EPFL

Thèse n° 8964

Computational Analysis and Design of Structurally Stable Assemblies with Rigid Parts

Présentée le 21 décembre 2021

Faculté informatique et communications
Laboratoire d'informatique géométrique
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

Ziqi WANG

Acceptée sur proposition du jury

Prof. T. C. Käser Jacober, présidente du jury
Prof. M. Pauly, P. Song, directeurs de thèse
Prof. S. Coros, rapporteur
Prof. C.-W. Fu, rapporteur
Prof. W. Jakob, rapporteur

École
polytechnique
fédérale
de Lausanne

2021

会当凌绝顶，一览众山小。
— 杜甫

Someday may I climb up to its highest summit.
With one sweeping view see how small all other mountains are.
— Fu Du

To my parents

Acknowledgements

First, I would like to thank my Ph.D. advisor Mark Pauly for allowing me to do computer graphics research in this beautiful school. I am grateful that he gives me great freedom to explore my favorite topics. Thanks for helping me improve the quality of my research and teaching me how to present my works formally. I am grateful to have his encouragement and constructive suggestions when I am struggling. It is a pleasant journey that I have been through under his supervision.

Second, I want to thank my co-supervisor, Peng Song, for introducing me to this amazing topic of assembly. We first met when I was an undergraduate student. Since then, he has proposed many interesting ideas, which significantly inspires me, especially when I start my research. I want to thank him for helping me with my paper writing. I want to express my hope that our collaboration will go on in the future.

Third, I must thank other professors who provided valuable suggestions during our project meetings: Stelian Coros, Niloy Mitra, Wenzel Jakob, Fabio Gramazio, and Matthias Kohler.

I am thankful to my thesis committee for finding the time in their busy schedule to review my thesis and conduct the oral exam: Tanja Käser, Mark Pauly, Peng Song, Chi-Wing Fu, Stelian Coros, and Wenzel Jakob.

I want to thank my colleagues for sharing the incredible moments at EPFL: Madeleine Robert, Anastasia Tkach, Mina Konakovic, Julian Panetta, Liane Makatura, Samara Ren, Quentin Becker, Florin Isvoranu, Tim Chen, Ulysse Martel, Robin Jodon, Uday Kusupati, Davide Pellis, Filip Goč, Seiichi Suzuki. I especially want to thank Quentin Becker for helping me translate my thesis abstraction into French.

I want to thank NCCR digital fabrication (# 51NF40-141853) for offering my funding and thank my colleagues at ETH for discussing with me during my time at ETH: Tom Van Mele, Ania Apolinarska, Gene Ting-Chun Kao, Simon Duenser, Victor Leung, Gonzalo Casas, Matteo Pacher, and Davide Tanadini.

I also want to thank other researchers for providing helpful suggestions for my research: Yijiang Huang and Hao Xu.

I want to thank my friends for their support: Zhao Ma, Yijiang Liu, Yu Zhang, Tin Yin, Meng Zhao, Qinkun Bao, and others.

Finally, my biggest thanks go to my parents for their love and support.

Acknowledgements

Lausanne, 8 Oct 2021 Ziqi Wang

Abstract

An assembly refers to a collection of parts joined together to achieve a specific form and/or functionality. Assemblies make it possible to fabricate large and complex objects with several small and simple parts. Such parts can be assembled and disassembled repeatedly, benefiting the transportation and maintenance of the assembly. Due to these advantages, assemblies are ubiquitous in our daily lives, including most furniture, household appliances, and architecture. The recent advancement in digital fabrication lowers the hurdles for fabricating objects with complex shapes. However, designing physically plausible assemblies is still a non-trivial task as a slight local modification on a part's geometry could have a global impact on the structural and/or functional performance of the whole assembly. New computational tools are developed to enable general users involved in the design process exploiting their imagination.

This thesis focuses on static assemblies with rigid parts. We develop computational methods for analyzing and designing assemblies that are structurally stable and assemblable. To address this problem, we use integral joints i.e., tenon and mortise, that are historically used because of their reversibility which simplifies the disassembly process significantly. Properly arranged integral joints can restrict parts' relative movement for improved structural stability. However, manually finding the right joints' geometry is a tedious and error-prone task. Inspired by the kinematic-static duality, we first propose a new kinematic-based method for analyzing the structural stability of assemblies. We then develop a two-stage computational design framework based on this new analyzing method. The kinematic design stage determines the amount of motion restrictions imposed by joints to make a given assembly stable in the motion space. The geometric design stage searches for proper shapes of the joints to satisfy the motion restriction requirements computed from the previous stage. To solve the problem numerically, we propose the joint motion cones to measure the motion restriction capacity of given joints. Compared with previous works, our framework can efficiently handle inputs with complex geometry. Besides, our design framework is very flexible and can easily be adapted to various applications:

1. First, we focus on designing globally interlocking assemblies that can withstand arbitrary external forces and torques. Our method abstracts the contacts between parts into a set of base directional blocking graphs and leverages efficient graph analysis tools to test for globally interlocking. These graphs can also be used to guide our kinematic design. Our method supports a wider range of assembly forms compared to previous methods and provides significantly more design flexibility.

2. Second, we are interested in designing assemblies of rigid convex blocks to approximate freeform surfaces. The convexity of our parts simplifies the fabrication though it brings a considerable challenge for design due to the blocks' loose planar joints. Our design framework can optimize the blocks' shape to generate assemblies with good resistance against lateral forces, and in some cases, globally interlocking assemblies.

3. Lastly, we present a method for designing complex assemblies with cone joints. Cone joints generalize the classic single-direction joints with a cone of disassembling directions. By optimizing the shapes of cone joints, our design framework can find the best trade-off between structural stability and assemblability.

We validate our computational tools by fabricating a series of physical prototypes. Our algorithms have great potential to be applied for solving various assembly design problems ranging from small-scale such as toys and furniture to large-scale such as art installation and architecture. For example, the proposed techniques could be applied for designing discrete architecture that can be automatically constructed with robots.

Keywords: 3D assembly, interlocking assembly, topological interlocking assembly, cone joint, structural stability, assemblability, computational design

Résumé

Un assemblage est un ensemble de pièces combinées pour obtenir une forme et/ou une fonctionnalité spécifique. Les assemblages permettent de fabriquer des objets volumineux et complexes à partir de plusieurs petites pièces simples. De telles pièces peuvent être assemblées et démontées de manière répétée, ce qui facilite le transport et favorise la maintenance de l'assemblage. En raison de ces avantages, les assemblages sont omniprésents dans notre vie quotidienne, que ce soit pour la conception de meubles d'intérieur, d'appareils électroménagers ou autres œuvres architecturales. Les récents progrès dans le domaine de la fabrication numérique abaissent les barrières quant à la fabrication d'objets aux formes complexes. Cependant, la conception d'assemblages physiquement réalisables reste une tâche non triviale car une légère modification locale de la géométrie d'une pièce pourrait avoir un impact global sur les performances structurelles et/ou fonctionnelles de l'assemblage. De nouveaux outils informatiques sont développés pour permettre aux utilisateurs non-experts impliqués dans le processus de conception de laisser libre court à leur imagination.

Cette thèse porte sur les assemblages statiques avec des pièces rigides. Nous développons des méthodes informatiques pour analyser et concevoir des assemblages structurellement stables et assemblables. Pour résoudre ce problème, nous considérons des jonctions intégrées i.e., mortaise, historiquement choisies pour leur réversibilité qui simplifient considérablement le processus de démontage. Correctement disposées, ces jonctions intégrées peuvent restreindre les mouvements relatifs des pièces pour améliorer la stabilité structurelle de l'ensemble. Cependant, trouver manuellement la bonne géométrie des jonctions est une tâche fastidieuse et sujette aux erreurs. Inspirés par la dualité cinématique-statique, nous proposons dans un premier temps une nouvelle méthode cinématique pour analyser la stabilité structurelle des assemblages. Nous développons ensuite une méthode de conception informatique en deux étapes basée sur cette nouvelle technique d'analyse. L'étape de conception cinématique détermine les restrictions de mouvement imposées par les jonctions nécessaires à la stabilité de l'assemblage dans l'espace des mouvements. L'étape de conception géométrique consiste à chercher les formes de jonctions satisfaisant les exigences de restriction de mouvement calculées lors de l'étape précédente. Pour résoudre le problème numériquement, nous proposons le concept de cônes de mouvement aux jonctions pour mesurer la capacité de restriction de mouvement des jonctions. Par rapport aux travaux précédents, notre système peut gérer efficacement des géométries complexes. De plus, notre méthode de conception est très flexible et peut facilement être adapté à diverses applications :

- Premièrement, nous nous concentrons sur la conception d'assemblages globalement imbriqués pouvant résister à des forces et des couples externes arbitraires. Notre méthode représente les contacts entre les pièces sous la forme d'un ensemble de graphes orientés de blocage et exploite des outils d'analyse de graphes efficaces pour tester l'imbrication globale. Ces graphes peuvent également être utilisés pour guider notre conception cinématique. Notre méthode prend en charge une plus large gamme de formes d'assemblage par rapport aux méthodes précédentes et offre une flexibilité de conception nettement supérieure.
- Deuxièmement, nous nous intéressons à la conception d'assemblages de blocs convexes rigides pour approximer des surfaces de forme libre. La convexité de nos pièces simplifie la fabrication bien qu'elle pose un défi considérable pour la conception en raison de l'instabilité des plans de jonction inter-blocs. Notre méthode de conception permet d'optimiser la forme des blocs pour générer des assemblages présentant une bonne résistance aux efforts latéraux et pouvant être globalement imbriqués.
- Enfin, nous présentons une méthode de conception d'assemblages complexes à joints coniques. Les joints coniques généralisent les joints monodirectionnels classiques en élargissant le champ des directions de démontage à un cône. En optimisant les formes des joints coniques, notre méthode de conception peut trouver le meilleur compromis entre stabilité structurelle et facilité d'assemblage.

Nous validons nos outils de calcul en fabricant un ensemble de prototypes physiques. Nos algorithmes ont un grand potentiel en ce qui concerne la résolution de divers problèmes de conception d'assemblage de petite à grande échelle. Jouets, meubles, installations artistiques et architecturales sont autant d'exemples d'applications de notre méthode. En particulier, les techniques proposées peuvent être appliquées pour concevoir une structure constructible automatiquement par des robots.

Mots-clés : assemblage 3D, assemblage imbriqué, assemblage topologiquement imbriqué, jonction conique, stabilité structurelle, facilité d'assemblage, conception informatique

Contents

List of Figures

1 Introduction

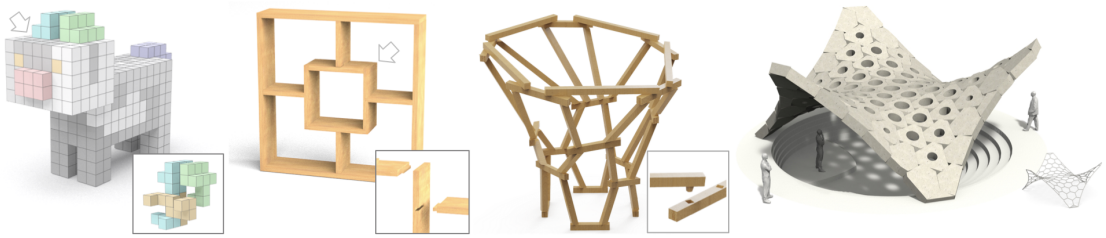


Figure 1.1 – Structurally stable assemblies with rigid parts connected by integral joints, from left to right: interlocking voxelized puzzle, interlocking bookshelf, spatial timber frame structure, and topological interlocking assembly.

An assembly refers to a collection of parts joined together to achieve a specific form and/or functionality. Compared with a monolithic object, parts of an assembly have relatively simple geometry, which modern digital machines can easily manufacture. During transportation, parts can be packed tightly into boxes to save storage space. The joints that connect parts provide structural integrity to the assembly and facilitate an easy (dis)assembly process. Malfunctioning parts can be replaced without damaging the unimpaired ones. Reconfigurable assemblies can accomplish multiple functionalities by transforming themselves into different forms. Dynamic assemblies can perform various mechanisms by transferring motions through parts. Assemblies are so ubiquitous in our daily lives that most of our consumer products, machines, and buildings are assemblies.

In general, "assembly" is a very broad concept covered by many research subjects. According to their functionality, assemblies can be classified as structures that transmit force to carry loads and mechanisms that transfer motion and force to perform mechanical work. This thesis limits its scope to static structures with rigid parts and focuses on the computational design of structurally stable assemblies. An assembly is structurally stable if it can preserve its form under external forces without collapse [Whiting et al., 2009]. Figure 1.1 illustrates some typical examples. In most of these applications, our rigidity assumption is a reasonable simplification

since parts (i.e., made by stone) often have high material strength. The structural failures are in consequence of parts' movements (sliding and rotating at the contact interfaces) rather than material failure [Shin et al., 2016]. The structural stability of our assemblies can be independent of the material selection.

With the development in digital fabrication, there are more and more demands from the general public to make their personalized assemblies. Most state-of-art CAD software is designed to model parts in a virtual environment. The physical feasibility of the assemblies is only examined after the modeling process. Not all designs are physically plausible, and users have to manually correct them, which requires effort and expertise. Therefore, we are interested in developing a new computational framework that optimizes the parts' geometry to satisfy the physical feasibility constraints automatically. In addition to the structural stability, the assemblability that requires assemblies to be dismantled into parts without deadlocking [Song et al., 2012] is another fundamental physical requirement that our designs need to satisfy. Other aspects, including fabricability, aesthetics, and scaffolding, are considered upon request.

Within the whole modeling process, selecting proper joints is a critical step for obtaining physically plausible assemblies. Using permanent or irreversible connectors (i.e., glue and nails), even though they can provide a trial solution to achieve stability, their parts can hardly be disassembled without any damage on them. Integral joints, on the other hand, connect parts into a steady structure purely based on geometric blocking. This intriguing property facilitates repeated assembly and disassembly and significantly simplifies the correct alignment of parts during construction. However, designing structurally stable assemblies with integral joints is an extremely challenging task even for experts. The relation between the joints' geometry and structural stability is not intuitive since a slight modification on an individual part geometry could have a global impact on the stability of the whole assembly. Simply adding excessive joints to an assembly for stability often ends up with deadlocking assemblies. Removing joints from assemblies until the deadlocking is resolved has no guarantee of maintaining their structural stability. Structural stability and assemblability are two conflicting design goals that are hard to be satisfied simultaneously without computational approaches.

To overcome the challenges mentioned above, a previous work [Whiting et al., 2009] proposed the rigid body equilibrium method (RBE), which analyzes the structural stability of a given assembly using force equilibrium. Their approach aims to find a network of interactive forces between parts that can balance the external forces (i.e., gravity). Rather than a binary output, the RBE method further measures the infeasibility energy by computing the minimal amount of unrealistic forces (i.e., tension forces for stone) needed to stabilize the assembly. They set up a gradient-based algorithm to optimize the parts' geometry for stability based on this measurement [Whiting et al., 2012]. When assemblies have complicated shapes (i.e., curved contacts), the interactive forces in the RBE test have to be densely sampled to obtain sufficient accuracy. The size of the optimization problem could become enormous, making the optimization program inefficient and heavily dependent on the initial values.

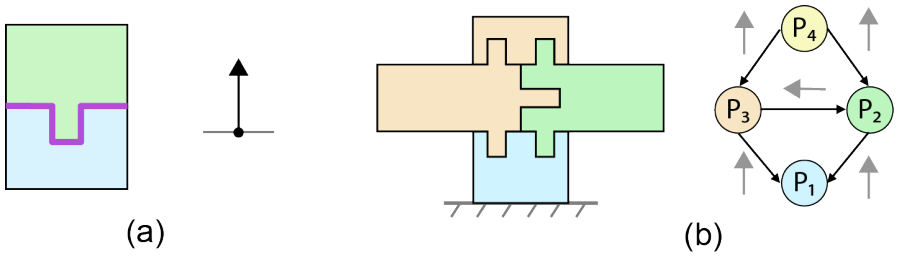
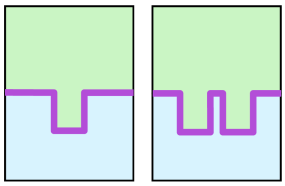


Figure 1.2 – (a) Two parts connected by a single-direction joint whose joint motion cone is just a vector pointing upward; and (b) a four-part assembly and its motion-based representation where the arrows at the graph edges are the joint motion cones.

We find that the design space of the joints' geometry has many redundancies. Both green parts in the right inset can only move upward though the joints of the two assemblies are geometrically different. We can reduce the size of the design space by clustering joints that have the same capacity to restrict the relative movements of associated parts. We invent the joint motion cone to measure the joint's motion restriction capacity. The joint motion cone contains all infinitesimal rigid motions that can separate the two associated parts without collision. For a single-direction joint, its motion cone has only one vector, the joint's disassembling direction; see Figure 1.2(a). Once two joints in two assemblies share the same joint motion cone, they can be interchanged without affecting the structural stability of the two assemblies. Due to the duality between kinematics and statics, we can derive an equivalent motion-based stability analysis from the previous forced-based equilibrium method. The method only takes the joint motion cones as inputs without needing the actual joints' geometry. We further can represent the assembly geometry by a lightweight data structure that only contains joint motion cones illustrated in Figure 1.2(b). Our new motion-based representation is developed from a part graph whose nodes are parts and edges are joints. We augment this part graph with joint motion cones at its edges. This new representation discards irrelevant geometric features of the original assembly without affecting the structural stability measurement. We can perform a gradient-based optimization on this new representation. Our optimization has two stages, the kinematic and geometric design stages. At the kinematic design stage, our method optimizes target joint motion cones to make the motion-based representation conceptually stable. The geometric design stage then optimizes the shapes of the joints to satisfy the target motion cones. The separation of motion and geometry design allows us to include many types of structural stability and joint in a unified framework.



The followings discuss the specific types of structural stability and joint used in this thesis. Structurally stable assemblies have divergent definitions in different external loading conditions. This thesis mainly discusses three typical types: assemblies being equilibrium under gravity, assemblies with lateral stability, and globally interlocking assemblies, sorted in the ascending order by the amount of external forces the structures can resist; see Figure 1.3.

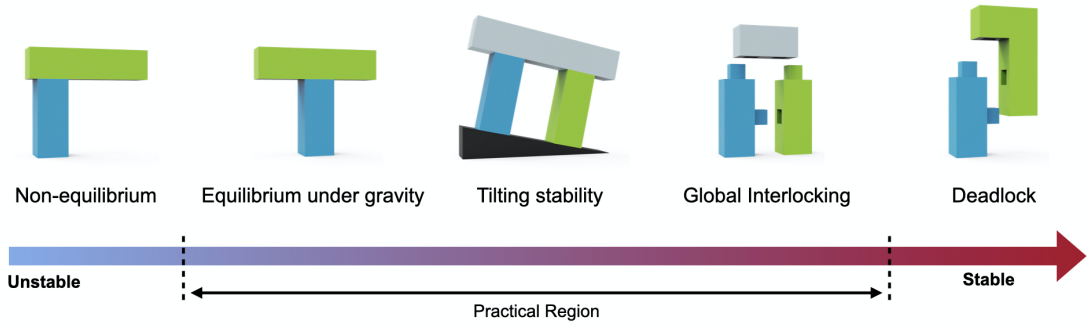


Figure 1.3 – The structures from left to right can withstand more external forces. The assemblies at the two ends are either unstable or not assemblable, making them infeasible in practice.

Being equilibrium under gravity is the minimum requirement when designing architecture assemblies. A self-supporting structure, as an example, transmits the self-weights of its parts through the integral joints to the ground to stay equilibrium [Panozzo et al., 2013]. However, during the assembling process, the partial assemblies may not be stable. Temporary scaffolds keep the structure in equilibrium before the last part is installed. An interesting extension is to design the partial assemblies to be stable at as many assembling steps as possible [Gao et al., 2019]. When an assembly is exposed to different forces (e.g., live loads), the equilibrium under gravity conditions might be insufficient. This motivates stability measures based on tilting analysis [Shin et al., 2016, Yao et al., 2017a]. The ground plane of a structure is rotated around a fixed axis to mimic the effect of unexpected lateral acceleration (i.e., wind or earthquake). The maximum angle the ground plane can be tilted is denoted as the critical tilt angle, which measures the structure's ability to resist against non-gravitational external forces. In the extreme, some structures can withstand arbitrary external forces. These assemblies are called globally interlocking assemblies [Song et al., 2012, Fu et al., 2015]. The main challenge of designing globally interlocking assemblies is to ensure two conflicting properties simultaneously: globally interlocking and disassemblability. This is because globally interlocking requires strict joining to restrict relative movement among parts, yet disassemblability demands at least one collision-free plan to separate the parts (i.e., not deadlocking).

