sum_{i=1}^{m} t_i, \quad \text{s.t. } \mathbf{BY} \geq \mathbf{t},\ 0 \leq t_i \leq 1, \tag{4}$$

where vector $\mathbf{t} = (t_1, \ldots, t_m)^\top$. The input to the deadlock test algorithm is the matrix $\mathbf{B}$. If the assembly is in deadlock, the output is $\mathbf{0}$; otherwise, return a non-zero solution $\mathbf{Y}$.

4 Modeling Coordinate-Motion Assemblies

In this section, we present our geometric modeling framework for coordinate-motion assemblies. We first introduce a parametric representation for the contact interfaces, and then describe the procedure for constructing full volumetric parts that match the target global shape.

4.1 Modeling Contact Interfaces

Motivation. Standard polygonal mesh representations are not suitable for designing coordinate-motion assemblies because kinematic feasibility is governed by contacts rather than vertex positions. Specifically, satisfying the unique disassembly conditions (Proposition 3.3) requires multiple sliding contacts with normals precisely perpendicular to relative velocity vectors. Enforcing such constraints by direct vertex manipulation is ineffective and prone to numerical instability. Instead, we employ a parametric model that explicitly defines sliding contact faces as the primary design variables, effectively decoupling kinematic constraints from the global shape.

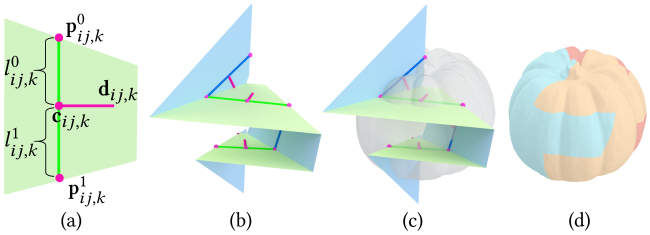

Fig. 7. Geometry modeling. (a) Sliding contact faces are parameterized by line segments (in green). (b) Connecting faces (in blue) join them into a continuous sequence that forms the closed contact interface between a part pair. (c) We cut the global shape (in gray) with the volumes derived from these faces to obtain (d) the final part geometries.

Contact representation. We represent the sliding contacts between two parts P_i and P_j as a set of n_{ij} planar faces derived from offsetting a set of line segments, indexed by $k = 1, \ldots, n_{ij}$. The geometry is defined by:

- *Sliding line segments.* We define a set of segments $\ell_{ij,k}$, each parametrized by a center point $\mathbf{c}_{ij,k}$ and extents $l^0_{ij,k}, l^1_{ij,k} > 0$. The segment direction is aligned with the relative translational velocity $\mathbf{t}_{ij}$; see Fig. 7 (a). This construction guarantees that any plane containing this segment automatically produces sliding contacts.
- *Offset direction.* Each segment has an offset direction $\mathbf{d}_{ij,k}$ used to generate the planar face. This direction lies in the face and is orthogonal to the face normal. This vector is orthogonal to $\mathbf{t}_{ij}$ and is parameterized by an angle $\theta_{ij,k}$ around the axis of $\mathbf{t}_{ij}$.

Loop generation. Because the sliding segments are parallel to each other, they are spatially disjoint. Consequently, the sliding faces alone do not form a continuous interface between parts. We therefore introduce *connecting segments* (shown in blue in Fig. 7 (b)) that bridge the gaps between sliding segments. Together with the sliding segments, they form a closed polyline loop for each part near the target shape surface. Unlike sliding faces, the orientation of the faces derived from connecting segments is flexible; they need only avoid violating the blocking condition. If a connecting face violates this condition, it can typically be rotated to become a sliding face without introducing interferences.

4.2 Modeling Full Part Geometry

Once the contact face loops are determined, we generate the final volumetric geometry in two steps:

(1) *Primitive construction.* We construct a watertight primitive for each part by connecting the inward-facing edges of the contact face loop to a central point C_{ref}, and extending the outward-facing edges along the disassembly direction far enough to ensure complete enclosure.
(2) *Shape integration.* The final part geometry is obtained by computing the Boolean intersection between these constructed primitives and the target global shape mesh; see Fig. 7 (d).

5 Designing Coordinate-Motion Assemblies

We aim to design physically feasible coordinate-motion assemblies, taking the user-specified global shape and part moving directions as inputs. A straightforward approach would be to directly search the geometric parameters, constructing parts and evaluating their motion at each step. However, this is computationally inefficient due to the high cost of constructing watertight geometry, and ineffective as direct search rarely converges to valid solutions. Instead, we propose a two-stage strategy decoupling kinematic constraints from geometry realization; see Fig. 6. First, the kinematic design stage optimizes a set of *contact proxies* to satisfy the sliding condition, avoiding expensive mesh operations. Second, the geometry realization stage constructs full parts only for kinematically valid candidates.