The integral joint is implicitly defined as the portion of each individual part that is in contact with adjacent parts. Figure 1.4 illustrates three frequently used integral joints (the planar, curved, and single-direction joints) and their most represented applications. Planar contacts (Figure 1.4(a)) are commonly seen in unreinforced masonry structures. Their simplest geometry facilitates easy fabrication and reduces stress concentration, which is important in avoiding material failure for rigid material (i.e., stone). Yet, these joints have the weakest capacity to restrict relative part motion. To achieve better structural stability, this thesis studies the topological interlocking assemblies whose parts have planar faces oriented in an alternating manner; see Figure 1.4(a). This modeling approach allows the assemblies to have large tilting angles or even become interlocking. For assemblies with sparse contacts, like furniture, the single-direction joints are the most reliable choice to provide structural stability;

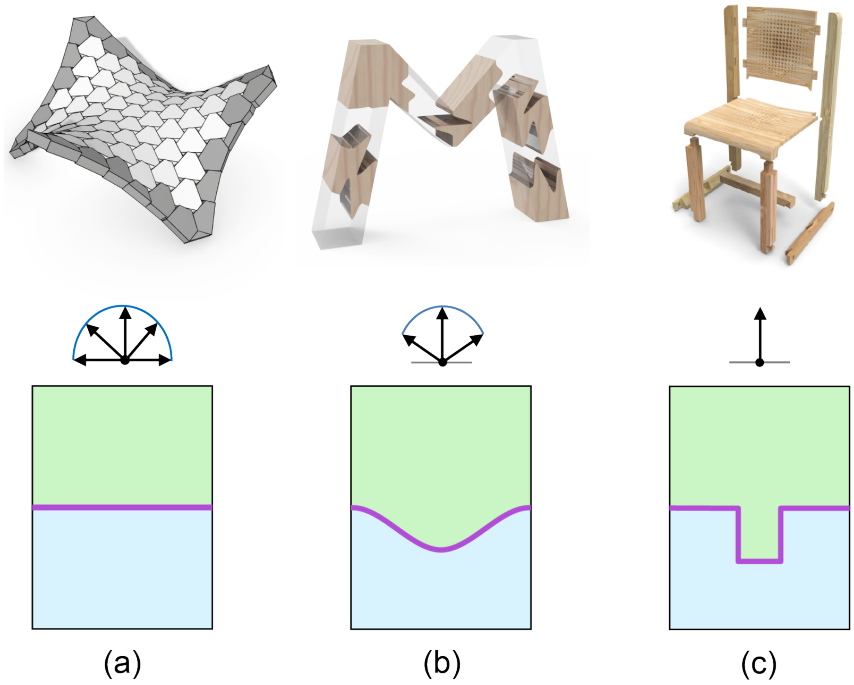


Figure 1.4 – Typical joints studied in this thesis: (a) a planar contact, (b) a cone joint with a curved contact, and (c) a single-direction joint (contacts are shown in purple). The top row shows the most represented assemblies where the joints are used.

see Figure 1.4(c). Their strong capacity to strengthen structural stability makes them widely used in globally interlocking assemblies. However, complex arrangements of single-direction joints could lead to deadlocking, making the assembly physically unrealizable. Moreover, these joints may complicate the assembly process as inserting a part precisely along a certain direction to fit the other could be a challenging task, especially in robotic assembly [Leung et al., 2021]. Curved joints shown in Figure 1.4(b) generalize single-direction joints and planar contacts in terms of restricting relative part motion. Structures made by curved joints can have balanced assemblability and stability. Curved joints are also called cone joints because they allow one part to be separated/inserted relative to the other by translation along any direction within a motion cone. Cone joints have been demonstrated to have good mechanical properties such as reduced stress concentration in building structurally stable assemblies [Dyskin et al., 2003b, Javan et al., 2016]. Parts with cone joints can be easily fabricated with 3D printing, CNC milling, and even hot-wire cutting for large-scale objects [Duenser et al., 2020].

1.1 Contributions

Contributions for analyzing structural stability:

- We establish a connection between the geometry of a joint and its motion space based

on convexity theory. We show that the joint motion space is always convex and present a sampling-based approach to compute the motion space of curved-contact joints. We further derive a motion-based method for static analysis of assemblies with cone joints from the existing force-based method due to the duality between static and kinematic. The strength of this new method is to quantify structural stability and assemblability coherently in the motion space.

- We demonstrate two methods to test for global interlocking. For assemblies made by voxels, we represent them with a set of base Directional Blocking Graphs (DBGs) and implement an efficient graph analysis algorithm that can test for global interlocking in polynomial time complexity. For assemblies with general contacts, our second test algorithm is based on solving a system of linear inequalities considering not only part translation but also rotation, avoiding false positives that can occur with existing methods.

- We formalize a theoretical link between static equilibrium conditions and a globally interlocking property with a mathematical proof. We show that global interlockings are the most stable assemblies that can withhold arbitrary external forces and torques. We utilize the tilting test to quantitatively measure the stability of the assemblies, which can withstand more than gravity.

Contributions for designing structurally stable assemblies:

- We introduce a general iterative framework for designing interlocking assemblies that can explore the full search space of all possible interlocking configurations by utilizing all existing part blocking relations described in the graphs. We demonstrate the flexibility of our framework for designing different classes of assemblies, including new types of interlocking forms that have not been explored in previous works.

- We develop an interactive design tool that allows a real-time preview and efficient exploration of a wide range of design parameters of topological interlocking assemblies. We present a gradient-based method that optimizes the geometry of blocks to maximize the tilting stability measure for topological interlocking assemblies.

- We develop an optimization approach to construct cone joints for designing structures that are assemblable and stable, assuming the assembly sequence is given. Our framework iterates between a kinematic design stage that determines the required motion cone for each part contact and a geometric realization stage that finds the geometry of each joint to match this motion cone.

1.2 Publication

This thesis mainly covers the following peer-reviewed publications:

- Ziqi Wang, Peng Song, and Mark Pauly. DESIA: A General Framework for Designing Interlocking Assemblies. ACM Trans. on Graph. (SIGGRAPH Asia 2018).
- Ziqi Wang, Peng Song, Florin Isvoranu, and Mark Pauly. Design and Structural Optimization of Topological Interlocking Assemblies. ACM Trans. on Graph. (SIGGRAPH Asia 2019).
- Ziqi Wang, Peng Song, and Mark Pauly. MOCCA: modeling and optimizing cone-joints for complex assemblies. ACM Trans. on Graph. (SIGGRAPH 2021).
- Ziqi Wang, Peng Song, and Mark Pauly. State of the Art on Computational Design of Assemblies with Rigid Parts. Computer Graphics Forum (Eurographics STAR 2021).

In addition, the following publications were published during the same time period but are not explicitly addressed in this thesis:

- Yang Xu, Ziq Wang, Siyu Gong, Yong Chen. Reusable support for additive manufacturing. Additive Manufacturing 2021.

1.3 Overview

We present computational methods of analyzing and designing structurally stable assemblies with rigid parts. The remainder of the thesis is organized as follows:

- In Chapter 2, we review the state-of-art methods for joint analysis, assembly planning, and structural stability measurement. We further discuss previous design methods for self-supporting assemblies, interlocking assemblies, and topological interlocking assemblies.
- In Chapter 3, we summarize the computational methods for analyzing different types of structural stability. We first review the previous force-based equilibrium test, followed by our motion-based equilibrium test. Then, we introduce the concept of joint motion cones and use it to construct the new motion-based representation. Next, we propose two ways of testing global interlocking assemblies, the directional blocking graph, and the inequality-based test. Both methods can be derived from our motion-based representation. Lastly, we formalize a measurement for lateral stability, which together with other stability tests can measure where an assembly with rigid parts is located in the stability spectrum.
- Chapter 4 focuses on designing global interlocking assemblies. An assembly is defined as global interlocking if only if any part and subset of parts is immobilized except a key part. The classic interlocking test method, which examines the immobility of every subset of parts has exponential time complexity. Introduced in Chapter 3, the

directional blocking graphs (DBGs) can test for the global interlocking in a polynomial time complexity, which motivates us to utilize these DBGs for designing interlocking assemblies. In Section 4.2, our computational framework starts with the full input model, then iteratively extracts successive parts for disassembly. We carefully design the split process such that the interlocking property of each DBG is maintained at every iteration. In Section 4.3 we show different types of assemblies generated with our approach, compare with previous works, and highlight several application examples.

- Chapter 5 discusses the algorithm to design topological interlocking assemblies. A topological interlocking (TI) assembly is an ensemble of convex blocks, arranged in a regular topology, to approximate a freeform surface. The internal blocks of TI assemblies are immobilized by a fixed periphery to obtain tilting stability. Section 5.2 introduces a parametric model for TI assemblies that facilitates a constructive approach for design exploration. Section 5.3 presents a gradient-based optimization to improve the structural stability of an assembly with respect to the measure. In Section 5.4 we show and discuss a variety of TI assemblies designed by our approach.

- Chapter 6 makes use of the cone joints described in Chapter 3 to design assemblies for both structural stability and assemblability. Unlike the single-direction joints (Chapter 4) mainly for stability and the planar joints (Chapter 5) mainly for assemblability, the cone joints can interpolate between the two extremes to have a balanced trade-off between our two design objectives. Section 6.2 discusses the method of modeling the 2D/3D cone joints in a parametric way. Each part geometry is constructed by combining the cone joints with non-contact features. The motion-based representation explained in Chapter 3 naturally becomes the most suitable abstraction for measuring the structural stability of assemblies with cone joints. Section 6.3 conducts the two-stage optimization approach on this motion-based representation to construct the geometry of cone joints such that the resulting assemblies can satisfy our two design goals. In Section 6.4, we show various 2D and 3D assemblies with cone joints designed by our technique, compare with previous works and highlight several application examples.

- Finally, Chapter 7 summarizes the main contributions of this thesis and includes a discussion of future work.

2 Related Works

2.1 Computational Analysis of Assemblies

Computational analysis of assemblies evaluates different aspects of assemblies with given geometry, including joining parts of an assembly (Section 2.1.1), planning of the assembly process (Section 2.1.2), and structural stability of the whole assembly (Section 2.1.3)

2.1.1 Joining Parts

To form an assembly that can be used in practice, component parts need to be joined together to restrict relative movements among the parts. This (potentially additional) geometry or material used to connect parts defines the joining method, or simply the joint.

Joint classification. Joints can be classified as *permanent joints* and *non-permanent joints*. Typical permanent joints include adhesive material (e.g., glue, mortar) and permanent mechanical fasteners (e.g., rivets). Although assemblies connected with permanent joints can be structurally very stable, a significant drawback is that the assembly cannot be disassembled without potential damage to the parts. In contrast, non-permanent joints encourage parts disassembly and reassembly, facilitating storage, transportation, maintenance, and reconfiguration of assemblies.

Non-permanent joints can be classified as *external joints* and *integral joints*, depending on whether the joint geometry is integrated on each individual part. Screws and pins are conventionally used external joints. To satisfy specific needs on parts joining in digital fabrication, customized external joints have been designed and used in practice [Magrisso et al., 2018]. These external joints are independent from the parts. Hence, they can be abstracted as "conceptual parts" in the analysis of assemblies, e.g., assembly planning.

Integral joints are implicitly defined as the portion of each individual part that is in contact with adjacent parts. The simplest integral joints would be planar contacts between neighboring parts [Whiting et al., 2009, Wang et al., 2019]. More complex integral joints include

curved contacts between parts [Krishnamurthy et al., 2021] and conventional woodworking joints [Fairham, 2013]; see Figure 2.1 for examples. Integral joints can significantly simplify the assembly process by assisting parts alignment and reducing the total number of assembly steps (i.e., no external fasteners need to be applied). Integral joints can also add to an assembly's structural durability (e.g., woodworking joints in architecture) and visual appeal (e.g., decorative joints in furniture [Yao et al., 2017a]). Due to this reason, integral joints are more and more widely used in digital fabrication of assemblies [Zheng et al., 2017, Larsson et al., 2020].

Joint analysis identifies contacts between each pair of parts in an assembly by computing the minimum distance between them and checking if this minimum distance is less than a given threshold (very small positive number). Figure 2.1 highlights part contacts as purple lines or curves. Based on the identified part contacts, joint analysis can obtain the following information of an assembly:

- *Parts connectivity.* Two parts are connected if they have at least one contact. All contacts between the two parts define joints that connect the two parts. A *parts-graph* [Fu et al., 2015] is typically used to represent parts connectivity in an assembly, where each node represents a part and each edge represents joints connecting the two associated parts. The dual of a parts-graph is a *joints-graph*. See Figure 2.2.

- *Parts mobility.* The contacts between two parts enforce constraints on their relative movement as collisions have to be avoided when moving one part relative to the other. These contact constraints typically can be formulated as a linear system [Wilson and Matsui, 1992], whose solution space corresponds to the infinitesimal motion space of one part relative to the other. See Figure 2.1 for examples, in which only translational motion is considered.

- *Joints strength.* Conceptually, arbitrarily small contacts or thin joints can constrain the relative movement between two rigid parts. However, in practice such joints should be avoided to reduce the risk of structural failure. To detect such issues, finite element methods can be used to analyze joint strength under external loads [Yao et al., 2017b].

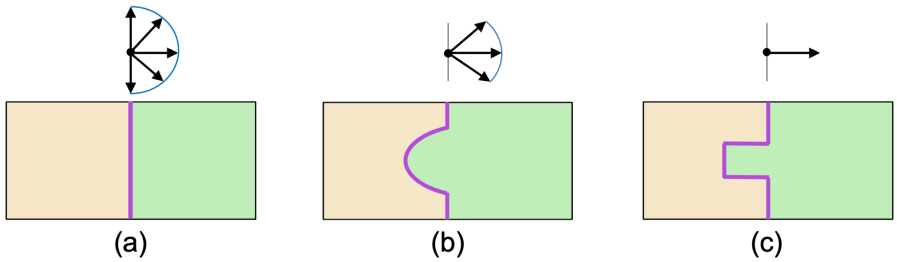


Figure 2.1 – Schematic of three kinds of integral joints with corresponding translational motion space of the green part illustrated on top. (a) Planar contact joint; (b) curved contact joint; and (c) mortise and tenon joint.

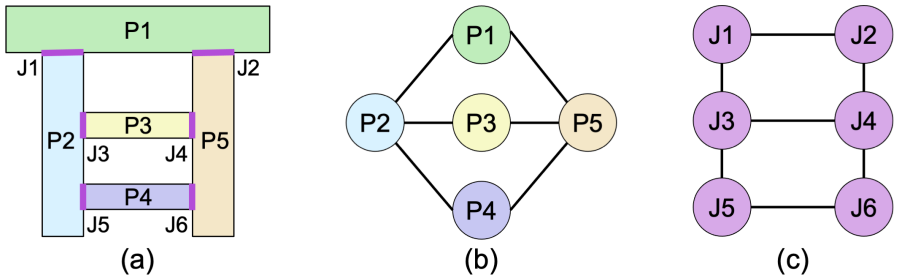


Figure 2.2 – (a) A 5-part assembly, where part contacts (i.e., joints) are highlighted in purple; (b) parts-graph; and (c) joints-graph.

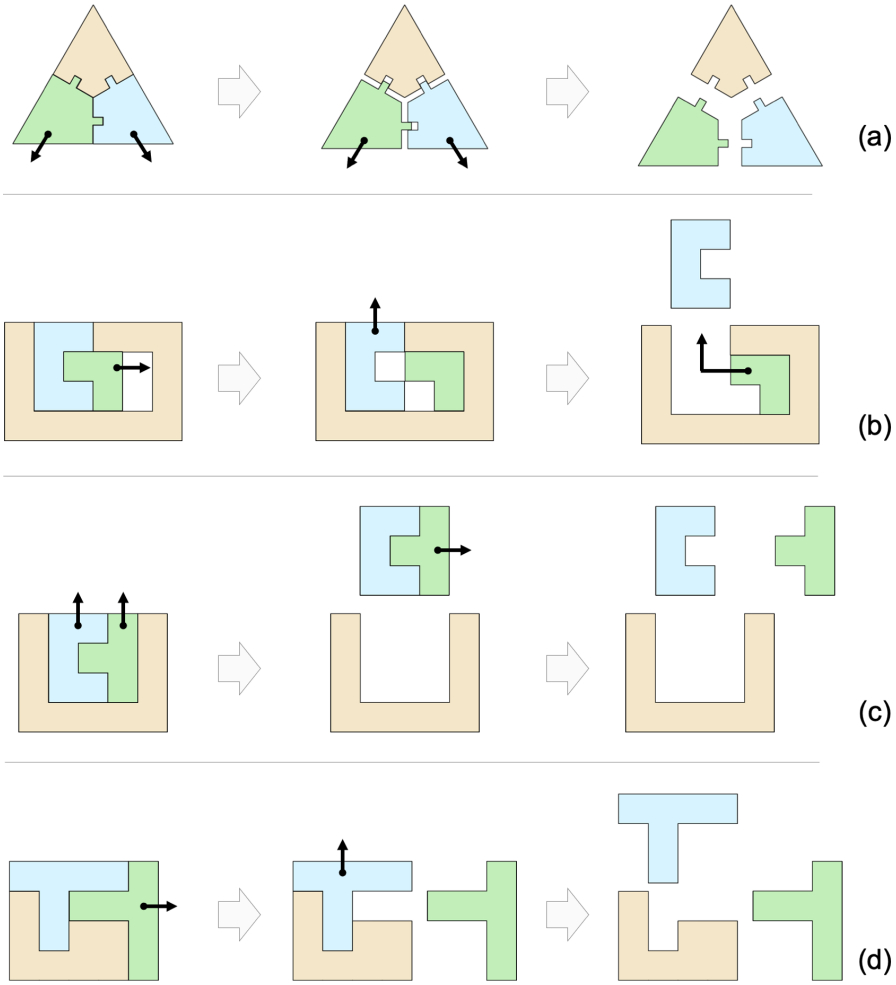


Figure 2.3 – Examples of disassembly plans, where the orange part is fixed as a reference. (a) A three-handed disassembly plan: the green and cyan parts translate along different directions simultaneously. (b) A non-monotone disassembly plan: intermediate placement of the green part is necessary.(c) A non-linear disassembly plan: the first disassembly operation is to translate the green and cyan parts together along the same direction. (d) A sequential, monotone, and linear disassembly plan.

2.1.2 Assembly Planning

Widely used in automated manufacturing, robotics, and architecture, assembly planning is the process of creating detailed instructions to combine separate parts into the final structure. The goal of assembly planning is to find a sequence of operations to assemble the parts (*assembly sequencing* [Jiménez, 2013]), determine the motions that bring each part to its target pose (*assembly path planning* [Ghandi and Masehian, 2015]), and propose the utilization of additional resources such as supports and tools to assist the assembly process.

A closely related problem is *disassembly planning*, which creates a plan for disassembling component parts from an installed assembly. An important strategy of assembly planning is *assembly-by-disassembly*, where an assembly plan is obtained by disassembling an installed product into its component parts and then reversing the order and path of disassembly. This strategy is feasible as there is a bijection between assembly and disassembly sequences and paths when only geometric constraints are concerned and all parts are rigid [Halperin et al., 2000]. The advantage of this strategy is that it can drastically reduce the size of the solution space (i.e., valid assembly plans), since parts in an assembled state have far more precedence and motion constraints than in a disassembled state. However, when physical constraints are taken into consideration, e.g., supports of incomplete assemblies [Deuss et al., 2014], this strategy is not directly applicable to compute assembly plans.

The complexity of assembly planning is measured generally in terms of the number of parts and their shape complexity. However, this measure alone does not express how difficult it is to obtain a valid assembly plan. Other involved features are:

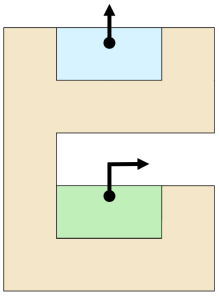
- *The number of hands:* the maximum number of moving subassemblies with respect to one another in any assembly operation.
- *Monotonicity:* whether or not operations of intermediate placement of subassemblies are required.
- *Linearity:* whether all assembly operations involve the insertion of a single part in the rest of the assembly.

Figure 2.3 shows disassembly plans to illustrate the above features. The simplest (dis)assembly plans are sequential (two-handed), monotone, and linear. Due to the simplicity, they are the most widely used (dis)assembly plans in computational design of assemblies.

Assembly planning problems can be broadly classified into two classes. The first aims to find a valid assembly plan to ensure assemblability of designed assemblies such as 3D puzzles. The second is to find a desired assembly plan to satisfy some objectives on the assembly process such as reducing usage of additional resources (e.g., formwork for construction of architecture). This thesis reviews existing works in the graphics community to address these problems. Readers are referred to [Ghandi and Masehian, 2015] for a more comprehensive survey on (dis)assembly planning problems and approaches.

Search for a valid assembly plan. Given a 3D assembly, there could exist a number of valid plans to assemble the parts. Here, we consider only geometric constraints, i.e., an assembly plan is defined as valid if there is no collision when assembling each part. As mentioned above, assemblies can be naturally represented as graphs. Graph data structures can also guide us in finding valid disassembly plans by maintaining a dynamic graph corresponding to the remaining assembly as parts are successively removed.

Parts-graph based approach. In this approach, a valid assembly plan is computed by using the assembly-by-disassembly strategy. The idea is to identify removable parts guided by the parts-graph (see Figure 2.2(b)) since a part with fewer neighbors in the parts-graph has higher chance to be removable. First, we compute mobility for each part in the parts-graph, e.g., using the joint analysis approach in Section 2.1. Next, we choose one movable part (usually with few neighbors in the parts-graph), remove it from the assembly using the computed motion, and update the parts-graph accordingly. Since the first step only ensures collision-free infinitesimal rigid motion for the movable part, we still need to check collision with the remaining parts when taking out the movable part in the second step; e.g., the cyan part in the inset can be removed along the translational direction yet the green part has to translate one more step to avoid collision with the orange part. We iterate the above two steps until there is only one part remaining in the parts-graph.



The above approach assumes sequential, monotone, and linear disassembly plans. Thus, finding a valid disassembly plan is a sufficient but not necessary condition of assemblability. To support non-linear disassembly plans, the approach should check mobility not just for each individual part but also for each subassembly. However, this extension will increase complexity of the approach from linear to exponential in the number of parts.

Blocking graph based approach. Finding an assembly plan requires identifying movable parts and part groups at each intermediate assembly state, often leading to a combinatorial search problem. To solve this task more efficiently, Wilson [Wilson, 1992] invented a *Directional Blocking Graph* and a *Non-Directional Blocking Graph* to represent blocking relations among parts in an assembly. The detailed description of *Directional Blocking Graph* and *Non-Directional Blocking Graph* is discussed in Section 3.5.2.

Search for a desired assembly plan. Assembly planning can be formulated as an optimization to find a desired assembly plan. Typical optimization objectives include minimizing assembly complexity (e.g., short assembly path, simple assembly motion), minimizing usage of additional resources (e.g., supports to maintain stability of incomplete architectural structures [Deuss et al., 2014]), and maximizing parts visibility for creating visual assembly instructions [Agrawala et al., 2003, Heiser et al., 2004]. Please refer to [Jones and Wilson, 1996] for an exhaustive list of objectives on searching assembly plans.

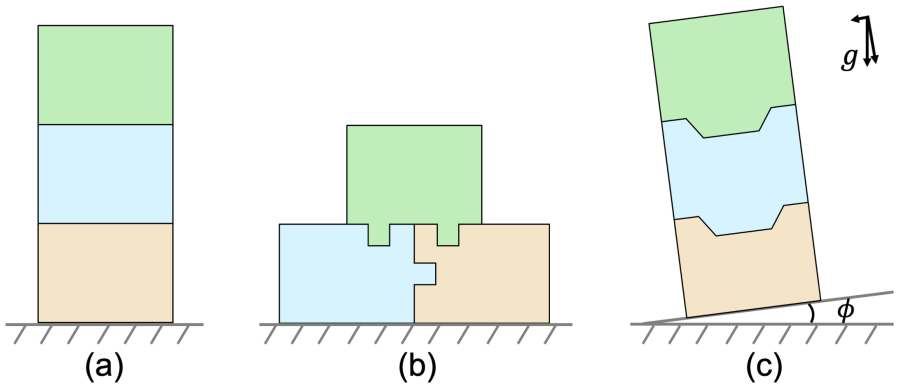


Figure 2.4 – Structurally stable assemblies: (a) an assembly in equilibrium; (b) an interlocking assembly, where the green part is the key; and (c) an assembly under tilt analysis, in which the assembly's stability is measured using the critical tilt angle ϕ.

To find an optimal assembly plan, we need to enumerate and evaluate all possible assembly plans based on a selected objective. Although this is possible for assemblies with a small amount of parts, e.g., by using AND/OR tree data structure [de Mello and Sanderson, 1990], the complexity increases exponentially with the number of parts. Due to this reason, various practical algorithms were developed to find sub-optimal solutions, e.g., using a greedy algorithm [Deuss et al., 2014, Mellado et al., 2014], a heuristic search [Agrawala et al., 2003], or an adaptive sampling followed by user editing [Kerbl et al., 2015].

The above existing works mainly focus on sequential and monotone assembly plans. Although these plans are relatively easy to execute, it is an open problem to study more complex assembly plans. One good example is a recent work [Zhang et al., 2020] that finds non-coherent assembly plans to solve two-part disentanglement puzzles, where a part that is inserted may not touch the other previously placed part. Other complex assembly plans include non-sequential plans (see Figure 2.3(a)) to stabilize parts in an assembly by making them harder to be taken out, and non-monotone plans [Masehian and Ghandi, 2020] (see Figure 2.3(b)) to resolve cases where already-assembled parts impede the movements of subsequent parts.

2.1.3 Structural Stability

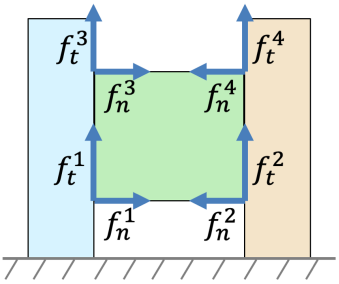
An assembly with rigid parts is structurally stable if it can preserve its form under external forces without collapse. Instability of assemblies can lead to catastrophic failure, e.g., in architecture, and thus must be analyzed and accounted for in the design process. Assemblies joined by permanent joints are usually very stable; e.g., certain glue is stronger than the part material. Such assemblies can then be analysed as a monolithic object using the finite element method. This thesis focuses on stability analysis of assemblies joined by non-permanent joints, which have intriguing property of encouraging disassembly. To analyze stability of assemblies, two critical conditions, *static equilibrium* and *global interlocking*, are defined

mathematically and identified computationally; see Figure 2.4(a&b). We review these two stability conditions below.

Static analysis. To identify whether an assembly is in equilibrium state under external forces or loads, there are two classes of static analysis methods: *linear elasticity analysis using finite element method (FEM)* and *rigid block equilibrium (RBE) method*. Shin et al. [Shin et al., 2016] proved that a small modification to the linear elastic FEM makes it equivalent to the RBE method to get the same answer to the same static analysis problem.

Given the geometry of a 3D assembly, a static equilibrium state means that there exists a network of interaction forces between the parts that can balance the external forces acting on each part; i.e., net force and net torque for each part equal zero [Whiting et al., 2009]. The RBE method combines equilibrium constraints for each part to build up a large linear system. An assembly is considered in static equilibrium if a solution of the linear system is found. Please refer to Section 3.2 for more details.

One limitation of the above method [Whiting et al., 2009] is its inability to accurately predict when parts will slide against one another, i.e., sliding failures [Yao et al., 2017a]; see the inset for a 2D example. A set of interface forces (see the dark blue arrows) can be found by [Whiting et al., 2009] to balance the gravity of the green part. However, the correct solution is that the green part should always fall under gravity with no resistance, no matter what coefficient of friction is used in this example. To address this limitation, Yao et al. [Yao et al., 2017a] proposed a *variational static analysis* method that amends the above method [Whiting et al., 2009] with a pair of variational principles from classical mechanics to exclude physically unrealizable forces.



The stability of masonry structures under lateral acceleration also can be analyzed based on static equilibrium [Ochsendorf, 2002, Zessin, 2012], which can be simulated with a tilt analysis that rotates the ground plane of the structure to apply both a horizontal and vertical acceleration to the structure; see Figure 2.4(c). For a given rotation axis, the critical tilt angle ϕ gives the minimum value of lateral acceleration to cause the structure to collapse, providing a measure of the structure's lateral stability [Shin et al., 2016, Yao et al., 2017a].

Interlocking test. In an interlocking assembly (with at least three parts), there is only one movable part, called the *key*, while all other parts as well as any subset of parts are immobilized relative to one another by their geometric arrangement [Song et al., 2012]; see Figure 2.4(b) for 2D examples. Starting from the key, the assembly can be gradually disassembled into individual parts by following a specific order. An assembly is called *recursive interlocking* if it has a unique (dis)assembly order, meaning that the assembly remains interlocking after the sequential removal of parts [Song et al., 2012]. An assembly is called *deadlocking* if there is no part that can be taken out from the assembly without collision (i.e., non-disassemblable). The

test for global interlocking essentially tries to identify if there exists a motion configuration that allows taking out any part(s) except the key from the assembly without collision. An assembly is considered to be interlocking if such a motion configuration does not exist. To test whether a given assembly is global interlocking by definition requires examining the immobilization of every subset of parts, which has exponential time complexity with respect to the number of parts. In this thesis, we introduce two polynomial-time methods that can efficiently test global interlocking. Please refer to Section 3.5 for more details.

2.2 Computational design of structurally stable assemblies

Computational design of assemblies generates different types of assemblies, including self-supporting assemblies (Section 2.2.1), global interlocking assemblies (Section 2.2.2), and topological interlocking assemblies (Section 2.2.3)

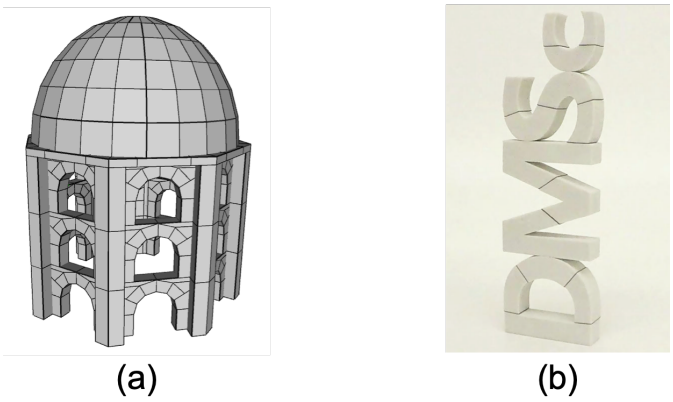


Figure 2.5 – Designing assemblies in equilibrium: (a) masonry building [Whiting et al., 2009], and (b) equilibrium puzzle [Frick et al., 2015]

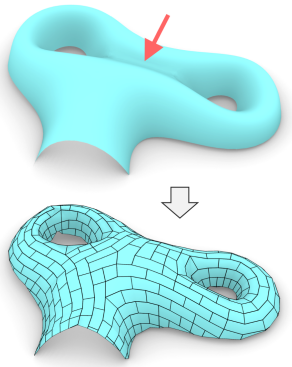
2.2.1 Assemblies in Equilibrium

An assembly is in static equilibrium if interaction forces between the parts can balance external forces acting on the assembly, mainly the gravity. These assemblies are common in architecture, furniture, and puzzles; see Figure 2.5. However, designing them is a non-trivial task as an equilibrium state depends on not only the parts (with integral joints) geometry, but also their geometric arrangement as well as the material property (i.e., friction coefficient).

The RBE method introduced in [Whiting et al., 2009] aims to test whether an assembly is in equilibrium, and to provide a quantitative measure about how unstable the structure is. Hence, a general way to design assemblies in equilibrium is using the RBE method to guide the search of a feasible configuration of parts geometry and arrangement. The other class of methods is more specific and focuses on designing free-form architectural assemblies like pavilion and dome, which relies on designing a *self-supporting surface*.

Design guided by the RBE method. The rigid body equilibrium (RBE) method described in Section 3.2 can not only test if a given assembly is in equilibrium under known external forces, but also provide a measure of the assembly's infeasibility to be in equilibrium when it fails the test. Whiting et al. [Whiting et al., 2009] integrated the RBE method with procedural modeling to design masonry structures that are in equilibrium under gravity, and they used a heuristic algorithm to search the parameter space such that the infeasibility measure can be decreased to zero; see Figure 2.5(a). Later, Whiting et al. [Whiting et al., 2012] extended this approach by using a gradient descent algorithm to explore the parameter space, in which a closed-form of the infeasibility measure's derivative with respect to the parts geometry variation is devised. Frick et al. [Frick et al., 2015] developed an interactive tool to design assemblies in equilibrium by decomposing a given 3D shape into a set of parts with planar cuts; see Figure 2.5(b). The tool keeps visualizing required tension forces computed by the RBE method and allows users to edit the planar cuts interactively until all the tension forces are removed.

Design based on self-supporting surfaces. According to the safety theorem [Heyman, 1966], an assembly is self-supporting (i.e., equilibrium under gravity) if there exists a thrust surface contained within the structure that forms a compressive membrane resisting the load applied to the assembly. Once the thrust surface, also called *self-supporting surface*, is ready, a self-supporting assembly can be easily generated by thickening and partitioning the surface into parts. Thrust Network Analysis (TNA) developed by Block and Ochsendorf [Block and Ochsendorf, 2007] is a well-known graphical approach for form exploration of self-supporting surfaces. Inspired by this work, the graphics community has proposed a number of geometry processing methods to approximate free-form surfaces with self-supporting ones; see [Ma et al., 2019] for an overview of these methods. In particular, Panozzo et al. [Panozzo et al., 2013] not just generated self-supporting surfaces, but also fabricated corresponding self-supporting assemblies to verify the stability. The inset figure shows an example target surface and the resulting self-supporting assembly. Note that a local geometric feature in the target surface (highlighted by a red arrow) is deformed to make the surface self-supporting.



In practice, equilibrium under gravity might be insufficient since an assembly could be exposed to different forces (e.g., live loads). This motivates the stability measure based on the tilt analysis described in Section 2.1.3, where an assembly can be in equilibrium for a cone of gravity directions, as well as the work of optimizing free-form architectural assemblies to maximize the stability.

2.2.2 Interlocking Assemblies

Compared with assemblies in equilibrium, interlocking assemblies are more structurally stable under unpredictable external forces yet enforce higher complexity on the parts geometry and their joining. The main challenge of designing interlocking assemblies is to ensure two conflicting properties simultaneously: interlocking and disassemblable. This is because interlocking requires strict joining to restrict relative movement among parts yet disassemblability demands at least one collision-free plan to separate the parts (i.e., not deadlocking).

A straightforward way to design interlocking assemblies is to exhaustive search all possible configurations and perform the interlocking test. This method has been tried by Cutler [Cutler, 1978] in the late 1970s to discover new six-piece interlocking configurations, which took almost three years to search a cubical volume of less than 4^3 voxels due to the combinatorial complexity. Later, Xin et al. [Xin et al., 2011] developed a retargeting approach to create 3D interlocking puzzles by replicating and connecting multiple instances of an existing six-piece interlocking burr structure within a given target shape. Until recently, a few computational methods have been developed to design new interlocking assemblies, making it possible to increase the number of parts and to enrich geometric forms of assemblies significantly; see Figure 2.6. The design problem in these works is formulated as, *shape decomposition* or *joint planning*, according to the given input.

Shape decomposition. When the input is a target shape, computational design of interlocking assemblies can be formulated as a *shape decomposition* problem. A typical approach to address this problem is to construct Local Interlocking Groups (LIGs), which are a subset of connected parts that are locked by a specific key in the group, and to enforce dependency among these LIGs. The advantage of this approach is that the resulting assemblies are guaranteed to be globally interlocking. Yet, the limitation is that the explored search space is restricted to a small subset of all possible interlocking configurations. Song et al. [Song et al., 2012] first proposed this approach and used it to construct 3D interlocking puzzles. Given a voxelized 3D shape, their method iteratively extracts pieces while enforcing a local interlocking condition among every three consecutive pieces; see Figure 2.6(a). This method was later extended to handle smooth non-voxelized shapes for 3D printing [Song et al., 2015] and to design 3D steady dissection puzzles [Tang et al., 2019].

Joint planning. When the input is a set of initial parts without joints, designing interlocking assemblies can be formulated as a *joint planning* problem. The goal is to plan and construct a set of predefined joints (e.g., mortise and tenon joint in Figure 2.1(c)) on the initial parts to make them interlocking. Fu et al. [Fu et al., 2015] focused on plate structures such as furniture that have been initially partitioned into parts, and computed an interlocking joint configuration following the LIG-based approach, where each LIG has only 3 or 4 parts and thus the joint configuration in each LIG can be searched exhaustively; see Figure 2.6(b). This method has been extended to interlock 2D laser-cut parts into a convex polyhedron [Song et al., 2016] and to design reconfigurable furniture with multi-key interlocking [Song et al.,

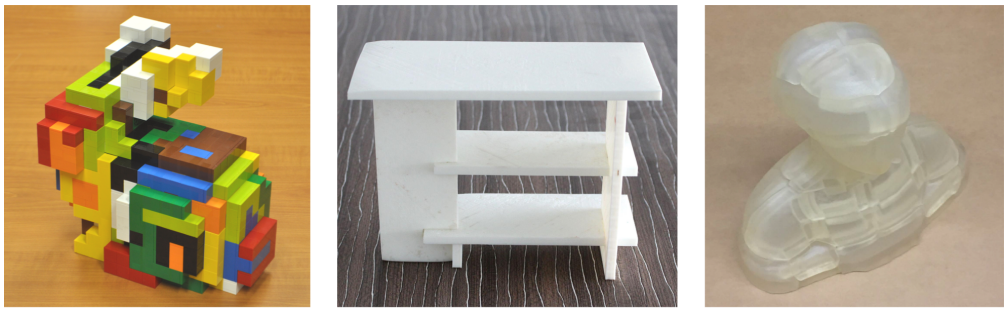


Figure 2.6 – Designing interlocking assemblies: (a) interlocking puzzle [Song et al., 2012], (b) furniture [Fu et al., 2015], and (c) object shell assembly for 3D printing [Yao et al., 2017b].

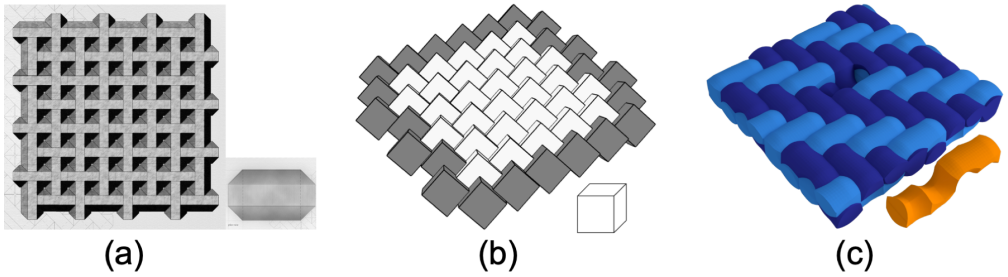


Figure 2.7 – Building planar assemblies with tilable blocks: (a) the Abeille flat vault; (b) a topological interlocking assembly with cubes [Dyskin et al., 2003a], where the boundary is highlighted in black; and (c) an assembly with bi-axial weaving patterns built with a single tileable block (in orange) [Krishnamurthy et al., 2021]

2017]. Yao et al. [Yao et al., 2017b] designed interlocking shell assemblies for 3D printing by using a randomized search with pruning to generate candidate joint configurations and verifying their global interlocking by using physically based simulation; see Figure 2.6(c).

Global interlocking might impose too strict constraints on the assembly's geometry, as real assemblies usually do not have to experience arbitrary external forces. Hence, some research works relax the constraint of global interlocking, e.g., by allowing multiple keys in the final assembly [Song et al., 2017] or allowing parts to be immobilized by using geometric arrangement together with friction [Tang et al., 2019].

2.2.3 Topological Interlocking Assemblies

Constructing structurally stable assemblies with tileable blocks is intriguing in engineering and architecture, and has recently attracted great interest in the graphics community. At the end of 17th century, Joseph Abeille discovered that identical tetrahedrons truncated at two opposite edges can be arranged to form a planar stable assembly, known as Abeille flat vaults; see Figure 2.7(a). Since then, several variants of these structures have been invented and studied under the name of *topological interlocking (TI) assemblies* [Dyskin et al., 2003a].

Physical experiments conducted on TI assemblies have shown that they possess interesting and unusual mechanical properties such as high strength and toughness [Mirkhalaf et al., 2018] and damage confinement [Siegmund et al., 2016].