5.1 Initialization

Reference sphere. Because coordinate motion relies on internal contacts rather than the appearance of parts, we embed a sphere inside the input mesh as reference, similar to [Xin et al. 2011]; see Fig. 6 (b). The default offset direction of contact faces is from line segment centers to the sphere center. The central point C_{ref} used in volume closure is also set as the sphere center. This sphere only serves as an internal proxy for initializing contacts and volume closure. It does not restrict the target shape, although very complex inputs may still lead to realized parts with disconnected components and/or fragile features.

Contact initialization. We determine the assembly contact graph (nodes represent parts and edges represent contact interfaces) by partitioning the reference sphere based on disassembly directions. We define the halfspace for part P_i relative to P_j as $H_{ij} = \{\mathbf{x} \in \mathbb{R}^3 | (\mathbf{t}_i - \mathbf{t}_j) \cdot \mathbf{x} \geq 0\}$. A suitable placement region for part P_i is the convex cone $F_i = \bigcap_{j \neq i} H_{ij}$. They form a partition of $\mathbb{R}^3$: $\mathring{F}_i \bigcap \mathring{F}_j = \varnothing$ for all $i \neq j$, and $\bigcup_i F_i = \mathbb{R}^3$. Although our formulation allows different velocity magnitudes across parts, we use the same magnitude in all our results. This avoids empty cone partitions except in extreme cases. Intersecting $\{F_i\}$ with the reference sphere yields a partition of the sphere surface; see Fig. 6 (c). The adjacent partition regions define the contact graph edges. We initialize $\mathbf{c}_{ij,k}$ evenly along the boundary curves γ_{ij}.

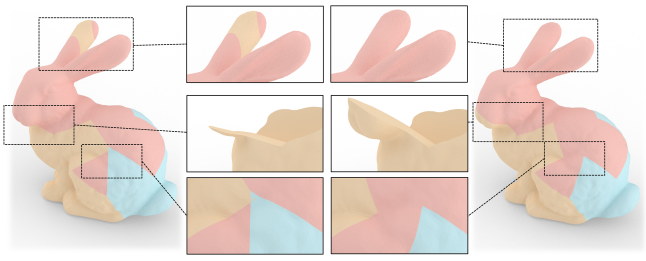

Fig. 8. Common fabrication issues (left) and their post-processing results (right). In each zoomed view: (top) a disconnected component (a bunny ear) is merged into the main body; (middle) thin features are thickened to improve structural robustness; (bottom) contact faces are narrowed to eliminate part intersections.

5.2 Kinematic Design

In the kinematic design stage, we search for contact face parameters that satisfy the constraints for unique coordinate motion without constructing watertight parts. We validate successful candidates in the geometry realization stage.

Contact proxies. Because unique coordinate motion depends primarily on sliding contacts, we avoid expensive CSG operations by approximating sliding faces as square patches (contact proxies); see Fig. 6 (d). We offset each line segment $\ell_{ij,k}$ to obtain four corner points. Let $\mathbf{p}^0_{ij,k}$ and $\mathbf{p}^1_{ij,k}$ denote the segment endpoints. The four corner points are defined as:

$$\mathbf{q}^{\zeta,\eta}_{ij,k} = \mathbf{p}^{\zeta}_{ij,k} + w_\eta \mathbf{d}_{ij,k}, \quad \zeta, \eta \in \{0, 1\}, \tag{5}$$

where $w_0 \leq 0$ and $w_1 \geq 0$ control the face width. Given the reference sphere radius R, we set $w_0 = -0.1R$, $w_1 = 0.3R$ for a conservative estimate. The projected contact matrix $\tilde{\mathbf{B}}$ in Eq. (3) is then constructed from the corner points and square patch normals.

Optimization. Since the sliding condition is equivalent to the deadlock condition,we use a deadlock measure as the optimization objective. Previous work has assessed motion infeasibility either over restricted motion sets [Whiting et al. 2009] or over all motions with high computational cost [Chen et al. 2024]. We use the optimal objective value of the linear program (4) as our deadlock measure, denoted by $E_{\text{kinematic}}$. When a candidate satisfies the target motion constraints, $E_{\text{kinematic}} = 0$. We solve:

$$\min_{\ell_{ij,k}, \mathbf{d}_{ij,k}, n_{ij}} E_{\text{kinematic}}(\tilde{\mathbf{B}}), \tag{6}$$

where the contact matrix $\tilde{\mathbf{B}}$ is constructed from the contact proxies for each part pair (P_i, P_j). The contact proxies move freely in the space: segment centers $\mathbf{c}_{ij,k}$ are constrained to the curve γ_{ij} (parameterized by $x_{ij,k} \in [0, 1]$), offset directions $\mathbf{d}_{ij,k}$ are parameterized by angles $\theta_{ij,k} \in [-\pi, \pi)$, and the extents $l^0_{ij,k}, l^1_{ij,k}$ vary within $[l_{\min}, l_{\max}]$; in all experiments we set $l_{\min} = 0.0$ and $l_{\max} = 0.4R$.