TI assemblies typically consist of a single tileable element that can be repeatedly arranged in such a way that the whole structure can be held together by a fixed boundary, while elements are kept in place by mutual blocking. Kanel-Belov et al. [Kanel-Belov et al., 2010] proposed a constructive approach to generate TI assemblies based on a tiling of the middle plane, where the single tileable block could be one of the five platonic solids; see Figure 2.7(b) for an example. Weizmann et al. [Weizmann et al., 2017] extended this approach and explored different 2D tessellations (regular, semi-regular and non-regular tessellations) to discover new TI blocks for building floors. Rather than relying on 2D tessellations, other researchers [Krishnamurthy et al., 2021,Akleman et al., 2020] made a connection between planar assemblies and bi-axial weaving patterns. They generated tileable blocks (with curved faces) by Voronoi partitioning of space using curve segments whose arrangement follows the weaving patterns; see Figure 2.7(c).

3 Kinematic-Based Stability Analysis

Several computational methods for testing various types of structural stability of assemblies with rigid parts are discussed in this chapter. Section 3.1 defines the notations used in the contact computation that are prerequisites for all tests. Section 3.2 reviews the previous force-based equilibrium method and Section 3.3 presents our new kinematic-based equilibrium method. The joint motion cones and motion-based representation which are used to accelerate our shape optimization algorithm in Chapter 6 are discussed in Section 3.4. We propose two distinct globally interlocking tests and establish a connection between globally interlocking and static equilibrium in Section 3.5. Lastly, Section 3.6 presents the lateral stability measurement and depicts a full picture of the stability spectrum. We ignore friction in all of our tests, which makes our test more conservative and independent from material selection.

3.1 Contact Discretization

We assume that our assembly $\boldsymbol{P}$ has N component parts and M contacts. This thesis mainly covers three contact types: the planar, piecewise planar, and curved contacts. To test structural stability numerically, we sample points on the continuous contact surface C_l between P_i and

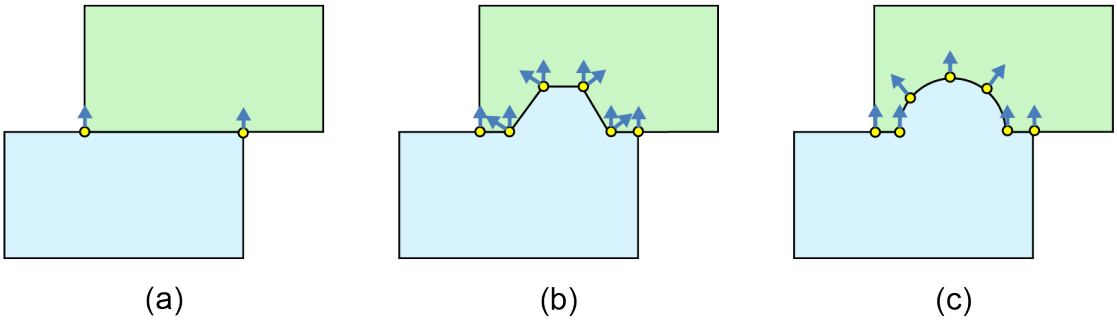


Figure 3.1 – Contact sampling methods for (a) the planar contact, (b) the piecewise planar contact and (c) the curved contact.

P_j. The sampled points are named $\{\boldsymbol{c}_l^k\}$ where $1 \le k \le v_l$, and $\boldsymbol{n}_l^k$ is the normal of the contact surface at the point $\boldsymbol{c}_l^k$. We enforce that $\boldsymbol{n}_l^k$ always points towards the part with the larger index. To simplify notation, we assume $i < j$ and thus $\boldsymbol{n}_l^k$ always points towards P_j. We apply different sampling strategies for different contact types to obtain faithful analyzing results.

- *Planar contact:* The contact surface is basically a planar polygon. We set the vertices of the polygon to be the sampled contact points. The contact normals are all equal to the face normal of the polygon; see Figure 3.1(a).
- *Piecewise planar contact:* We decompose the contact surface into a set of planar contacts and sample each planar contact using the above mentioned method. Figure 3.1(b).
- *Curved contact:* We approximate the curved contact surface as a piecewise planar surface. In practice, our method simply sample points uniformly on the contact to obtain the contact points and normals. To obtain a good approximation, we typically require 50 (200) sample points per 2D (3D) joint; see Figure 3.1(c).

Note that the number of sample points per curved contact is much higher than the number of vertices (typically 4) per planar contact, increasing the computational cost significantly for our numerical tests.

3.2 Force-based Equilibrium Method

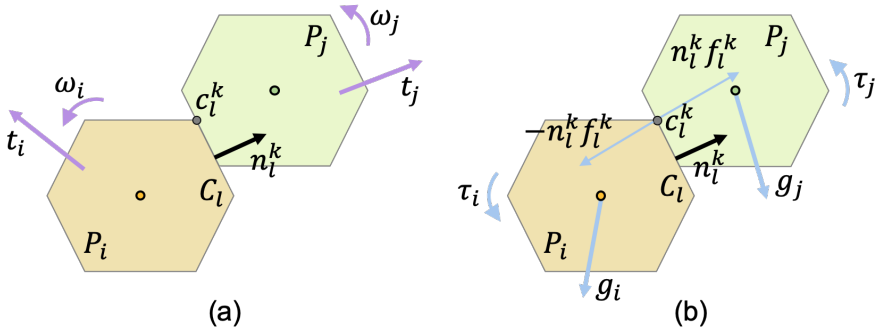


Figure 3.2 – Supposed that two parts P_i and P_j have a planar contact C_l, $\boldsymbol{c}_l^k$ is a sampled contact point and $\boldsymbol{n}_l^k$ is the contact normal at $\boldsymbol{c}_l^k$. (a) P_i and P_j should not collide with each other at the contact during their movement, e.g., translation $\boldsymbol{t}_i$ and rotation $\boldsymbol{\omega}_i$ of P_i. (b) Each block P_i is in equilibrium if there exists a system of interaction forces (e.g., $-\boldsymbol{n}_l^k f_l^k$) that balance the external force $\boldsymbol{g}_i$ and torque $\boldsymbol{\tau}_i$ acting on it.

Test of equilibrium Let $\boldsymbol{g}_i$ be the external force and $\boldsymbol{\tau}_i$ the torque acting on part P_i of an assembly $\boldsymbol{P}$; see Figure 3.2(b). Noted that in the original paper [Whiting et al., 2009] the rotation center of part P_i is chosen to be the P_i's center of gravity since the torque $\boldsymbol{\tau}_i = \boldsymbol{0}$ when gravity

is the only external force. To have a unified definition with our kinematic-based equilibrium method, we choose the world origin O to be the rotation centers of all parts. We show that this definition leads to a symmetric formulation for our kinematic-based equilibrium method in Section 3.3. Because of our new rotation center, the gravity $\boldsymbol{G}$ can generals torque $\boldsymbol{\tau}_i = \boldsymbol{R}_i \times \boldsymbol{G}$ for part P_i where $\boldsymbol{R}_i$ is the P_i's center of mass.

Let's consider the general case where the external force $\boldsymbol{w}_i = [\boldsymbol{g}_i^T, \boldsymbol{\tau}_i^T]^T$ acting on P_i. Given all the external forces and torques $\boldsymbol{w} = [\boldsymbol{w}_1^T, \ldots, \boldsymbol{w}_n^T]^T$, The goal of the force-based equilibrium method [Whiting et al., 2009] is to find a network of interaction forces between the parts that can balance the external forces and torques acting on each part. In Section 3.1, the contact C_l is represented by a set of contact points and normals $\{(\boldsymbol{c}_l^k, \boldsymbol{n}_l^k)\}$. The interaction forces therefore are discretized as a finite number of forces at each contact points $\boldsymbol{c}_l^k$. Since we assume rigid parts and ignore friction, interaction forces only include compression forces along the contact normal direction; see Figure 3.2-b.

For a vertex $\boldsymbol{c}_l^k$ in C_l between P_i and P_j $(i < j)$, we denote the contact force size as f_l^k $(f_l^k \geq 0)$. Hence, the contact force applied on P_i is $-f_l^k\ \boldsymbol{n}_l^k$, and consequently $f_l^k\ \boldsymbol{n}_l^k$ on P_j. Static equilibrium conditions require that the net force and the net torque for each block P_i are equal to zero:

$$\sum_{l=1}^{M} \sum_{k=1}^{v_l} -\boldsymbol{n}_l^k\ f_l^k\ =\ -\boldsymbol{g}_i \tag{3.1}$$

$$\sum_{l=1}^{M} \sum_{k=1}^{v_l} -(\boldsymbol{c}_l^k \times \boldsymbol{n}_l^k) \cdot\ f_l^k\ =\ -\boldsymbol{\tau}_i \tag{3.2}$$

Combining the equilibrium constraints in Equation 3.1 and 3.2 for each block gives a linear system of equations:

$$\boldsymbol{A}_{\text{eq}}\ \boldsymbol{f} = -\boldsymbol{w} \qquad \text{s.t. } \boldsymbol{f} \geq \boldsymbol{0} \tag{3.3}$$

where $\boldsymbol{f}$ represents the unknown interaction forces in the assembly (i.e., contact force sizes at each vertex of each contact C_l), $\boldsymbol{A}_{\text{eq}}$ is the matrix of coefficients for the equilibrium equations [Whiting et al., 2009], and $\boldsymbol{w}$ represents the external forces and torques acting on the system. The system has a solution meaning the assembly $\boldsymbol{P}$ is stable against the external forces $\boldsymbol{w}$. Commonly, finding a feasible solution for a linear system is formulated as a convex optimization [Whiting et al., 2009] that inspires the measure of infeasibility.

Measure of infeasibility When an assembly is not in equilibrium, Whiting et al. [Whiting et al., 2009] proposed a method to measure its distance to a feasible solution (i.e., an equilibrium state) by introducing tension forces that act as "glue" at part contact interfaces to hold the

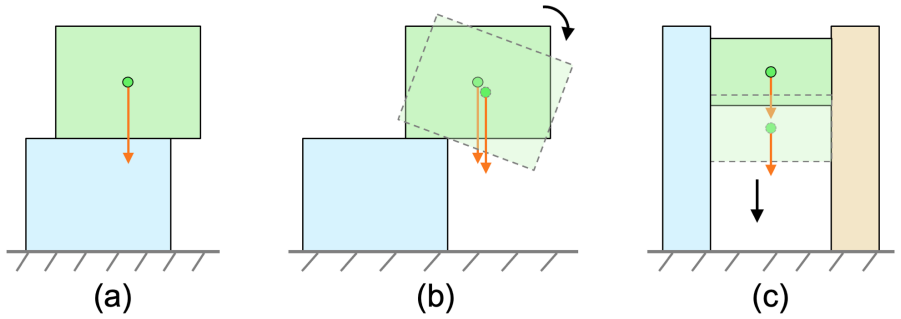


Figure 3.3 – Kinematic-based equilibrium method for static analysis. In these examples, external forces are gravity (in orange) and ground supporting forces only, and friction is ignored. (a) The assembly is in equilibrium as we cannot find any parts motion that satisfies Equations 3.9 and 3.10. (b&c) The two assemblies are not in equilibrium, where a parts motion solution that satisfies Equations 3.9 and 3.10 is shown as a dashed boundary and a black arrow.

assembly together and penalizing these tension forces:

$$\begin{aligned} \min_{\boldsymbol{f}^+,\boldsymbol{f}^-} \quad & \frac{1}{2}\boldsymbol{f}^- \cdot \boldsymbol{f}^- \\ \text{s.t.} \quad & \boldsymbol{A}_{\text{eq}}\,(\boldsymbol{f}^+ - \boldsymbol{f}^-) = -\boldsymbol{w}, \\ & \boldsymbol{f}^+, \boldsymbol{f}^- \geq 0 \end{aligned} \tag{3.4}$$

where $\boldsymbol{f}^+$ is the positive parts (i.e., compression forces) of the contact forces $\boldsymbol{f}$, and $\boldsymbol{f}^-$ is the negative parts (i.e., tension forces) of $\boldsymbol{f}$. The quadratic programming in Equation 3.4 enables to measure infeasibility and to test equilibrium in a unified way, since an assembly is in equilibrium if the infeasibility measure equals zero. Existing algorithms to solve the quadratic programming typically have a polynomial complexity with respect to the number of its variables (i.e., M).

Remark on assemblies with curved contacts As mentioned in Section 3.1, for assemblies with curved contacts, we approximate its contacts with in conscience planar faces. If there exists a solution $\boldsymbol{f}$ for Equation 3.3, the assembly is considered as in equilibrium. However, non-existence of such a solution does not mean that the assembly is not in equilibrium since our interaction forces are only samples of the actual forces.

3.3 Kinematic-based Equilibrium Method

As an alternative to the force-based approach, we propose a motion-based equilibrium method, inspired by static-kinematic duality.

Test of equilibrium Suppose each part P_i can translate and rotate [1] freely in 3D space. We denote the linear velocity of P_i as $\boldsymbol{t}_i$, the angular velocity of P_i as $\boldsymbol{\omega}_i$, and the local motion of P_i as a 6D spatial vector $\hat{\boldsymbol{v}}_i = [\boldsymbol{t}_i^T, \boldsymbol{\omega}_i^T]^T$; see Figure 3.2-a. For an arbitrary contact point $\boldsymbol{c}_l^k$ on

[1] The center of rotation of part P_i is chosen to be the world origin O

the contact C_l between P_i and P_j, $\hat{\boldsymbol{v}}_i$ and $\hat{\boldsymbol{v}}_j$ will cause $\boldsymbol{c}_l^k$ to undergo an infinitesimal motion together with P_i and P_j respectively:

$$\boldsymbol{v}_i^c = \boldsymbol{t}_i + \boldsymbol{\omega}_i \times \boldsymbol{c}_l^k \tag{3.5}$$

$$\boldsymbol{v}_j^c = \boldsymbol{t}_j + \boldsymbol{\omega}_j \times \boldsymbol{c}_l^k \tag{3.6}$$

During the parts movement, the constraint is to avoid collision at their contacts. Since our interlocking test considers only infinitesimal motions of each block, we assume that the contact normal $\boldsymbol{n}_l^k$ remain fixed during the test. Hence, the collision-free constraint between P_i and P_j at contact point $\boldsymbol{c}_l^k$ can be modeled as:

$$(\boldsymbol{v}_j^c - \boldsymbol{v}_i^c) \cdot \boldsymbol{n}_l^k \geq 0 \tag{3.7}$$

By substituting Equations 3.5&3.6 in Equation 3.7, we obtain:

$$\begin{bmatrix} -(\boldsymbol{n}_l^k)^T & -(\boldsymbol{c}_l^k \times \boldsymbol{n}_l^k)^T & (\boldsymbol{n}_l^k)^T & (\boldsymbol{c}_l^k \times \boldsymbol{n}_l^k)^T \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{v}}_i \\ \hat{\boldsymbol{v}}_j \end{bmatrix} \geq \boldsymbol{0} \tag{3.8}$$

Equation 3.8 describes the constraint of a point-plane contact between P_i and P_j. By stacking the point-plane constraint in Equation 3.8 for each vertex of each contact in the assembly $\boldsymbol{P}$, we obtain a system of linear inequalities:

$$\boldsymbol{B}_{\text{in}} \cdot \hat{\boldsymbol{v}} \geq \boldsymbol{0} \tag{3.9}$$

where $\hat{\boldsymbol{v}}$ is the generalized velocity of assembly $\boldsymbol{P}$, and $\boldsymbol{B}_{\text{in}}$ is the matrix of coefficients for the non-penetration constraints among the blocks in the system. The parts that touch the ground are fixed by setting their velocities to be zero. Note that the coefficient matrix $\mathbf{B}_{\text{in}}$ in Equation 3.9 and $\mathbf{A}_{\text{eq}}$ in Equation 3.3 are transposed to each other, according to the well-known close relation between velocity kinematics and statics [Davidson and Hunt, 2004].

When the inequality system in Equation 3.9 does not have any non-zero solution, the assembly is considered as deadlocking, meaning no part can move in the assembly. Otherwise, parts are movable. If the part movement is driven by the given external forces (e.g., gravity), the assembly should not be in equilibrium; see Figure 3.3(b&c) for two examples. Inspired by this observation, we add an additional constraint for the equilibrium test:

$$\boldsymbol{w}^T \cdot \hat{\boldsymbol{v}} > 0 \tag{3.10}$$

where $\boldsymbol{w}$ is the generalized external forces defined in Equation 3.3, and $\boldsymbol{w}^T \cdot \hat{\boldsymbol{v}}$ can be understood as the total power created by the external forces $\boldsymbol{w}$ for a given motion configuration $\hat{\boldsymbol{v}}$.

For an assembly to be in equilibrium, there should not exist any solution $\hat{\boldsymbol{v}}$ that satisfies the

two linear constraints described in Equations 3.9 and 3.10; see Figure 3.3(a) for an example. This statement can be proved by showing that it is actually equivalent to the force-based equilibrium test (Equations 3.3) using Farkas' lemma [Farkas, 1902].

Measure of infeasibility Similar to our test of equilibrium, we propose a measure of infeasibility based on the parts motion $\hat{\boldsymbol{v}}$:

$$\begin{aligned} \max_{\hat{\boldsymbol{v}}} \quad & \boldsymbol{w}^T \hat{\boldsymbol{v}} - \frac{1}{2} \hat{\boldsymbol{v}}^T \hat{\boldsymbol{v}} \\ \text{s.t.} \quad & \boldsymbol{B}_{\text{int}}\, \hat{\boldsymbol{v}} \geq 0 \end{aligned} \tag{3.11}$$

where the second term in the objective function is a regularization to prevent the energy from becoming infinity. The assembly should be in equilibrium when the infeasibility measure equals zero.

To understand the relation between our measure (Equation 3.11) and that of [Whiting et al., 2009] (Equation 3.4), we apply the strong duality theorem [Boyd and Vandenberghe, 2004] to our measure, and obtain the following formulation:

$$\begin{aligned} \min_{\boldsymbol{f}, \boldsymbol{s}} \quad & \frac{1}{2} \boldsymbol{s}^T \boldsymbol{s} \\ \text{s.t.} \quad & \boldsymbol{A}_{\text{eq}}\, \boldsymbol{f} + \boldsymbol{s} = -\boldsymbol{w}, \\ & \boldsymbol{f} \geq 0 \end{aligned} \tag{3.12}$$

where $\boldsymbol{s}$ is additional forces/torques required to make each part in balance; see the supplementary material for a proof (Appendix C). This dual formulation of our measure can be understood as an alternative of the measure in [Whiting et al., 2009].

3.4 Motion-Based Representation

In this section, we reformulated the non-penetration constraints (Equation 3.9) as a motion-based representation used by shape optimization framework discussed in Chapter 6. The abstract representation is a part graph enhanced by the motion cones. Section 3.4.1 describes the way of analyzing the motion space of contacts, and Section 3.4.2 illustrates the construction of the motion-based graph.

3.4.1 Motion Space Analysis of Contact

Assuming part P_i is fixed, our motion space analysis of contact C_l considers all possible infinitesimal rigid motion to take out P_j from P_i without collision on contact C_l. This motion space of part P_j, denoted as $\boldsymbol{V}(C_j)$ (abbreviated as $\boldsymbol{V}$ in the following equations), abstracts the ability of contact C_l to restrict relative part motion.

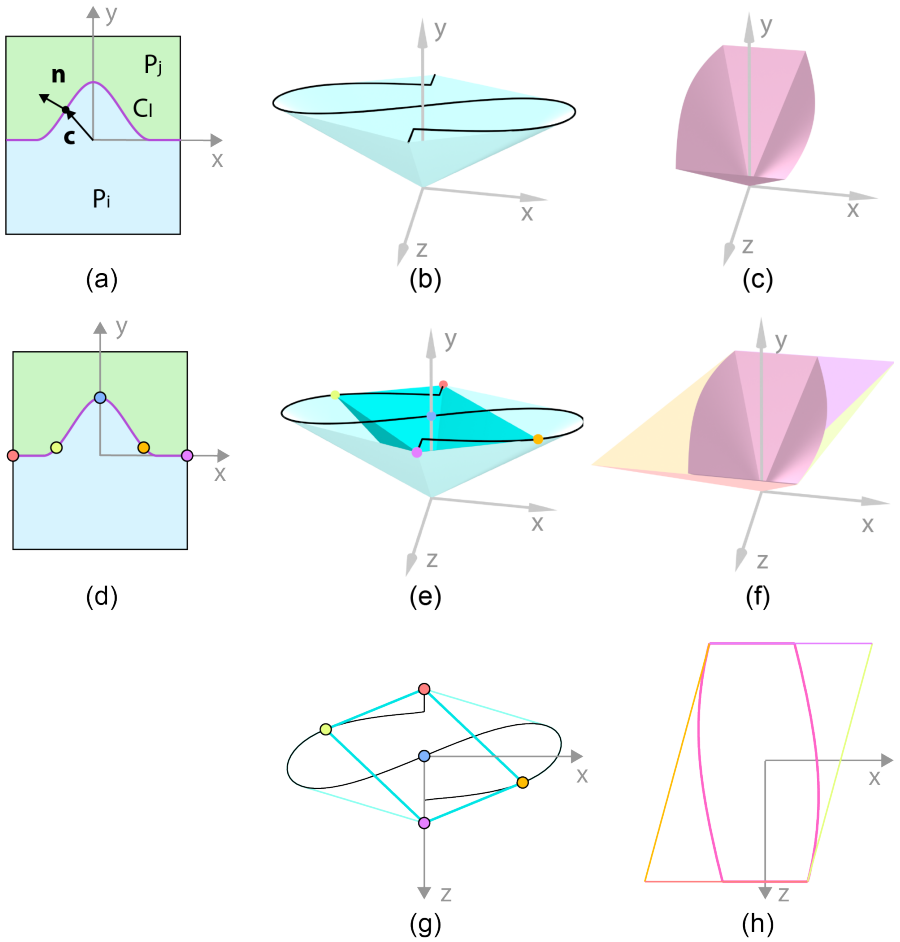


Figure 3.4 – Motion space analysis of (a) a contact C_l in a local coordinate frame. (b) Generalized normal curve $\hat{\boldsymbol{N}}$ (in black) and its minimum convex cone envelope (in light cyan). (c) Motion cone $\boldsymbol{V}$. (d) Sampling the curved contact with five points. (e) Minimum convex cone envelope (dark cyan) of the sampled generalized normals is contained in the original minimum cone envelope. (f) Motion cone allowed by the sampled points (pyramid with four colored faces) contains the original motion cone, where the color of each face matches that of the corresponding sampled contact point and generalized normal. (g) Conic sections of the two cone envelopes in (e) by cut plane $y = 1$. (h) Conic sections of the two motion cones in (f) by cut plane $y = 1$.

We express the infinitesimal rigid motion of part P_j using the generalized velocity $\hat{\boldsymbol{v}}_j$, which is composed of both linear velocity $\boldsymbol{t}_j$ and angular velocity $\boldsymbol{\omega}_j$. For any contact point $\boldsymbol{c}$ on the curved contact of C_l and its contact normal as $\boldsymbol{n}$; see Figure 3.4(a). The motion space $\boldsymbol{V}$ can be obtained by solving a linear inequality system that represents non-collision constraints for every point on the contact of C_l [Wilson and Matsui, 1992]

$$\begin{aligned} \boldsymbol{V} &= \{\hat{\boldsymbol{v}} \mid \hat{\boldsymbol{n}} \cdot \hat{\boldsymbol{v}} \geq 0,\ \forall \hat{\boldsymbol{n}} \in \hat{\boldsymbol{N}}\} \\ \hat{\boldsymbol{n}} &= \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{c} \times \boldsymbol{n} \end{bmatrix} \end{aligned} \tag{3.13}$$

where $\hat{\boldsymbol{n}}$ is the generalized normal of contact point $\boldsymbol{c}$ with normal $\boldsymbol{n}$, and $\hat{\boldsymbol{N}}$ is the generalized normal space of contact C_l.

For a smooth 2D contact, the generalized normal space $\hat{\boldsymbol{N}}$ is a curve in 3D space (2 dimensions for normal $\boldsymbol{n}$ and 1 dimension for $\boldsymbol{c} \times \boldsymbol{n}$); and the motion space $\boldsymbol{V}$ is a cone in 3D space (2 dimensions for translation and 1 dimension for rotation); see Figure 3.4(b&c). We make a connection between the generalized normal space $\hat{\boldsymbol{N}}$ of contact C_l and the motion space $\boldsymbol{V}$ of part P_j based on the dual cone concept in convexity theory, and formulate the following theorems and lemma; please refer to Section 2.6.1 in [Boyd and Vandenberghe, 2004] for a derivation.

Theorem 1 (Dual Cone) *The dual cone $\boldsymbol{C}^*$ of a set $\boldsymbol{C}$ in R^n, defined as*

$$\boldsymbol{C}^* = \left\{ \boldsymbol{y} \in R^n : \boldsymbol{y} \cdot \boldsymbol{x} \geq 0, \quad \forall \boldsymbol{x} \in \boldsymbol{C} \right\},$$

is always convex, even when the original set $\boldsymbol{C}$ is not.

According to Theorem 1 and Equation 3.13, the motion space $\boldsymbol{V}$ can be viewed as the dual cone of the generalized normal space $\hat{\boldsymbol{N}}$, i.e., $\boldsymbol{V} = \hat{\boldsymbol{N}}^*$. Hence, the motion space $\boldsymbol{V}$ must be convex no matter whether the generalized normal space $\hat{\boldsymbol{N}}$ is convex or not; see Figure 3.4(c). Replacing the generalized normal space $\hat{\boldsymbol{N}}$ with its minimum convex cone envelope (see the cyan cone in Figure 3.4(b)) will not affect its dual cone. This minimum convex cone envelope of $\hat{\boldsymbol{N}}$ is called the generalized normal cone, or simply $\texttt{cone}(\hat{\boldsymbol{N}})$.

A motion cone can be exactly defined by a cut plane and a cross section from that cut (called the conic section); see Figure 3.4(h). For a symmetric 2D contact like Figure 3.4(a), the cut plane is simply chosen as $y = 1$. However, for general motion cones, it is critical to choose a proper cut plane such that the conic section is finite; please refer to the supplementary material for details (Appendix B). Similarly, the generalized normal cone $\texttt{cone}(\hat{\boldsymbol{N}})$ also can be represented and visualized by a cut plane and the corresponding conic section; see Figure 3.4(b&g).

Theorem 2 (Sampling) *If $\tilde{\boldsymbol{N}}$ is a subset of space $\hat{\boldsymbol{N}}$, the corresponding dual cone $\tilde{\boldsymbol{V}}$ of $\tilde{\boldsymbol{N}}$ should contain $\hat{\boldsymbol{N}}^*$ or equivalently $\boldsymbol{V}$:*

$$\tilde{\boldsymbol{N}} \subseteq \hat{\boldsymbol{N}} \Rightarrow \boldsymbol{V} \subseteq \tilde{\boldsymbol{V}}$$

Theorem 2 enables us to compute an approximation $\tilde{\boldsymbol{V}}$ of the motion space $\boldsymbol{V}$ numerically. First, we sample the generalized normal space $\hat{\boldsymbol{N}}$ to obtain a finite subset $\tilde{\boldsymbol{N}}$; see the five colored points in Figure 3.4(d&e). Next, an approximated motion cone $\tilde{\boldsymbol{V}}$ is computed by intersecting a finite number of half-spaces described by Equation 3.13; see the pyramid with four colored planes in Figure 3.4(f). Note that the half-space corresponding to the dark blue point is not shown since it does not contribute to the approximated motion cone $\tilde{\boldsymbol{V}}$. Theorem 2 guarantees that the approximated motion cone $\tilde{\boldsymbol{V}}$ must cover all possible motions in the original motion

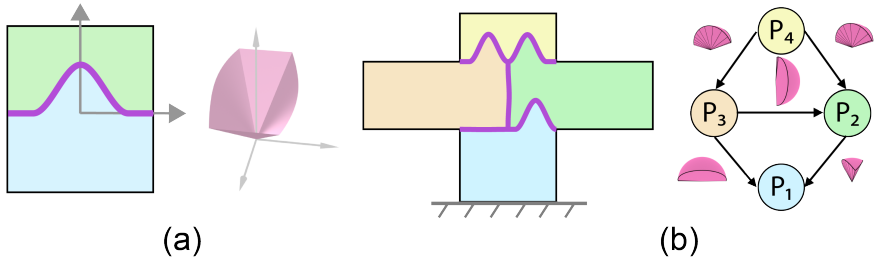


Figure 3.5 – (a) The 3D joint motion cone of a 2D curved joint and (b) the motion-based representation of a four-part assembly where the 3D cones at graph edges are the joint motion cones.

cone $\boldsymbol{V}$; see Figure 3.4(f). In our experiments, we find that 50 (200) sample points per 2D (3D) contact provide a good approximation of the motion cone $\boldsymbol{V}$.

Lemma 1 *If a motion cone $\boldsymbol{V}$ is contained in a given space $\boldsymbol{V}_0$, it is equivalent to require that $cone(\hat{\boldsymbol{N}})$ includes the dual cone of $\boldsymbol{V}_0$:*

$$\boldsymbol{V} \subseteq \boldsymbol{V}_0 \Leftrightarrow \boldsymbol{V}_0^* \subseteq cone(\hat{\boldsymbol{N}})$$

Lemma 1 can be used to generate joint geometry that have a bounded motion cone $\boldsymbol{V}_0$. For instance, if the motion cone $\boldsymbol{V}_0$ is chosen to be a circular cone with an angle α (i.e., $\{(\boldsymbol{x}, y) : \|\boldsymbol{x}\| \leq \tan(\alpha) y\}$), then its dual cone $\boldsymbol{V}_0^*$ is simply $\{(\boldsymbol{x}, y) : \|\boldsymbol{x}\| \leq \frac{y}{\tan\alpha}\}$, which is also a circular cone with an angle $\frac{\pi}{2} - \alpha$. Classic geometric processing algorithms can be applied to find joint geometry whose generalized normal cone contains the dual circular cone $\boldsymbol{V}_0^*$.

3.4.2 Motion Graph

We can reformulate the non-penetration constraints Equation 3.11 to a symmetric form.

$$\begin{bmatrix} -\boldsymbol{n}^T & -(\boldsymbol{c}\times\boldsymbol{n})^T & \boldsymbol{n}^T & (\boldsymbol{c}\times\boldsymbol{n})^T \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{v}}_i \\ \hat{\boldsymbol{v}}_j \end{bmatrix} \geq \boldsymbol{0} \tag{3.14}$$

$$\Leftrightarrow \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{c}\times\boldsymbol{n} \end{bmatrix}^T (\hat{\boldsymbol{v}}_j - \hat{\boldsymbol{v}}_i) \geq \boldsymbol{0} \tag{3.15}$$

Collecting all inequalities for all the contact points $\boldsymbol{c}_l^k$ $(k = 1, \cdots, v_k)$, the inequality system is equivalent to:

$$\hat{\boldsymbol{v}}_j - \hat{\boldsymbol{v}}_i \in \boldsymbol{V}(C_l) \tag{3.16}$$

Hence, the infeasibility measure (Equation 3.11) can be reformulated as:

$$\begin{aligned} E(\boldsymbol{w}, \{\boldsymbol{V}(C_l)\}) \;=\; \max_{\hat{\boldsymbol{v}}_i} \quad & \boldsymbol{w}^T \hat{\boldsymbol{v}} - \frac{1}{2}\hat{\boldsymbol{v}}^T \hat{\boldsymbol{v}} \\ \text{s.t.} \quad & \hat{\boldsymbol{v}}_j - \hat{\boldsymbol{v}}_i \in \boldsymbol{V}(C_l), \;\; \text{for each } C_l \end{aligned} \tag{3.17}$$

The part graph of an assembly is an undirected graph representing the connectivity between parts. Each part P_i is a node and the contact C_l is an edge between node P_i and P_j. Our motion graph, which is a directed graph, regards the constraint (Equation 3.16) as an direct edge from P_j to P_i associated with the motion cone $V(C_l)$; see Figure 3.5. In some assemblies, the contacts between two parts could be more than one, and thus the edges between two nodes could be more than one as well. To simplify our motion graph, we often use one direct edge with the motion space $V_{i,j}$, which is the intersection of all motion spaces $V(C_l)$ between part P_i and P_j, to replace the many edges between P_i and P_j.

3.5 Kinematic-Based Interlocking Test

In a globally interlocking assembly, parts must follow specific orders to be assembled into the target object. Once assembled, there is only one movable part, called the *key*, while all other parts, as well as any subset of parts, are immobilized relative to one another [Song et al., 2012]. Assumed that the parts are sorted in an ascending way by their assembling orders, P_n refers to the last part to be assembled, which is the key part.

To test whether a given assembly is globally interlocking by definition requires examining the immobilization of every subset of parts, which has exponential time complexity with respect to the number of parts. In this section, we will introduce two polynomial-time interlocking testing methods, both can be derived from our motion-based representation. The method discussed in Section 3.5.1 utilizes the inequalities in Equation 3.9 that is applicable for testing most assemblies. The method presented in Section 3.5.2 is based on the directional blocking graphs, a simplified version of our motion-based representation, which is only applicable for testing voxelized assemblies or assemblies with single-direction joints. Finally, we establish a connection between globally interlocking with static equilibrium in Section 3.5.3.

3.5.1 Inequality-based Interlocking Test

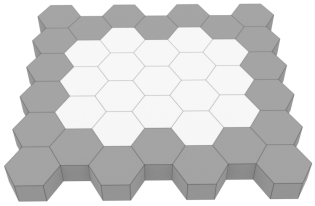
If parts can be disassembled by rigid motions without causing collision among themselves, even the motions are infinitesimal, the structure is not global interlocking. Equation 3.9 formulates these collision constraints as a linear inequality system, which has non-zero solutions if the assembly is not globally interlocking. However, the linear system always has trivial non-zero solutions (i.e., the key part must be movable). To exclude these obvious solutions, we fix the key part P_N as well as any of the rest parts, saying P_{N-1}, by setting their velocities to be zero.

$$\boldsymbol{B}_{\text{in}}\hat{\boldsymbol{v}} \geq \mathbf{0}, \qquad \hat{\boldsymbol{v}}_N = \hat{\boldsymbol{v}}_{N-1} = \mathbf{0} \tag{3.18}$$

The assembly $\boldsymbol{P}$ is globally interlocking if Equation 3.18 does not have any non-zero solution. We solve the Equation 3.18 using a linear programming.

$$\begin{aligned} \max_{\boldsymbol{t}=\{t_i\},\hat{\boldsymbol{v}}} \quad & \sum t_i \\ \text{s.t.} \quad & 0 \le t_i \le 1, \\ & \boldsymbol{B}_{\text{in}}\hat{\boldsymbol{v}} \ge \boldsymbol{t}, \\ & \hat{\boldsymbol{v}}_N = \hat{\boldsymbol{v}}_{N-1} = \boldsymbol{0} \end{aligned} \tag{3.19}$$

If the optimal solution of Eq. 3.19 is non-zero, the assembly is not globally interlocking. Otherwise, the assembly is globally interlocking except for one special case where the optimal solution of Eq. 3.19 is zero yet $\boldsymbol{B}_{\text{in}} \cdot \hat{\boldsymbol{v}} = \boldsymbol{0}$ actually has non-zero solutions. The inset shows such an example. In this assembly, each component part can move vertically (i.e., not interlocking) without any collision among the parts (i.e., the optimal solution of Eq. 3.19 is zero) since the translation vector of each component part is always perpendicular to the contact normals. To identify this special case, we compute the rank of first $N-2$ columns of $\boldsymbol{B}_{\text{in}}$ after solving the linear program. The equation $\boldsymbol{B}_{\text{in}}\hat{\boldsymbol{v}} = \boldsymbol{0}$, where $\hat{\boldsymbol{v}}_N = \hat{\boldsymbol{v}}_{N-1} = 0$ has non-zero solutions if and only if rank of the first $N-2$ columns of $\boldsymbol{B}_{\text{in}}$ is smaller than $N-2$ according to the theorem of homogeneous linear equations.



3.5.2 DBG-based Interlocking Test

The kinematic-based interlocking test, though can handle a variety of inputs, is less intuitive since it requires solving a linear inequality system. For those assemblies with only planar or piecewise planar contacts, we assume that the assembly $\boldsymbol{P}$ can be disassembled by single-part translational motions, i.e., part rotation is not required and all other parts remain fixed when removing a part. We propose a graph-based interlocking test which relates the globally interlocking with strong connectivity property in graph theory. Due to its simplicity, the graph-based test is used for designing globally interlocking assemblies discussed in Chapter4.

Directional Blocking Graph (DBG). We denote as $G(\boldsymbol{d},\boldsymbol{P})$ the *directional blocking graph* of assembly $\boldsymbol{P}$ for translation along direction $\boldsymbol{d}$. This directed graph has nodes representing the parts of $\boldsymbol{P}$ and directed edges $e_{i\to j}$ from P_i to P_j if and only if P_j prevents any translational motion of P_i along $\boldsymbol{d}$. In other words, $e_{i\to j}$ can be read as "P_i is blocked by P_j" in direction d.See Figure 3.6(c&d) for two examples.

The $G(\boldsymbol{d},\boldsymbol{P})$ can be derived from our motion-based representation (Section 3.4). Supposed that $V_{i,j}$ is the motion cone between P_i and P_j, if $\boldsymbol{d} \notin -V_{i,j}$, there must be an edge from P_i to P_j (i.e., $e_{i\to j}$) in the directional blocking graph $G(\boldsymbol{d},\boldsymbol{P})$. The directional blocking graph is just a projection of the motion based representation onto the direction of $\boldsymbol{d}$.

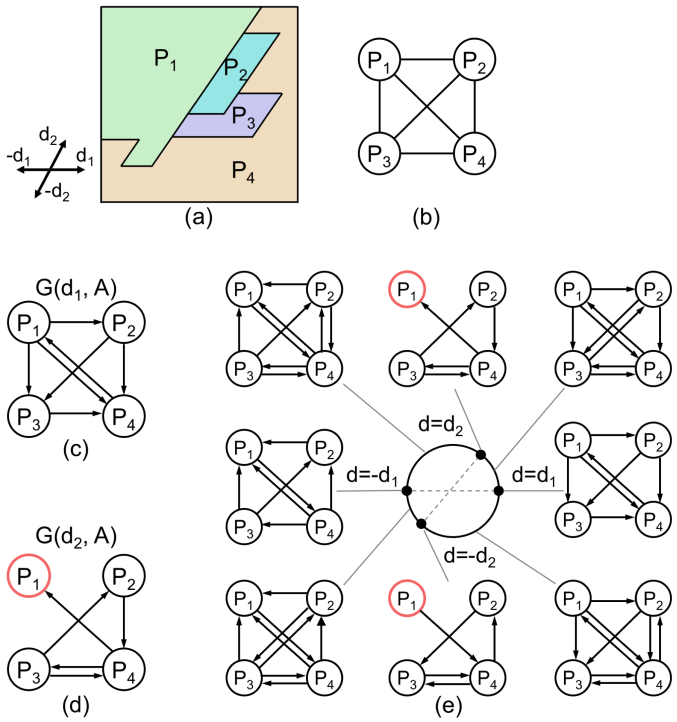
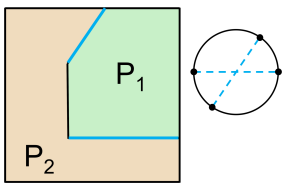


Figure 3.6 – Example DBGs and NDBG. (a&b) A 2D interlocking assembly and its parts-graph, where the key P_1 is movable along d_2; (c&d) Two DBGs of the assembly; and (e) NDBG of the assembly. A part with zero out-degree or in-degree in a DBG is highlighted with a red circle.

If $G(\boldsymbol{d}, \boldsymbol{P})$ is *strongly connected*, i.e. if every node can be reached from every other node, no part or part group is movable along d; see Figure 3.6(c). A part group $\boldsymbol{S}$ of $\boldsymbol{P}$ is locally free to translate in direction $\boldsymbol{d}$ ($-\boldsymbol{d}$), if and only if the out-degree (in-degree) of $\boldsymbol{P}$ in $G(\boldsymbol{d}, \boldsymbol{P})$ is zero; see P_1 in Figure 3.6(d).

Non-directional Blocking Graph (NDBG). We represent the set of all translation directions in 2D by the unit circle denoted as C. For every pair of parts in contact in $\boldsymbol{P}$, we draw the diameter that is parallel with the contact line. The drawn diameters partition C into an arrangement of regions, for which the corresponding DBG $G(\boldsymbol{d}, \boldsymbol{P})$ remains constant when $\boldsymbol{d}$ varies over a region. For any pair of parts in contact (e.g., P_1 and P_2 in the inset), if there are more than two contact lines, we only retain the two diameters of C (e.g., two contact lines in blue) which bound the cone of directions in which one part is free to translate relative to the other. The arrangement of points and intervals on C, and the associated DBGs form the *non-directional blocking graph* of $\boldsymbol{P}$; see Figure 3.6(e). An NDBG of a 3D assembly can be built similarly by constructing DBGs for each point and regular region on a unit sphere that represents all possible translation directions in 3D; please refer to [Wilson and Latombe, 1994] for more details.



Base Directional Blocking Graphs. An NDBG represents the parts blocking relations with

redundancy in two aspects. First, the DBG corresponding to an arc in C can be derived by performing union operations on the DBGs associated with the two end points of the arc; see again Figure 3.6(e). Second, we can obtain $G(-\boldsymbol{d}, \boldsymbol{P})$ from $G(\boldsymbol{d}, \boldsymbol{P})$ easily by reversing the direction of every edge in $G(\boldsymbol{d}, \boldsymbol{P})$ due to the reciprocity of blocking relations among the parts.