The discrete variable n_{ij} (the number of sliding contact faces) is coupled with the continuous parameters. We first fix n_{ij} and optimize the continuous parameters using differential evolution [Storn and Price 1997] with multiple random seeds to explore the design space. A candidate is valid if $E_{\text{kinematic}} = 0$ and $\tilde{\mathbf{B}}$ has full rank. If

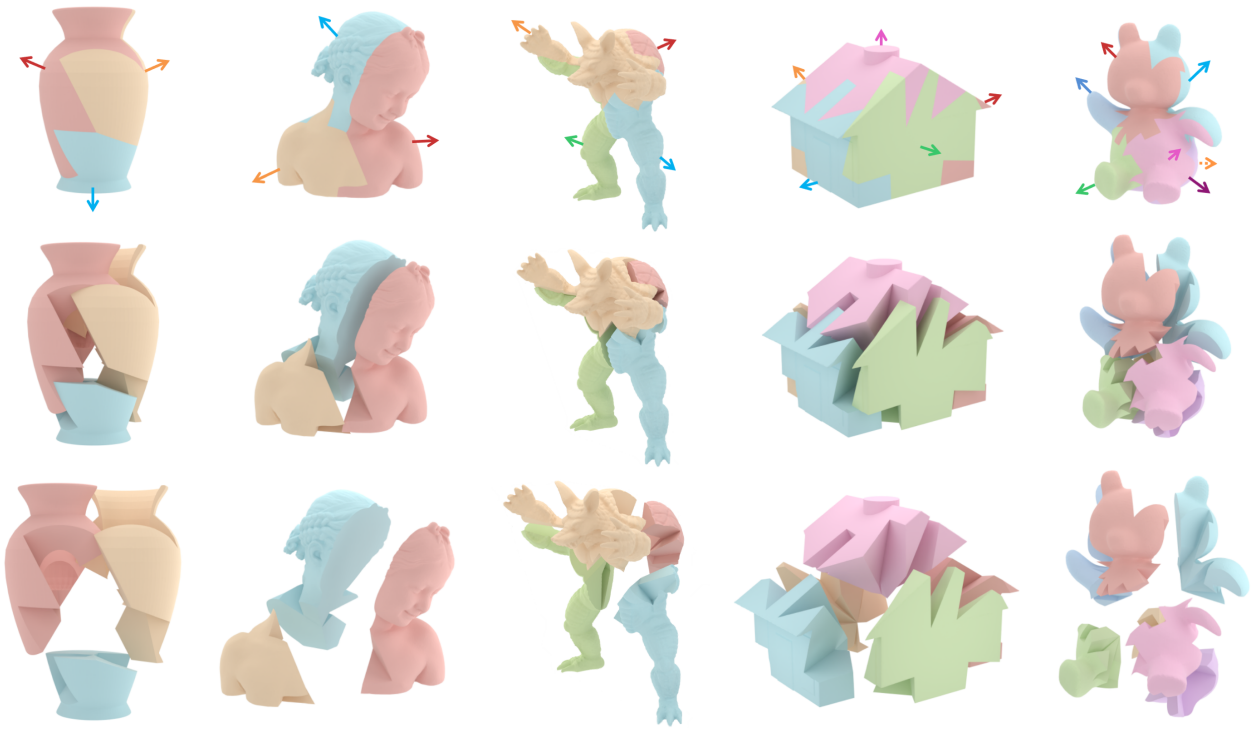

Fig. 9. Our approach allows modeling coordinate-motion assemblies with various shapes.

no valid candidate is found, we rerun the optimization with increased n_{ij}. This adds degrees of freedom to the contact model by introducing more sliding contacts, reducing the chance of failure.

5.3 Geometry Realization

In the geometry realization stage, we take a kinematically valid candidate and construct full part geometry. The constructed parts must satisfy common fabrication constraints; see Fig. 8.