Therefore, it is sufficient to model the blocking relations in $\boldsymbol{P}$ by using only a set of *base DBGs* denoted as $\{G(\boldsymbol{d}, \boldsymbol{P})\}$, which we select as the DBGs corresponding to the end points in a half circle of C. For example, two DBGs in Figure 3.6(c&d) form $\{G(\boldsymbol{d}, \boldsymbol{P})\}$. We call the set of directions corresponding to the base DBGs as *base directions*, denoted as $\{\boldsymbol{d}\}$. The number of base DBGs (as well as base directions) is $O(N^2)$ since every pair of parts provides at most two diameters in C.

DBG-based Testing Approach. To test immobilization of a part group $\boldsymbol{S}$, we need to compute blocking relations between $\boldsymbol{S}$ and $\boldsymbol{P}-\boldsymbol{S}$: the part group $\boldsymbol{S}$ is immobilized if $\boldsymbol{S}$ is blocked by $\boldsymbol{P}-\boldsymbol{S}$ in all translation directions. Explicitly testing interlocking by checking immobilization of every part and every part group has exponential time complexity. However, treating each part group $\boldsymbol{S}$ independently ignores significant redundancies in the blocking relations across the parts. We exploit these redundancies and propose a more efficient approach to test interlocking. The key idea is to utilize the blocking relations encoded in the set of base DBGs to implicitly test immobilization of every part and every part group along a finite number of translation directions, i.e., the base directions $\{\boldsymbol{d}\}$.

In detail, an assembly with at least three parts is interlocking, if all base DGBs are either

1. strongly connected, or

2. have only two strongly connected components one of which has a single part that is identical across all DGBs.

Here the strongly connected component with a single part is the key of the assembly. Direction $\boldsymbol{d}$ associated with each DBG with two strongly connected components is the key's (reversed) movable direction according to the in-edge (out-edge) of the key in the DBG; e.g., the assembly in Figure 3.6(a) is interlocking since its two base DBGs in Figure 3.6(c&d) satisfy the above requirement.

In our implementation, we use Tarjan's algorithm [Tarjan, 1972] to find strongly connected components in each DBG. Runtime complexity is linear in the number of edges and nodes in the graph, i.e., $O(N^2)$ since there are at most N^2 edges in the graph. As the set of base DBGs has $O(N^2)$ graphs, the worst-case complexity of our interlocking testing algorithm is $O(N^4)$, which is much lower than $O(2^N)$ of the previous approach [Song et al., 2012]. In particular, the complexity to test interlocking of a well-structured assembly, where each part connects with at most $L \ll N$ parts, is $O(L^2N^2)$ since the number of base DBGs is $O(LN)$ and running

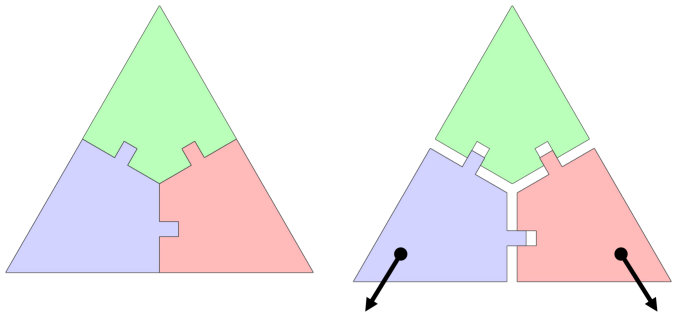


Figure 3.7 – A 2D assembly for which translating two parts along different directions (black arrows) simultaneously is the only way to disassemble it.

Tarjan's algorithm on each DBG is also $O(LN)$.

Remark. This DBG-based approach is sufficient for testing interlocking of 3D assemblies where parts are orthogonally connected; see supplementary material (Appendix A.3) for a proof. However, it is only necessary but not sufficient for testing interlocking of 3D assemblies with non-orthogonal part connections. Figure 3.7 shows a counter example. The DBG-based approach identifies this assembly as deadlocking yet two parts actually can move along different directions simultaneously. To address this issue, we suggest to use inequality-based approach described in Section 3.5.1.

3.5.3 Connection between Interlocking and Equilibrium

Interlocking and equilibrium describe two specific structural states of 3D assemblies. We make a formal connection between interlocking and equilibrium as follows:

> An *interlocking* assembly is an assembly that is in *equilibrium* under arbitrary external forces and torques.

This connection relies on the fact that the coefficient matrix $\boldsymbol{B}_{\text{in}}$ in Equation 3.9 and $\boldsymbol{A}_{\text{eq}}$ in Equation 3.3 are transposed to each other, according to the well-known close relation between velocity kinematics and statics [Davidson and Hunt, 2004].

The above statement can be formally proved based on a solvability theorem for a finite system of linear inequalities, in particular Farkas' lemma [Farkas, 1902]:

Lemma 2 (Farkas' Lemma) *Let $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ and $\boldsymbol{b} \in \mathbb{R}^n$. Then the following two statements are equivalent:*

(1) There exists an $\boldsymbol{x} \in \mathbb{R}^m$ such that $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ and $\boldsymbol{x} \geq \boldsymbol{0}$.

(2) There does not exist a $\boldsymbol{y} \in \mathbb{R}^n$ such that $\boldsymbol{A}^T\boldsymbol{y} \geq \boldsymbol{0}$ and $\boldsymbol{b}^T\boldsymbol{y} < \boldsymbol{0}$.

Our observation is that the mathematical formulations of equilibrium and interlocking in Sub-

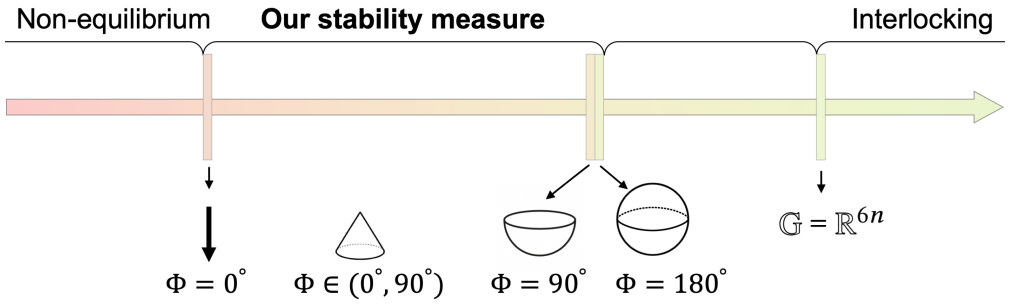


Figure 3.8 – Spectrum of assembly stability in which the stability increases from left to right, i.e., *non-equilibrium* (under single load, e.g., gravity), *equilibrium but not interlocking* that can be quantified by our stability measure Φ, and *global interlocking*. The gap between our stability measure and interlocking in the spectrum represents stability conditions where an assembly is in equilibrium under all possible gravity directions but not an arbitrary $\boldsymbol{w}$.

section 3.3 and 3.9 correspond to the first and second statement in Farkas' lemma, respectively. In particular, $\boldsymbol{A} = \boldsymbol{A}_{\text{eq}}$, $\boldsymbol{x} = \boldsymbol{f}$, and $\boldsymbol{b} = -\boldsymbol{w}$ relate statement 1 to Equation 3.3 while $\boldsymbol{A}^T = \boldsymbol{B}_{\text{in}}$ and $\boldsymbol{y} = \hat{\boldsymbol{v}}$ relate statement 2 to Equation 3.9.

By assuming that $\boldsymbol{b} = -\boldsymbol{w}$ can be an arbitrary vector (i.e., arbitrary external forces and torques), we can see that the condition of $\boldsymbol{b}^T\boldsymbol{y} < 0$ in statement 2 is equivalent to $\boldsymbol{y} \neq \boldsymbol{0}$. Statement 2 then becomes exactly consistent with the formulation of interlocking in Equation 3.18 and the formal connection between interlocking and equilibrium is proved.

3.6 Lateral Stability Measure

Our analysis shows that static equilibrium means that the assembly is stable under a constant external force and torque configuration $\boldsymbol{w}$, while global interlocking indicates that the structure is stable under an arbitrary $\boldsymbol{w}$. Equilibrium and interlocking are two binary states of structural stable 3D assemblies. In practice, ensuring static equilibrium for a single $\boldsymbol{w}$ might be insufficient since the assembly could be exposed to different forces (e.g., live loads). On the other hand, a global interlocking requirement might impose too strict constraints on the assembly's geometry, as real assemblies usually do not have to experience arbitrary external forces.

This motivates us to consider stability conditions that are more strict than single-load equilibrium; see Figure 3.8. Our idea for quantifying these stability conditions is based on the set of external force and torque configurations $\boldsymbol{w} \in \mathbb{R}^{6N}$ under which the assembly $\boldsymbol{P}$ is in equilibrium, denoted as the feasible set $\mathbb{G}(\boldsymbol{P})$, which has the following properties:

1. If $\boldsymbol{w} \in \mathbb{G}$, then $\lambda\boldsymbol{w} \in \mathbb{G}$ $(\lambda \geq 0)$, since we can multiply both sides of Equation 3.3 with λ.

2. If $\boldsymbol{w}_1 \in \mathbb{G}$ and $\boldsymbol{w}_2 \in \mathbb{G}$, then $\lambda\boldsymbol{w}_1 + (1-\lambda)\boldsymbol{w}_2 \in \mathbb{G}$ $(\lambda \geq 0)$ due to the linearity of Equation 3.3.

Hence, $\mathbb{G}(\boldsymbol{P})$ forms a convex cone in $\mathbb{R}^{6N}$. The case where $\mathbb{G}(\boldsymbol{P}) = \mathbb{R}^{6N}$ indicates that the assembly is global interlocking.

Similar to [Whiting et al., 2009], we consider a specific class of external force and torque configurations for the analysis and design of TI assemblies, in which each part P_i experiences a force $\boldsymbol{g}_i$ that passes through P_i's center of mass ($\boldsymbol{\tau}_i = \boldsymbol{R}_i \times \boldsymbol{g}_i$) and has a constant size (i.e., $\|\boldsymbol{g}_i\|$ equals to P_i's weight). Moreover, we assume that all $\boldsymbol{g}_i$ have the same direction, denoted by the unit vector $\boldsymbol{d}$. This assumption is motivated by the tilt analysis for measuring lateral stability of masonry structures in architecture [Zessin, 2012, Ochsendorf, 2002]; see Figure 3.8 By this, we reduce the degrees of freedom of $\boldsymbol{w}$ from $6N$ to 2 (i.e., a normalized vector $\boldsymbol{d}$).

We represent each normalized force direction $\boldsymbol{d}$ in spherical coordinates as $\boldsymbol{d}(\theta,\phi)$, where $\theta \in [0°, 360°)$ is the azimuthal angle and $\phi \in [0°, 180°]$ is the polar angle (relative to $-z$, the gravity direction). To compute $\mathbb{G}(\boldsymbol{P})$ we need to find all $\boldsymbol{d}(\theta,\phi) \in \mathbb{G}(\boldsymbol{P})$. Here, we check if $\boldsymbol{d}(\theta,\phi) \in \mathbb{G}(\boldsymbol{P})$ by testing whether the assembly $\boldsymbol{P}$ is in equilibrium under external forces with direction $\boldsymbol{d}(\theta,\phi)$ by solving Equation 3.3. Assuming that an assembly $\boldsymbol{P}$ is in equilibrium under gravity (i.e., $\boldsymbol{d}_g = (0,0,-1) \in \mathbb{G}(\boldsymbol{P})$), we approximate $\mathbb{G}(\boldsymbol{P})$ by uniformly sampling θ and finding the critical ϕ for each sampled θ using binary search, thanks to the convexity of $\mathbb{G}$. Figure 3.9 shows an example feasible cone $\mathbb{G}(\boldsymbol{P})$ computed using our approach, as well as its cross section with plane $z = -1$, called the feasible section $\mathbb{S}(\boldsymbol{P})$.

Given the feasible cone $\mathbb{G}(\boldsymbol{P})$, we define our stability measure as:

$$\Phi(\boldsymbol{P}) \;=\; \min\{\, \phi \mid \boldsymbol{d}(\theta,\phi) \in \partial\mathbb{G}(\boldsymbol{P}) \,\} \tag{3.20}$$

where $\partial\mathbb{G}(\boldsymbol{P})$ denotes boundary of the feasible cone $\mathbb{G}(\boldsymbol{P})$. Our measure is actually the minimum critical tilt angle among all possible azimuthal tilt axes, which can be considered as a generalization of the critical tilt angle for a fixed axis [Zessin, 2012]; see Figure 3.9-b&c. Figure 3.8 shows how our stability measure is embedded in the whole stability spectrum, where $\Phi = 0°, 90°, 180°$ highlight some special stability states. Specifically, the stability states corresponding to $\Phi = 90°$ and $\Phi = 180°$ are adjacent in the spectrum since the feasible cone $\mathbb{G}(\boldsymbol{P})$ cannot be in-between a half sphere and a whole sphere due to the property of convexity.

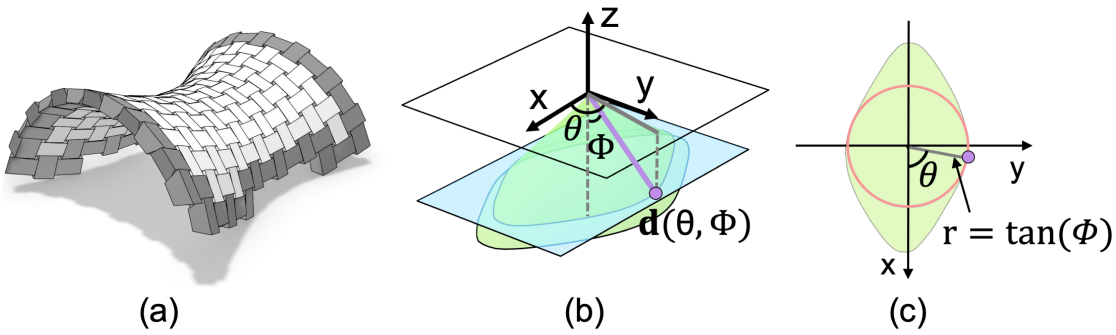


Figure 3.9 – (a) A assembly $\boldsymbol{P}$ and (b) its feasible cone $\mathbb{G}(\boldsymbol{P})$. (c) We visualize $\mathbb{G}(\boldsymbol{P})$ as the feasible section $\mathbb{S}(\boldsymbol{P})$ by intersecting it with the cyan plane ($z = -1$) in (b). The external force direction corresponding to our stability measure Φ is shown as a purple vector in (b) and a purple dot in (c). Note that the purple dot is the point of tangency between the feasible section $\mathbb{S}(\boldsymbol{P})$ and its largest inner circle (in red) centered at the origin.

4 Computational Design of Interlocking Assemblies

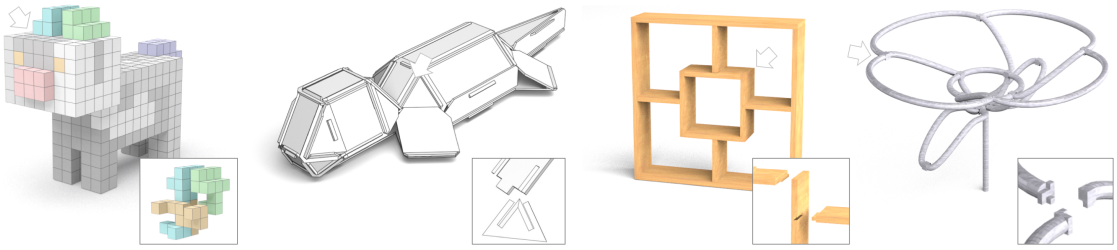


Figure 4.1 – Various interlocking assemblies designed using our framework, from left to right: voxelized puzzle, plate structure, furniture, and frame structure. Our method supports different types of joints as highlighted in the zooms. Please refer to the accompanying video for assembly sequences and the supplementary material for the blocking graphs defining the interlocking configurations.

Interlocking assemblies have a long history in the design of puzzles, furniture, architecture, and other complex geometric structures. The key defining property of interlocking assemblies is that all component parts are immobilized by their geometric arrangement, preventing the assembly from falling apart. Computer graphics research has recently contributed design tools that allow creating new interlocking assemblies. However, these tools focus on specific kinds of assemblies and explore only a limited space of interlocking configurations, which restricts their applicability for design.

In this chapter, we propose a new general framework for designing interlocking assemblies. The core idea is to represent part relationships with a family of base *Directional Blocking Graphs* and leverage efficient graph analysis tools to compute an interlocking arrangement of parts. This avoids the exponential complexity of brute-force search. Our algorithm iteratively constructs the geometry of assembly components, taking advantage of all existing blocking relations for constructing successive parts. As a result, our approach supports a wider range of assembly forms compared to previous methods and provides significantly more design flexibility.

We show that our framework facilitates efficient design of complex interlocking assemblies, including new solutions that cannot be achieved by state of the art approaches.

4.1 Introduction

3D assemblies refer to objects that combine multiple component parts into a structure with a specific form and/or functionality. Connection mechanisms are usually required to prevent the parts from moving relative to one another and make the assembly steady for practical use. However, these connectors can be irreversible (e.g., glue), impair the structural integrity of parts (e.g., nails), or degrade the external appearance of the assembly (e.g., clamps).

Rather than relying on additional explicit connectors, interlocking assemblies connect parts into a steady structure based only on the geometric arrangement of the parts. This intriguing property facilitates repeated assembly and disassembly and significantly simplifies the correct alignment of parts during construction. Consequently, interlocking assemblies have been used in a variety of applications, including puzzles [Stegmann, 2018], furniture [Fu et al., 2015], architecture [Deepak, 2012], and 3D printing [Yao et al., 2017b].

In an interlocking assembly, parts need to follow certain orders to be assembled into the target object. Once assembled, there is only one movable part, called the *key*, while all other parts as well as any subset of parts are immobilized relative to one another [Song et al., 2012]. However, this defining property of parts immobilization makes designing interlocking assemblies highly challenging. Explicitly testing the immobilization of every subset of parts requires costly computations; optimizing for the geometry of parts that satisfy these immobilization requirements, while avoiding dead-locking, is even more complex.

Recently, several computational approaches have been developed to address this problem [Xin et al., 2011, Song et al., 2012, Fu et al., 2015, Song et al., 2016, Zhang and Balkcom, 2016, Song et al., 2017, Yao et al., 2017b]. The common idea is to directly guarantee global interlocking by constructing and connecting multiple local interlocking groups (LIGs), which avoids the overhead of testing all part subsets for immobilization. While these methods show successful results, they only focus on specific sub-classes of interlocking assemblies, e.g., recursive interlocking puzzles [Song et al., 2012], but do not explore the full search space of all possible interlocking configurations. As a consequence, these approaches are restricted in the kind of input shapes they can handle and have limited flexibility to satisfy additional design requirements besides interlocking, e.g., related to aesthetics or functional performance.

Contributions In this chapter, we propose a new general framework for DESigning Interlocking Assemblies, called *DESIA*, that avoids the restrictions of previous LIG-based methods. Specifically, we make the following contributions:

1. We represent interlocking assemblies with a set of base *Directional Blocking Graphs*

(DBGs) and implement an efficient graph analysis algorithm that can test for global interlocking in polynomial time complexity.

2. We introduce a general iterative framework for designing interlocking assemblies that can explore the full search space of all possible interlocking configurations by utilizing all existing part blocking relations described in the graphs.

3. We demonstrate the flexibility of our framework for designing different classes of assemblies, including new types of interlocking forms that have not been explored in previous works.

The rest of the chapter is organized as follows. Section 4.2 describes our computational framework for designing interlocking assemblies. In Section 4.3 we show different types of assemblies generated with our approach, compare with previous works, and highlight several application examples. We conclude with a discussion of limitations of our approach and some thoughts on future research problems.

4.2 Computational Design Framework

In Section 3.5.2, we have demonstrated our efficient algorithms to test for interlocking. Our main goal now is to provide effective algorithms for designing interlocking assemblies. We first provide a high-level overview of our framework before presenting the conceptual and algorithmic details.