(1) *Connectivity.* Each part must form a single connected component. We compute connected components of each part after clipping by the global shape, and merge small disconnected components into the nearest large component.
(2) *Collision.* Each part must be a valid watertight surface without self-intersections and must not intersect any other part. Intersections usually come from overly wide contact faces (large $l_{ij,k}$) or excessive segment offsets; see Fig. 8 (bottom). We maximize offsets by binary search under collision avoidance, then shrink face widths and reconstruct local geometry when needed.
(3) *Fragility.* Each part must avoid thin features that are fragile to fabricate. Various approaches have been proposed to address fragility in assembly design [Chen et al. 2015; Wang et al. 2021a]. We detect thin regions using the fragility test of [Luo et al. 2012], adding two checks specific to our contact model: (i) small dihedral angles between adjacent contact faces, and (ii) contact faces that lie too close to each other. We use a thickness threshold τ_d set to 2% of the global bounding box diagonal, and an angle threshold $\tau_a = 18^\circ$. When a violation is detected, we locally adjust the contact faces by translating faces to increase thickness or widening dihedral angles, until it passes the fragility test or the candidate is rejected.

Final kinematic validation. When these fabrication constraints are satisfied, we perform kinematic validation with algorithm 1. If the current candidate fails this test due to geometric adjustments, it is discarded. If all candidates fail the test, we return to the kinematic design stage to search for new solutions while decreasing $l_{\max}$ to restrict the search space to geometrically safer regions.

6 Results

We implement our approach in C++ using libigl [Jacobson et al. 2018] and CGAL [The CGAL Project 2025] for geometry processing, and ALGLIB [Bochkanov 2025] for optimization. All experiments run on a laptop with a 2.5 GHz CPU and 16 GB RAM.

Results. Our tool successfully processes a diverse range of geometries, handling shapes with complex surfaces (e.g., Bimba, Armadillo), objects with deep concave cavities (Vase), and architectural models with sharp features (House); see Fig. 9. It also accommodates higher genus topologies, such as the Fertility model (genus 4); see Fig. 10. Furthermore, our method scales effectively to assemblies with a high part count, generating a fifteen-piece Sphere that maintains a unique coordinate motion; see Fig. 11. Fig. 12 shows detailed disassembly sequences for three examples; please refer to the accompanying video for additional animations. Table 1 summarizes the statistics for the results shown in the paper.

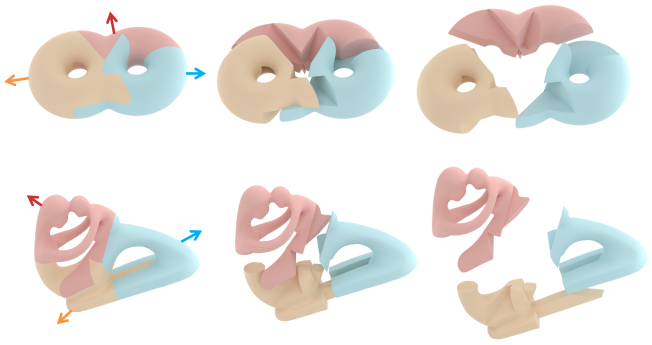

Fig. 10. Our approach can generate coordinate-motion assemblies with higher-genus shapes, including global surfaces of genus 2 (left) and genus 4 (right).

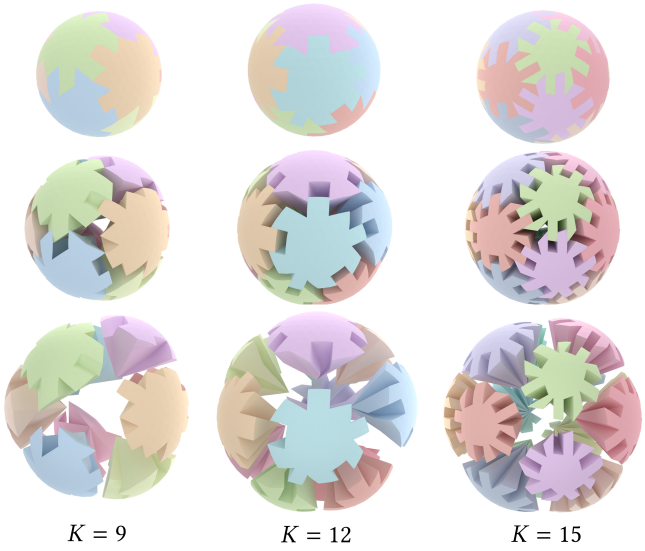

Fig. 11. We evaluate the scalability of our approach by generating coordinate-motion assemblies with varying numbers of parts (up to 15).

Comparison with joint-based assemblies. We model a coordinate-motion assembly for the Dog Bowl model by decomposing the shape into parts and adding mortise-and-tenon joints; see Fig. 13 (right). Because a linear joint is constrained to lie along a straight line segment, its length is severely limited by the global circular shape. Moreover, the thin wall of the bowl restricts the joint width, resulting in tiny joints. In contrast, our result (Fig. 13 (left)) utilizes the entire interface surface for contact. Compared to the joint-based design, our contact-based design has a much larger total sliding contact area (14.7 vs. 3.1), and yields more structurally sound parts by avoiding thin and weak joints. Finally, joints often require precise alignment to avoid jamming, which is difficult in practice.

User guidance. Without user guidance, our method may place part boundaries in regions the user wants to preserve. To address this, we allow the user to define protected regions. For each adjacent part pair (P_i, P_j), we compute the intersection curve C_{ij} between the contact interfaces and the protected area, and use its length $l(C_{ij})$ to quantify the overlap. We then add a feature preservation constraint:

$$\sum_{(i,j)\in E} l(C_{ij}) = 0, \tag{7}$$

where E is the part contact graph edge set. We then solve the optimization problem (6) using an $L2$ penalty. This discourages cuts through protected regions; see Fig. 14.

Table 1. Statistics for our designed results. Columns 3–8 report the number of parts (K), the maximum number of sliding contact faces between any part pair ($M = \max\{n_{ij}\}$), initialization time (T_I), kinematic design time (T_K), geometry realization time (T_R), and total time (T_Σ).

Fig.	Shape	K	M	Time (min)			
				T_I	T_K	T_R	T_Σ
9	Vase	3	2	0.04	0.20	0.81	1.05
	Bimba	3		0.02	0.08	0.11	0.21
	Armadillo	4		0.04	0.19	0.26	0.49
	House	5		0.06	1.73	1.12	2.91
	Teddy	7		0.07	0.25	0.14	0.46
10	Double Torus	3	2	0.02	0.11	0.07	0.20
	Fertility			0.03	0.58	0.10	0.71
11	Sphere	9	2	0.12	0.44	1.05	1.61
		12		0.17	0.56	0.24	0.97
		15	3	0.18	1.96	0.37	2.51
12	Gargoyle	4	2	0.16	3.45	3.67	7.28
	Cow	5		0.07	8.78	0.99	9.84
	Duck	8		0.06	0.24	1.59	1.89
14	Max Planck (left)	4	3	0.04	1.39	0.07	1.50
	Max Planck (right)			0.05	39.31	0.06	39.42
15	Sphere	6	2	0.08	1.89	0.14	2.11
	Torus	3	2	0.05	3.49	0.02	3.56
	Pumpkin			0.06	1.33	0.04	1.43
	Bunny		3	0.07	0.50	0.38	0.95
	Dog Bowl			0.05	0.54	1.21	1.80
16	Sphere	4	2	0.02	1.81	0.01	1.84
			3	0.02	1.07	0.01	1.10
			4	0.02	2.87	0.02	2.91

Ablation study. We study the effect of M, the maximum number of sliding contact faces per part pair (Fig. 16). Larger M enlarges the feasible set for satisfying uniqueness. In this example, $M = 1$ fails, $M = 3$ is fastest (1.07 min vs. 1.81 for $M = 2$). When $M = 4$, the feasible set becomes even larger, but the higher-dimensional search space slows optimization to 2.87 min. Overall, in this example, $M = 3$ offers a good trade-off between design feasibility and optimization cost.

Fabrication. We fabricate five assemblies of our design using SLA 3D printing; see Fig. 15. All prototypes assemble successfully and require the precise, designed coordinate motions to disassemble. In the Torus example, individual part pairs have at most two sliding contact faces ($M = 2$), implying non-unique pairwise motions. Nevertheless, the full assembly enforces a unique disassembly motion due to the global contact design. We also fabricate models with $M = 3$ and observe that they exhibit greater stability during the sliding phase. We perform stability tests on our six-piece Sphere

Fig. 12. We show the disassembly of three results via coordinate motion of the component parts.

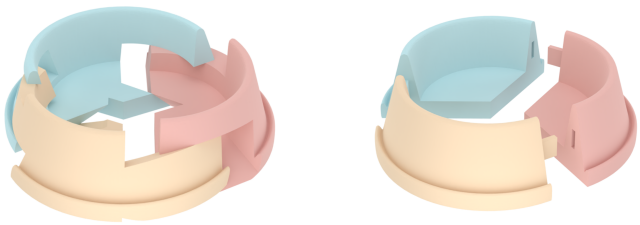

Fig. 13. Comparison between our contact-based design (left) and a joint-based design (right) for Dog Bowl. Adapting joints to the rounded global shape forces them to be short and narrow, while our approach uses global contact over the whole shape.