As input we expect the final shape of the assembly, from which the component parts are either constructed from scratch as in [Xin et al., 2011, Song et al., 2012, Song et al., 2015] or explicitly initialized as in [Fu et al., 2015, Song et al., 2016, Yao et al., 2017b]. Our computational process for creating an interlocking assembly starts with the full input model, then iteratively splits off successive parts for disassembly. At each iteration, we first identify a set of suitable blocking relations to be generated between the current assembly and the new part such that the interlocking property is maintained. Then we search for the part geometry that satisfies these blocking relations. The selection of a new part is guided by a ranking function that takes into account certain geometric properties, e.g. part size, or other requirements, e.g. on part fabrication. The search space is then explored in a tree traversal process that uses automatic backtracking when no interlocking solution could be found in a specific iteration; see Figure 4.2. We also provide a user interface to interactively explore different options for part decomposition, allowing the user to overwrite the generic ranking function for part selection; see Figure 4.12.

4.2.1 Iterative Design Framework

Given the input shape denoted as R_0, we iteratively construct the geometry of each part (or introduce appropriate joints in the geometry of each initialized part; see Section 4.3.2), one by

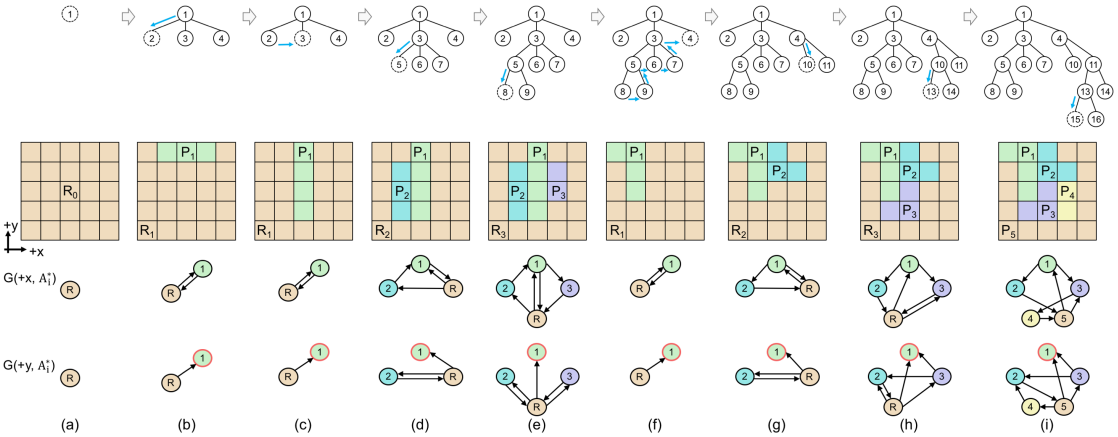


Figure 4.2 – Overview of our framework on designing a 2D interlocking assembly. (a) Given a 5×5 square as input, (b-i) our framework tries to generate a 5-part interlocking 2D assembly, where each part should have at least two pixels. Top row: construction tree, where each node at depth i represents a candidate of $\mathbf{A}_i$. Here we show the $m = 3$ highest ranked options at each level with the top-ranked on the left. The blue arrows indicate the procedure to visit the nodes for generating parts. If the framework cannot find any child for the current node (in dashed circle, denoted as $\mathbf{A}_i^*$), it will backtrack to (c) its siblings or (f) ancestors. Middle row: geometric examples corresponding to the dashed node in the tree. Bottom row: base DBGs of the geometric examples, where the key is indicated by a red circle. For simplicity, we show nodes P_j ($j \leq i$) as j, and R_i as R in (a-h), and show the last part R_4 as P_5 in the final assembly(i).

one. This forms a sequence of constructed parts, P_1, P_2, ..., P_n, with R_n, the remaining part of R_0, as the last part:

$$[R_0] \rightarrow [P_1, R_1] \rightarrow [P_1, P_2, R_2] \rightarrow ... \rightarrow [P_1, ...P_n, R_n] \,.$$

Here we denote each intermediate assembly $[P_1, ..., P_i, R_i]$ as $\mathbf{A}_i$ ($0 \leq i \leq n$), and its base DBGs as $\{G(d, A_i)\}$. Figure 4.2 shows an example where the parts are constructed from scratch.

To guarantee that the resulting assembly $\mathbf{A}_n = [P_1, \cdots, P_n, R_n]$ is interlocking and disassemblable, we have the following requirements when decomposing R_{i-1} into P_i and R_i:

(i) *Connected.* The geometries of P_i and R_i should each be connected, making $\mathbf{A}_i$ a valid assembly.

(ii) *Interlocking.* $\mathbf{A}_i$ ($i \geq 2$) is interlocking with P_1 as the key. In other words, $\{G(d, A_i)\}$ should satisfy the interlocking requirement described in Section 3.5.2.

(iii) *Disassemblable.* P_i can be removed from $[P_i, R_i]$, so we can disassemble $\mathbf{A}_i$ in the order of P_1, P_2, ..., P_i, R_i.

The advantage of this iterative design framework is that we achieve the goal of global in-

Algorithm 1 Algorithm to design an interlocking assembly $\mathbf{A}_n$ from a given shape R_0.

function CREATEINTERLOCKASSEMBLY(R_0)
  $i \leftarrow 0$
  $A_i^* \leftarrow [R_0]$
  while $i < n$ **do**
    if i=0 **then**
      $\{A_{i+1}\} \leftarrow$ GenerateKey(A_i^*) ▷ See Subsection 4.2.2
      if $\{A_{i+1}\} = \emptyset$ **then**
        return NULL
      end if
    else
      $\{A_{i+1}\} \leftarrow$ GenerateParts(A_i^*) ▷ See Subsection 4.2.3
    end if
    if $\{A_{i+1}\} \neq \emptyset$ **then**
      RankCandidates($\{A_{i+1}\}$) ▷ In descending order
      if $i+1 = n$ **then**
        return A_n^1
      else
        $i \leftarrow i+1$
        $A_i^* \leftarrow A_i^1$
      end if
    else if $A_i^*.sibling \neq NULL$ **then**
      $A_i^* \leftarrow A_i^*.sibling$ ▷ $A_i^*.sibling$ is the one second to A_i^* in the ranked $\{A_i\}$
    else
      while $A_i^*.parent \neq$ NULL &&
          $A_i^*.parent.sibling =$ NULL **do**
        $A_i^* \leftarrow A_i^*.parent$
        $i \leftarrow i-1$
        if $i = 0$ **then** ▷ Quit if backtrack to A_0
          return NULL
        end if
      end while
      if $A_i^*.parent \neq$ NULL &&
        $A_i^*.parent.sibling \neq$ NULL **then**
        $A_i^* \leftarrow A^*.parent.sibling$
      else
        return NULL
      end if
    end if
  end while
end function

terlocking by satisfying a set of local requirements when constructing each pair of P_i and

R_i.

Tree Traversal. Since we cannot guarantee that the construction of P_i and R_i succeeds at every iteration, we propose an iterative approach with backtracking to construct $\mathbf{A}_n$; see Figure 4.2 and Algorithm 1. The key idea is to build and maintain a construction tree, where each node represents a candidate of $\mathbf{A}_i$. For each node, we generate a set of children denoted as $\{\mathbf{A}_{i+1}\}$, among which the only different parts are P_{i+1} and R_{i+1}. Our approach ranks these candidate assemblies at each iteration to facilitate the construction of successive parts. For example, we rank $\{\mathbf{A}_{i+1}\}$ according to the compactness of R_{i+1} measured by using the accessibility in [Song et al., 2012], since parts extracted from a compact R_{i+1} are more likely to be connected; compare R_1 in Figure 4.2(c&f). In case the user has other design goals besides interlocking, e.g., regarding the appearance of the assembly, we support user intervention to adjust the ranking; see again Figure 4.12. In case we cannot generate any valid result from the selected candidate in $\{\mathbf{A}_{i+1}\}$, we can backtrack the tree to try other nodes without restarting the whole design process. The size of $\{\mathbf{A}_{i+1}\}$ is denoted as m. A large m requires more time for generating $\{\mathbf{A}_{i+1}\}$, but also provides more choices for ranking and backtracking. We set $m = 30$ by default in our experiments but it can be adjusted, depending on the input model.

Below we explain our approach to generate the key part (Subsection 4.2.2) and the remaining parts of the assembly (Subsection 4.2.3). These steps can be customized to design different kinds of interlocking assemblies as discussed in Section 4.3. Here, we take 2D interlocking puzzle design as an example for illustration.

4.2.2 Generating the key

We first partition the input model R_0 into P_1 and R_1, where P_1 is the key and R_1 is the remaining part. We construct the geometry of P_1 following the procedure in [Song et al., 2012], i.e., select a seed pixel, ensure its blocking and mobility, and expand the key part. Recall that the key is

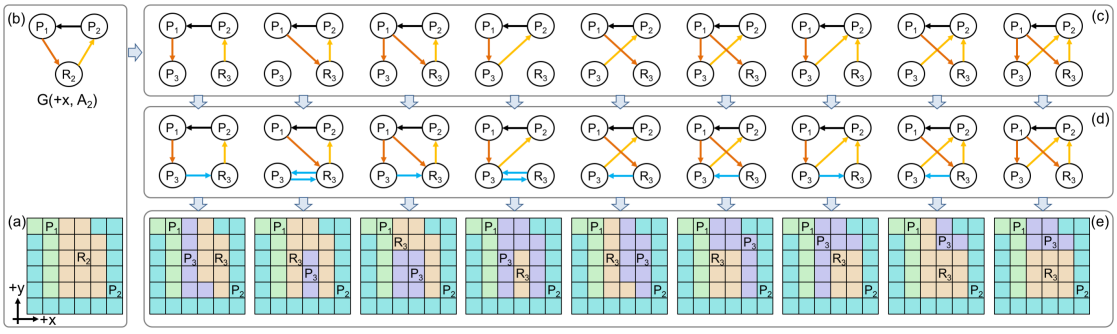


Figure 4.3 – (a) An intermediate assembly $\mathbf{A}_2$ and (b) its base DBG $G(+x, A_2)$, where the in-edge and out-edge of R_2 are colored in dark and light orange respectively; (c) all cases of distributing existing blocking relations (dark and light orange edges) to P_3 and R_3; (d) the interlocking graph designs that require the fewest internal blocking relations (blue edges) between P_3 and R_3; and (e) the corresponding geometric examples.

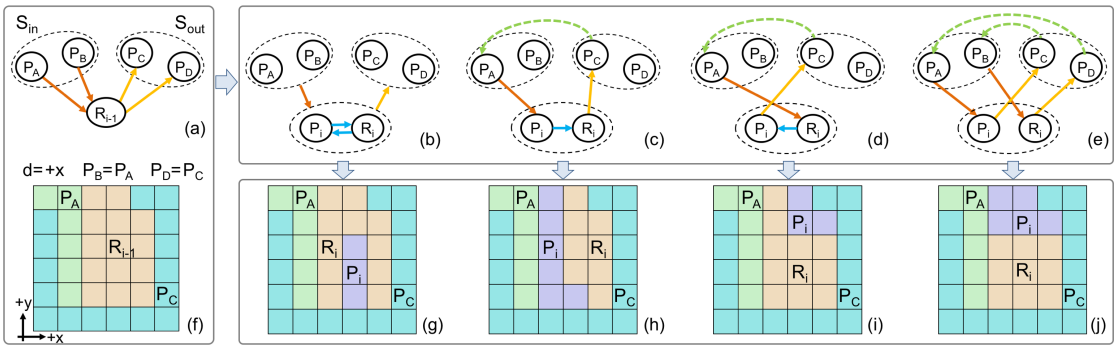


Figure 4.4 – (a) Given $G(d, A_{i-1})$, (b-e) we ensure that $G(d, A_i)$ is strongly connected by constructing a cycle that includes both P_i and R_i. A dashed ellipse in (a-e) indicates a subset of parts, i.e., S_{in}, S_{out}, and $\{P_i, R_i\}$, where S_{in} (S_{out}) denotes the set of parts with an edge to (from) R_{i-1}. The directed edges from (to) a dashed circle in (b) indicate that the edge can be from (to) any part in the associated subset. The dashed green edges in (c-e) indicate that a part can reach the other part in $G(d, A_{i-1})$ without passing through R_{i-1}. (f-i) Geometric examples corresponding to (a-e), where $d = +x$, $S_{in} = \{P_A\}$, and $S_{out} = \{P_C\}$.

the only movable (thus unstable) part in an interlocking assembly. Therefore, we restrict P_1 to have a single movable direction in $\mathbf{A}_1$ denoted as d_1, and usually select d_1 being upward to stabilize P_1 with gravity; see Figure 4.2(b&c) for two examples. We rank the candidates in $\{\mathbf{A}_1\}$ according to the compactness of R_1.

4.2.3 Generating P_i and R_i ($i > 1$)

Next, we construct P_i and R_i from R_{i-1} in two stages: *graph design* and *geometry realization*. The first stage constructs base DBGs $\{G(d, A_i)\}$ that satisfy the interlocking requirement conceptually. And the second stage aims at realizing the blocking relations described in $\{G(d, A_i)\}$ in the embedded geometry while satisfying the part connectivity and disassemblability requirements defined in Subsection 4.2.1. Note that geometric constraints (e.g., supported joint types) can be used to simplify graph design by eliminating potential graph edges that cannot be realized geometrically anyway.

Graph Design for P_i and R_i. Starting from $\mathbf{A}_{i-1}$ ($i \geq 2$), the goal is to find blocking relations for P_i and R_i such that the updated assembly $\mathbf{A}_i$ is still interlocking. In other words, after splitting R_{i-1} into P_i and R_i in $\{G(d, A_{i-1})\}$ to form $\{G(d, A_i)\}$, we need to construct a set of new edges for P_i and R_i in each $G(d, A_i)$ such that the graph remains strongly connected, except the key; see Figure 4.2. To achieve this goal, we first classify blocking relations to be constructed into two classes:

(i) *External blocking relations* between $\{P_1, .., P_{i-1}\}$ and P_i, as well as those between $\{P_1, .., P_{i-1}\}$ and R_i are inherited from those between $\{P_1, .., P_{i-1}\}$ and R_{i-1}. We need to distribute these existing blocking relations to P_i and R_i; see Figure 4.3(c).

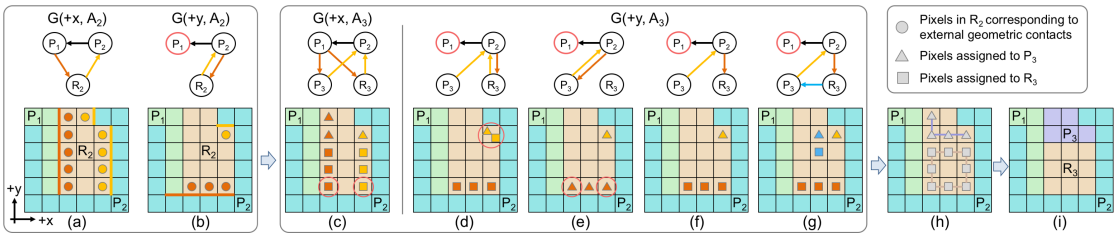


Figure 4.5 – Geometry realization of $\{G(d, A_3)\}$. (a&b) Identify geometric contacts between $\{P_1, P_2\}$ and R_2 in $\mathbf{A}_2$. (c-f) Distribute external geometric contacts between P_3 (shows as triangles) and R_3 (shown as squares) for (c) $G(+x, A_3)$ and (d-f) $G(+y, A_3)$, where (d&e) show two failure examples. (g) Realize internal blocking relation between P_3 and R_3 in $G(+y, A_3)$. (h) Construct initial geometry of P_3 and R_3. (i) Resulting $\mathbf{A}_3$.

(ii) *Internal blocking relations* between P_i and R_i. For each case of distributing external blocking relations, we may need to construct internal blocking relations between P_i and R_i such that each $G(d, A_i)$ remains strongly connected[1]; see Figure 4.3(d).

Given these observations, we could find all valid graph designs by enumerating the distribution of external blocking relations, constructing the corresponding internal blocking relations, and testing the strongly connected property of the DBGs. However, this generate-and-test approach could be very inefficient. The number of choices to distribute external blocking relations is 3^l, where l is the number of edges of R_{i-1} in $G(d, A_{i-1})$, since each edge of R_{i-1} can be distributed to P_i, R_i, or both. Figure 4.3 shows an example with $3^2 = 9$ graph designs, where $l = 2$.

Rather than enumerating all possible graph designs, we propose an efficient approach to find a desired number of designs that are interlocking conceptually; see Figure 4.4. The key idea is to directly guarantee that each $G(d, A_i)$ is strongly connected by constructing a cycle in the graph that includes both P_i and R_i, given that $G(d, A_{i-1})$ is already strongly connected. Denote S_{in} (S_{out}) as the set of parts with an edge to (from) R_{i-1}, and P_{in} (P_{out}) as an arbitrary part in S_{in} (S_{out}); see Figure 4.4(a&f). According to the number of internal blocking relations to be constructed between P_i and R_i denoted as K, we have the following three cases to construct the cycle that we can choose independently for each DBG.

(i) *K=2.* $P_i \rightarrow R_i \rightarrow P_i$ forms a cycle, i.e., any distribution of external blocking relations works for this case; see Figure 4.4(b).

(ii) *K=1.* $P_i \rightarrow R_i \rightarrow P_{out} \dashrightarrow P_{in} \rightarrow P_i$ ($R_i \rightarrow P_i \rightarrow P_{out} \dashrightarrow P_{in} \rightarrow R_i$) forms a cycle if the single directed edge is from P_i to R_i (from R_i to P_i); see Figure 4.4(c&d). Here, $P_{out} \dashrightarrow P_{in}$ means that P_{out} can reach P_{in} in $G(d, A_{i-1})$ without passing through R_{i-1}, or P_{out} and P_{in} are the same part.

[1] If the key is movable along d, $G(d, A_i)$ is strongly connected without considering the key. Otherwise, the whole graph of $G(d, A_i)$ should be strongly connected.

(iii) *K=0.* $P_i \rightarrow P_{out} \dashrightarrow P_{in} \rightarrow R_i \rightarrow P'_{out} \dashrightarrow P'_{in} \rightarrow P_i$ forms a cycle, where P_{in} and P'_{in} (as well as P_{out} and P'_{out}) are possible to be the same part; see Figure 4.4(e).

Compared with case 1, cases 2 and 3 rely more on external blocking relations than on internal blocking relations to immobilize P_i and R_i. As a consequence, these two cases impose fewer constraints on the subsequent geometry construction of P_i and R_i, resulting in a higher chance to be successfully realized in the embedded geometry; compare the geometric examples in Figure 4.4(g-j).

Besides interlocking, we also need to ensure that P_i is disassemblable in $[P_i, R_i]$. Thus, we require that there are fewer than two directed edges between P_i and R_i (i.e., case 2 and 3) in at least one base DBG. The output of this stage is a set of $\{G(d, A_i)\}$ that satisfy the interlocking requirement, denoted as $\mathbf{C}_i$.

Geometry Realization of P_i and R_i. In order to realize $\{G(d, A_i)\} \in \mathbf{C}_i$ in the embedded geometry, we perform the following steps, each corresponding to a counterpart of the graph design stage:

i) Identify external geometric contacts between $\{P_1, ..., P_{i-1}\}$ *and* R_{i-1}. Recall that a directed edge $e_{i\rightarrow j}$ from P_i to P_j in $G(d, A)$ means that P_j blocks the translation of P_i along d. This indicates that P_i contacts P_j along d, and P_j locates further than P_i along d; see again Figure 3.6. In an assembly $\mathbf{A}_{i-1}$, we identify such blocking contacts between P_l $(1 \leq l \leq i-1)$ and R_{i-1} for each base direction d by computing the overlap of the respective boundaries

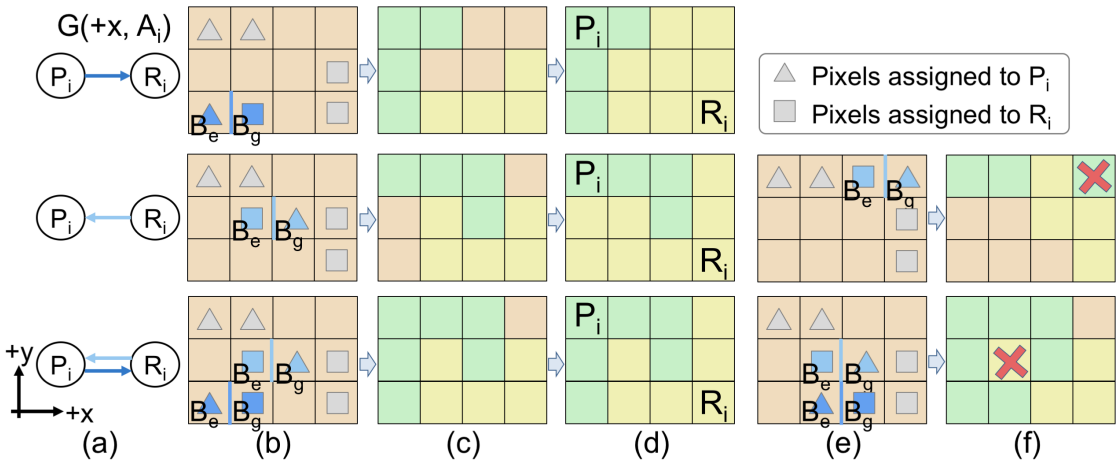


Figure 4.6 – (a) Internal blocking relations between P_i and R_i in $G(+x, A_i)$. (b) Find blocking and blockee pixels in R_{i-1} (in orange) according to the blocking relations, where blocking and blockee pixels in (b) and their associated blocking relation in (a) are colored the same (light or dark blue). (c) Initial geometry of P_i and R_i. (d) Final geometry of P_i and R_i. (e&f) Two failure examples due to disconnectivity of P_i or R_i (see the red cross).