Fig. 14. Users can select a protected region (middle) to preserve salient features during part decomposition. Compared to the result without user guidance (left), our guided design better preserves the human facial features (right).

and three-piece Pumpkin assemblies. Both prototypes demonstrate robust structural stability, remaining fully intact when thrown and successfully withstanding the impact shocks from catching or dropping. The six-piece Sphere assembly has a convex global shape, making manual disassembly difficult. Instead, we demonstrate disassembly by spinning the sphere rapidly on a table; the centrifugal force pulls all parts outward simultaneously, separating the assembly along the unique coordinate motion. Please refer to the accompanying video for these physical results.

7 Conclusion

This paper studies the design of contact-based coordinate-motion assemblies that have a unique coordinate motion and match a global shape specified by users. We derive theoretical conditions that guarantee uniqueness of the disassembly motion and present a corresponding test algorithm for unique coordinate motion. Building on the results, we introduce a two-stage strategy that models contact-based geometry and optimizes it to satisfy the conditions for unique coordinate motion. We then generate full part geometries that realize the optimized contact faces while satisfying fabrication constraints

Fig. 15. The disassembly sequence of our 3D-printed results. The fabricated prototypes confirm that our designs realize the intended disassembly via unique coordinate motion.

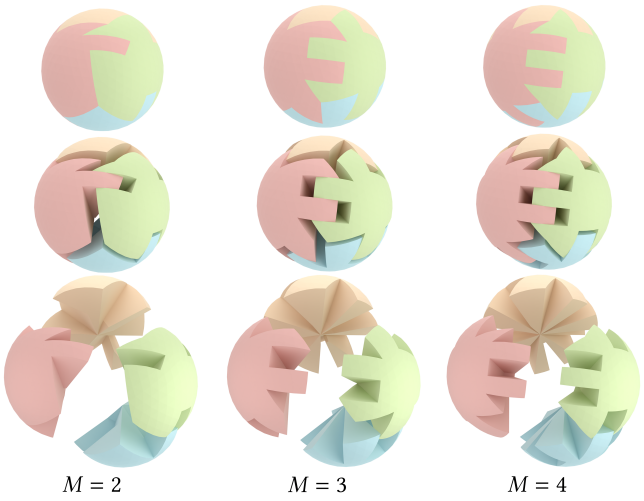

Fig. 16. Ablation study on the maximum number of sliding contact faces between any part pair (M). Larger M provides more sliding contacts, which not only enlarges the design space but also increases the shape complexity of the parts.

and matching the target shape. Our approach supports a wide range of target shapes, part counts, and assembly topologies while preserving the intended motion behavior. We validate our method through extensive experiments and by fabricating physical prototypes that demonstrate correct motion and structurally stable assemblies.

Limitations and future work. Our work has several limitations that open up interesting directions for future research. First, our theoretical analysis relies on infinitesimal rigid motions; hence, an assembly might lose the unique coordinate motion property after a small finite displacement. Future work could test uniqueness at sampled disassembly states to quantify robustness under finite motion. Second, our strongest guarantees hold for the abstract contact model; geometry realization may alter a kinematically valid candidate, so the final geometry must be revalidated. Future work could characterize and reduce this gap. Third, we do not analyze assembly stability in terms of physical forces and torques. Incorporating physics-based analysis, such as using the minimum friction required for stability as a quality metric, would allow for optimizing resistance against unintended separation forces.

Acknowledgments

We thank the reviewers for their valuable comments. This work was supported by the Singapore MOE AcRF Tier 2 Grants (MOE-T2EP20222-0008, MOE-T2EP20123-0016), Laoshan Laboratory (No. LSKJ202300305) and the National Natural Science Foundation of China (62025207).

References

Chrystiano Araújo, Daniela Cabiddu, Marco Attene, Marco Livesu, Nicholas Vining, and Alla Sheffer. 2019. Surface2Volume: Surface Segmentation Conforming Assemblable Volumetric Partition. *ACM Trans. on Graph. (SIGGRAPH)* 38, 4 (2019), 1:1–1:16.
George Bell. 2011. Sphere Octahedron Puzzles. *Cubism For Fun* (2011). Article No. 84.
George Bell. 2012. More Icosahedron Puzzles. *Cubism For Fun* (2012). Article No. 87.
George Bell. 2014. The Pennyhedron Revisited. *Cubism For Fun* (2014). Article No. 93.
George Bell. 2024. An Initial Attempt at a Mathematical Treatment of Translational Coordinate-Motion Puzzles. In *Proceedings of Bridges 2024: Mathematics, Art, Music, Architecture, Culture.* Tessellations Publishing, Phoenix, Arizona, 187–194.
Sergey Bochkanov. 2025. ALGLIB. www.alglib.net.

Ke Chen, Siqi Li, Peng Song, Jianmin Zheng, and Ligang Liu. 2024. mpcMech: Multi-Point Conjugation Mechanisms. *ACM Trans. on Graph. (SIGGRAPH Asia)* 43, 6 (2024), 211:1–211:14.

Rulin Chen, Ziqi Wang, Peng Song, and Bernd Bickel. 2022. Computational Design of High-level Interlocking Puzzles. *ACM Trans. on Graph. (SIGGRAPH)* 41, 4 (2022), 150:1 – 150:15.

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. on Graph. (SIGGRAPH Asia)* 34, 6 (2015), 213:1–213:12.

Stewart T. Coffin. 2007. *Geometric Puzzle Design.* A. K. Peters.

Mario Deuss, Daniele Panozzo, Emily Whiting, Yang Liu, Philippe Block, Olga Sorkine-Hornung, and Mark Pauly. 2014. Assembling Self-Supporting Structures. *ACM Trans. on Graph. (SIGGRAPH Asia)* 33, 6 (2014), 214:1–214:10.

Chi-Wing Fu, Peng Song, Xiaoqi Yan, Lee Wei Yang, Pradeep Kumar Jayaraman, and Daniel Cohen-Or. 2015. Computational Interlocking Furniture Assembly. *ACM Trans. on Graph. (SIGGRAPH)* 34, 4 (2015), 91:1–91:11.

Aditya Ganeshan, Kurt Fleischer, Wenzel Jakob, Ariel Shamir, Daniel Ritchie, Takeo Igarashi, and Maria Larsson. 2025. MiGumi: Making Tightly Coupled Integral Joints Millable. *ACM Trans. on Graph. (SIGGRAPH Asia)* 44, 6 (2025), 193:1–193:14.

Somayé Ghandi and Ellips Masehian. 2015. Review and Taxonomies of Assembly and Disassembly Path Planning Problems and Approaches. *Computer-Aided Design* 67-68 (2015), 58–86.

Julien Glath, Tristan Gobin, Romain Mesnil, Marc Mimram, and Olivier Baverel. 2020. Design Method for Non-sequential Assembly. In *Proc. IASS Annual Symposium*. 1–11.

Julien Glath, Tristan Gobin, Romain Mesnil, Marc Mimram, and Olivier Baverel. 2023a. Thinking and Designing Reversible Structures with Non-sequential Assemblies. In *Proc. Design Modelling Symposium Berlin*. 249–259.

Julien Glath, Romain Mesnil, Marc Mimram, and Olivier Baverel. 2023b. Non-sequential Assembly of Mortise and Tenon Joints. *Automation in Construction* 154 (2023), 104986:1 – 104986:18.

Jianwei Guo, Dong-Ming Yan, Er Li, Weiming Dong, Peter Wonka, and Xiaopeng Zhang. 2013. Illustrating the Disassembly of 3D Models. *Comp. & Graph. (SMI)* 37, 6 (2013), 574–581.

Michael E. Houle and Godfried T. Toussaint. 2015. Quadrangles Which Cannot Be Separated with Two Hands. In *Proc. the 13th International Conference of Numerical Analysis and Applied Mathematics*. 480021:1–480021:4.

Michael E. Houle and Godfried T. Toussaint. 2017. On the Separability of Quadrilaterals in the Plane by Translations and Rotations. *Contributions to Algebra and Geometry* 58 (2017), 267–276.

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

P. Jiménez. 2013. Survey on Assembly Sequencing: A Combinatorial and Geometrical Perspective. *Journal of Intelligent Manufacturing* 24, 2 (2013), 235–250.

Bernhard Kerbl, Denis Kalkofen, Markus Steinberger, and Dieter Schmalstieg. 2015. Interactive Disassembly Planning for Complex Objects. *Comp. Graph. Forum (Eurographics)* 34, 2 (2015), 287–297.

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

Smaragda Markaki and Costas Panagiotakis. 2023. Jigsaw Puzzle Solving Techniques and Applications: A Survey. *Visual Computer* 39 (2023), 4405–4421.

Ellips Masehian and Somayé Ghandi. 2020. ASPPR: A New Assembly Sequence and Path Planner/Replanner for Monotone and Nonmonotone Assembly Planning. *Computer-Aided Design* 123 (2020), 102828:1–102828:22.

Andrea Rossi, Romain Mesnil, Julien Glath, Mathieu Tissot, Morgane Sanquer, Hyo Wook Kim, Olivier Baverel, and Philipp Eversmann. 2024. Robotic Non-sequential Interlocking Assemblies: Computational Design and Multi-robot Synchronous Assembly of Reversible Structures. In *Proc. Design Modelling Symposium Berlin*. 434–444.

Fabian Schwarzer, Florian Bieberbach, Achim Schweikard, and Leo Joskowicz. 1998. Efficiently Testing for Unboundedness and m-Handed Assembly. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. 1164–1169.

Fabian Schwarzer and Achim Schweikard. 2002. On the Complexity of One-shot Translational Separability. *Inform. Process. Lett.* 83, 4 (2002), 187–194.

Achim Schweikard and Fabian Schwarzer. 1997. General Translational Assembly Planning. In *Proc. IEEE International Conference on Robotics and Automation*. 612–619.

Jack Snoeyink and Jorge Stolfi. 1994. Objects that Cannot Be Taken Apart with Two Hands. *Discrete & Computational Geometry* 12 (1994), 367–384.

Peng Song, Chi-Wing Fu, and Daniel Cohen-Or. 2012. Recursive Interlocking Puzzles. *ACM Trans. on Graph. (SIGGRAPH Asia)* 31, 6 (2012), 128:1–128:10.

Peng Song, Chi-Wing Fu, Yueming Jin, Hongfei Xu, Ligang Liu, Pheng-Ann Heng, and Daniel Cohen-Or. 2017. Reconfigurable Interlocking Furniture. *ACM Trans. on Graph. (SIGGRAPH Asia)* 36, 6 (2017), 174:1–174:14.

Peng Song, Zhongqi Fu, Ligang Liu, and Chi-Wing Fu. 2015. Printing 3D Objects with Interlocking Parts. *Comp. Aided Geom. Des. (GMP)* 35-36 (2015), 137–148.

Rainer Storn and Kenneth Price. 1997. Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J. of Global Optimization* 11, 4 (Dec. 1997), 341–359. doi:10.1023/A:1008202821328

The CGAL Project. 2025. *CGAL User and Reference Manual* (6.1 ed.). CGAL Editorial Board. https://doc.cgal.org/6.1/Manual/packages.html

Yunsheng Tian, Jie Xu, Yichen Li, Jieliang Luo, Shinjiro Sueda, Hui Li, Karl D. D. Willis, and Wojciech Matusik. 2022. Assemble Them All: Physics-Based Planning for Generalizable Assembly by Disassembly. *ACM Trans. on Graph. (SIGGRAPH Asia)* 41, 6 (2022), 278:1–278:15.

Ziqi Wang, Peng Song, Florin Isvoranu, and Mark Pauly. 2019. Design and Structural Optimization of Topological Interlocking Assemblies. *ACM Trans. on Graph. (SIGGRAPH Asia)* 38, 6 (2019), 193:1–193:13.

Ziqi Wang, Peng Song, and Mark Pauly. 2018. DESIA: A General Framework for Designing Interlocking Assemblies. *ACM Trans. on Graph. (SIGGRAPH Asia)* 37, 6 (2018), 191:1–191:14.

Ziqi Wang, Peng Song, and Mark Pauly. 2021a. MOCCA: modeling and optimizing cone-joints for complex assemblies. *ACM Trans. on Graph. (SIGGRAPH)* 40, 4 (2021), 181:1–181:14.

Ziqi Wang, Peng Song, and Mark Pauly. 2021b. State of the Art on Computational Design of Assemblies with Rigid Parts. *Comp. Graph. Forum (Eurographics)* 40, 2 (2021), 633–657.

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

R. H. Wilson and T. Matsui. 1992. Partitioning An Assembly For Infinitesimal Motions In Translation And Rotation. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, Vol. 2. 1311–1318. doi:10.1109/IROS.1992.594555

Shi-Qing 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. on Graph. (SIGGRAPH)* 30, 4 (2011), 97:1–97:8.

Jiaxian Yao, Danny M. Kaufman, Yotam Gingold, and Maneesh Agrawala. 2017b. Interactive Design and Stability Analysis of Decorative Joinery for Furniture. *ACM Trans. on Graph.* 36, 2 (2017), 20:1–20:16.

Miaojun Yao, Zhili Chen, Weiwei Xu, and Huamin Wang. 2017a. Modeling, Evaluation and Optimization of Interlocking Shell Pieces. *Comp. Graph. Forum (Pacific Graphics)* 36, 7 (2017), 1–13.

Xinya Zhang, Robert Belfer, Paul G. Kry, and Etienne Vouga. 2020. C-Space Tunnel Discovery for Puzzle Path Planning. *ACM Trans. on Graph. (SIGGRAPH)* 39, 4 (2020), 104:1–104:14.

Yinan Zhang and Devin Balkcom. 2016. Interlocking Structure Assembly with Voxels. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. 2173–2180.