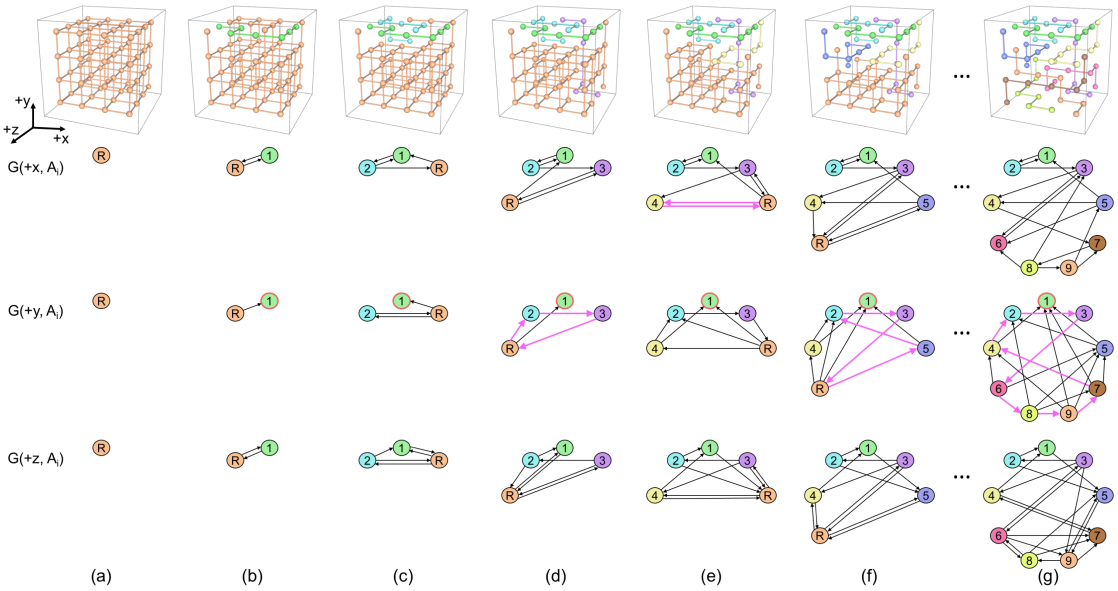


Figure 4.7 – (a) Starting from a 4×4×4 voxel grid, (b-g) our framework iteratively constructs 9 parts of an interlocking CUBE. Three DBGs are drawn for each intermediate assembly at the bottom. Our approach allows immobilizing P_i and R_i by constructing cycles of various sizes (examples colored in purple).

along d ($-d$), see Figure 4.5(a&b).

ii) Distribute external geometric contacts. An external blocking relation, say between P_l and R_{i-1}, in a DBG $G(d, A_{i-1})$ can be distributed to P_i, R_i, or both. For the first two cases, the corresponding geometric contacts need to be all assigned to P_i or R_i respectively; see Figure 4.5(f). For the last case, the geometric contacts need to be partitioned into two subsets and assigned to P_i and R_i separately; see Figure 4.5(c).

However, this step could fail for two reasons. First, the external geometric contact could be too small to be partitioned. For example, R_2 contacts P_2 along $+y$ with a single pixel in Figure 4.5 (b). Yet, this single pixel (marked with a red circle in Figure 4.5(d)) needs to be assigned to both P_3 and R_3 according to the computed blocking relations, which is not feasible. Second, the assignment of geometric contacts may conflict with one another across multiple DBGs. For example, two pixels marked with red circles in Figure 4.5(c) need to be assigned to R_3 to realize $G(+x, A_3)$. However, these two pixels also need to be assigned to P_3 to realize $G(+y, A_3)$ in Figure 4.5(e), leading to a conflict.

iii) Construct internal geometric contacts. If $G(d, A_i)$ has $K \in \{1, 2\}$ internal blocking relations, we need to construct geometric contacts between P_i and R_i. Here, we take as an example the case of realizing a single directed edge from P_i to R_i to illustrate our approach; see Figure 4.6(top). Inspired by [Song et al., 2012], we find among all unassigned pixels in R_{i-1} a pair of blocking and blockee pixels that contact each other along d, denoted as B_g and B_e

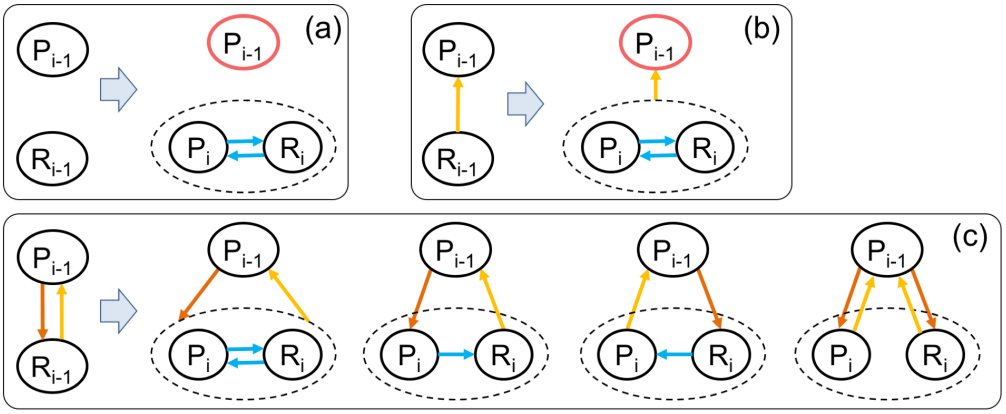


Figure 4.8 – Illustration of the model of [Song et al., 2012] based on our DBG-based representation. Their approach achieves global interlocking of A_n by requiring every $[P_{i-1}, P_i, R_i]$ $(2 \le i \le n)$ to form a local interlocking group with P_{i-1} as the key. In detail, P_{i-1} and R_{i-1} in $G(d, A_{i-1})$ are possible to have (a) zero, (b) one, and (c) two directed edges. (a&b) P_i and R_i are immobilized in a 2-part cycle $[P_i, R_i]$; (c) P_i and R_i are immobilized in either a 2-part cycle $[P_i, R_i]$, $[P_{i-1}, P_i]$, $[P_{i-1}, R_i]$, or a 3-part cycle $[P_{i-1}, P_i, R_i]$.

respectively. We then assign B_e to P_i and B_g to R_i. Other cases of realizing internal blocking relations can be handled similarly; see Figure 4.6.

iv) Construct initial parts geometry. By now, we have identified all the pixels in R_{i-1} that need to be assigned to P_i or R_i to make $\mathbf{A}_i$ interlocking. To form an initial P_i (R_i), we connect these pixels into a single part using the shortest path; see Figure 4.5(h) and 4.6(c). Note that this connection process can fail since we may not be able to find such a shortest path without disconnecting parts; see Figure 4.6(e&f) for examples.

v) Ensure disassemblability. To make P_i movable in $[P_i, R_i]$, we first identify all possible moving directions of P_i in $\{G(d, A_i)\}$, i.e., the directions where P_i is unblocked by R_i; e.g., P_3 could be movable along $\{-x, +x, +y\}$ in $[P_i, R_i]$ according to the blocking graph in Figure 4.5(c&g) . We try each possible moving direction of the initial P_i and discard those that cannot be achieved in the embedded geometry. We consider that P_i is disassemblable in $[P_i, R_i]$ if we can find one movable direction of P_i, along with a disassembly path. Lastly, we assign those remaining pixels in R_{i-1} (see orange pixels in Figure 4.6(c)) to P_i and R_i respectively according to geometric proximity with preference to R_i, while maintaining the disassemblability of P_i in $[P_i, R_i]$; see Figure 4.5(i) and 4.6(d).

A graph design is realized if all above steps succeed. Otherwise, we discard this design and try another one in $\mathbf{C}_i$. If all candidates in $\mathbf{C}_i$ fail, we backtrack to the other nodes in the construction tree following the procedure in Algorithm 1.

4.3 Results and Discussion

In this section we show how our framework can be used to design various kinds of interlocking assemblies, with example applications as puzzles, furniture, sculptures, or architectural designs. We highlight differences to previous approaches to show how our method improves the state of the art and enables new kinds of interlocking assemblies not possible before. For more detailed comparisons and results, we refer to the supplementary material (Appendix A).

4.3.1 Interlocking Voxelized Structures

Given a voxelized shape and a desired number of parts N as input, our goal here is to decompose the voxel set into a collection of parts that form an interlocking assembly [Song et al., 2012]. Figure 4.7 shows our iterative design process for creating a 9-part $4 \times 4 \times 4$ interlocking CUBE.

The major difference between our approach and [Song et al., 2012] is the graph design of P_i and R_i to ensure interlocking of $\mathbf{A}_i$. Our approach makes use of all previous parts $\{P_1, ..., P_{i-1}\}$ to immobilize P_i and R_i. (i.e., form a cycle in each $\{G(d, A_i)\}$; see Figure 4.4 and 4.7), while [Song et al., 2012] only relies on P_{i-1} to immobilize P_i and R_i (i.e., form a 2-part or a 3-part cycle in the DBGs; see Figure 4.8). Note that our approach can easily generate recursive interlocking puzzles as [Song et al., 2012] by constraining our graph design as shown in Figure 4.8.

Exploiting all existing blocking relations to immobilize P_i and R_i provides significantly more

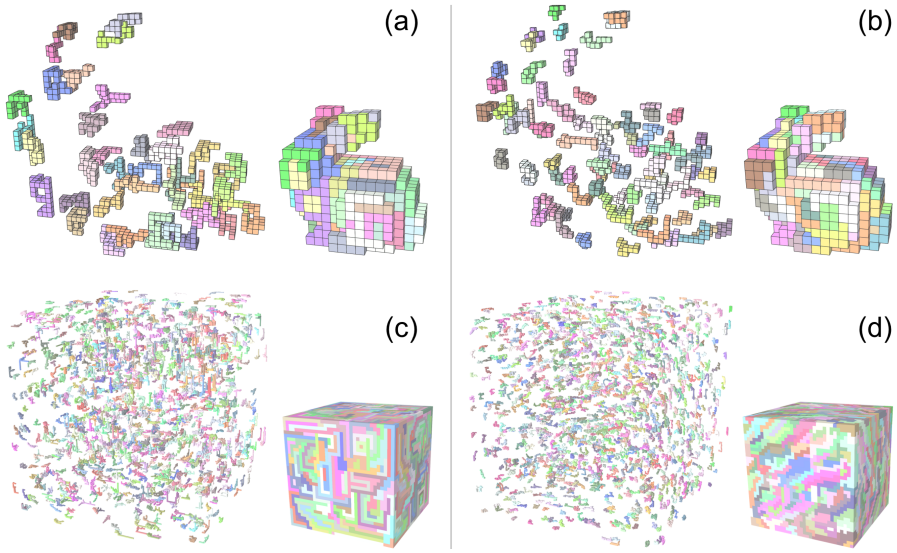


Figure 4.9 – (a&b) Interlocking Bunnies (966 voxels). (a) The method of [Song et al., 2012] can find an assembly with maximally 40 parts while (b) our approach can find one with 80 parts. (c&d) Interlocking $35 \times 35 \times 35$ Cubes. (c) The method of [Song et al., 2012] can find an assembly with maximally 1250 parts while (d) our approach can find one with 1500 parts.

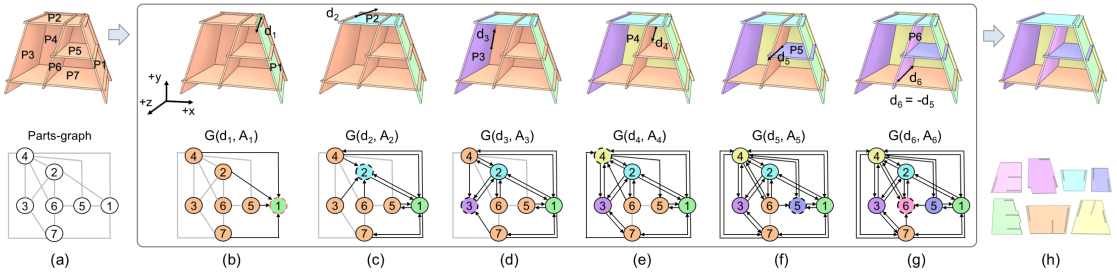


Figure 4.10 – Design of a 7-part interlocking CABINET by our approach. (a) Input design and parts-graph. (b-g) The iterative procedure to plan the joints, where the removal direction d_i of P_i is shown in the top row and the active DBG $G(d_i, A_i)$ is shown at the bottom row. All orange nodes in each DBG form R_i and the node with dashed boundary is P_i. (h) Interlocking result and the corresponding parts.

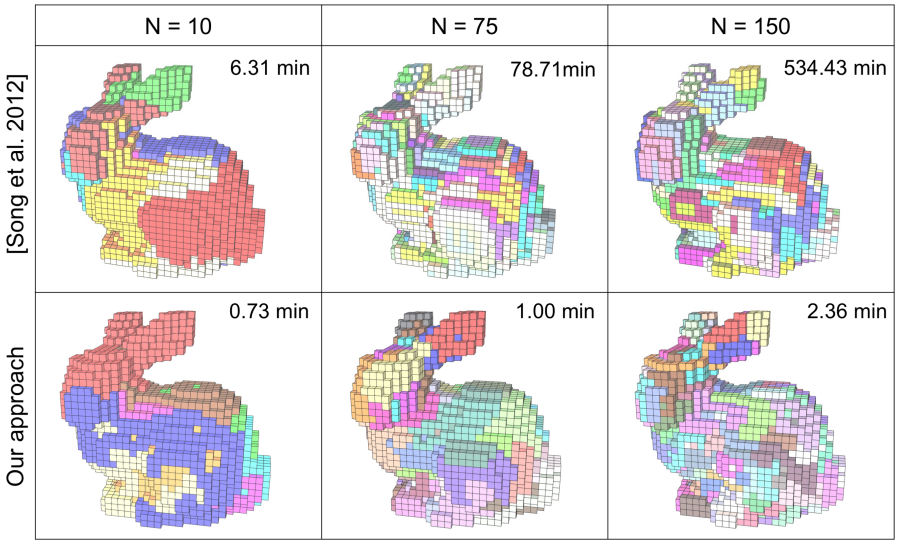


Figure 4.11 – Our approach is much faster than [Song et al., 2012] when generating 10-, 75-, and 150-part 30 × 29 × 23 BUNNIES.

degrees of freedom for designing interlocking assemblies. This allows us to incorporate additional design goals, for example, on object appearance as for the CARTOON DOG in Figure 4.1. Here we impose constraints that avoid cutting seams across geometric features, so that eyes, ears, nose, and tail are each assigned to a single assembly part. Second, as shown in Figure 4.9, our approach can find interlocking assemblies with more parts than [Song et al., 2012] for the same input. Given the same input model and the same N, our approach also takes a much shorter time than [Song et al., 2012] to generate results, even though we explore a much richer space of possible assemblies; see Figure 4.11. Lastly, our approach gives users more control for generating results by allowing them to select their desired criteria for ranking $\{\mathbf{A}_{i+1}\}$; see Figure 4.12 for an example.

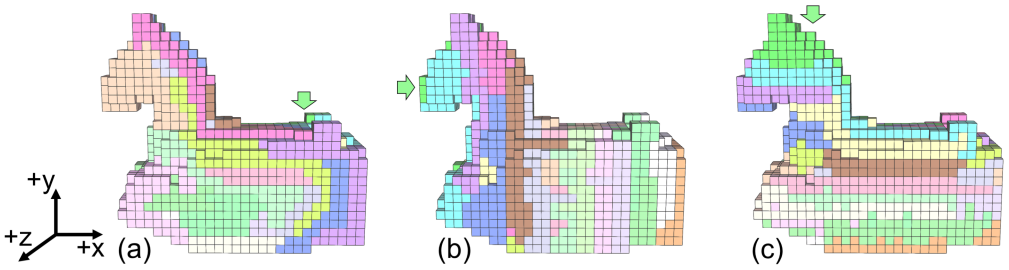


Figure 4.12 – Design a 20-part ISIDORE HORSE with different criteria for ranking $\{\mathbf{A}_{i+1}\}$: (a) compactness of R_{i+1} (default criteria); (b) X-coordinate of P_{i+1}; and (c) Y-coordinate of P_{i+1}. The ranking criteria could affect the assembly sequence; e.g., parts are disassembled from left to right for (b) and from top to bottom for (c), starting from the green key part (see the green arrow).

4.3.2 Interlocking Plate Structures

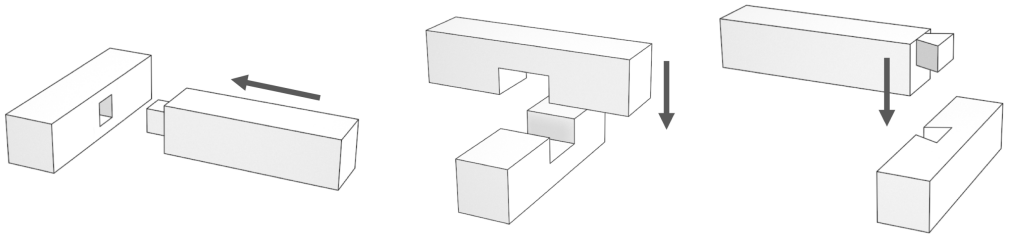
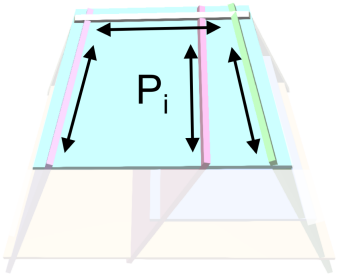


Figure 4.13 – Example woodworking joints. From left to right: mortise-and-tenon, halved joint, and dovetail joint, where the black arrow shows the single movable direction of the part allowed by the joint.

A second class of assemblies that we can create with our approach are interlocking plate structures that have applications in furniture design or architecture, for example. These assemblies differ in two main ways from the voxelized assemblies described above. First, the geometry of parts and their connections are predefined. We model part connections with an undirected *parts-graph*, in which nodes represent parts, and edges connect two contacting/intersecting parts; see Figure 4.10(a). Blocking relations can only be constructed between parts connected in the parts-graph. The dual of a *parts-graph* is a *joints-graph*, where nodes represent joints and edges represent parts; see Figure 4.17(a). Second, we use a set of predefined joint geometries to impose blocking relations between each pair of adjacent parts in the structure. Specifically, we consider mortise-and-tenon, halved, and dovetail joints that restrict each part to move along a single direction; see Figure 4.13. To support non-orthogonal part connections, we consider suitable variants of mortise-and-tenon and halved joints; see Figure 4.14.



In plate structures, the edge vectors shared between each part P_i and its adjacent parts indicate the base directions $\{d\}$ (see the arrows in the inset for examples), which degenerate into

six axial directions for structures where parts are orthogonally connected; see Figure 4.15(a).

To address the above specifics of interlocking plate structures, we have the following adaptations compared to the voxelized structures. First, instead of decomposing R_{i-1} into P_i and R_i, we iteratively select a single part P_i from the input, and consider the set of unselected parts as R_i; see Figure 4.10(b-g). Second, rather than constructing geometry of P_i and R_i, we construct joints between P_i and each part in R_i that are connected with P_i in the parts-graph denoted as $R_i^{'}$ such that $\mathbf{A}_i$ ($i \geq 2$) is interlocking. Since all our employed joints allow a single removal direction of the parts, the removal direction of P_i in $[P_i, R_i]$ denoted as d_i completely defines the joints to be constructed between P_i and each part in $R_i^{'}$. Third, to achieve interlocking, we only need to ensure that the *active base DBG* $G(d_i, A_i)$ is strongly connected; see DBGs in Figure 4.10(b-g). The other base DBGs should remain strongly connected since the newly introduced joints only allow part moving along d_i but not the other base directions.

Our iterative approach is detailed as follows:

1. *Iterative Design Framework.* Starting from the key P_1, we iteratively select a single part P_i from parts in R_{i-1}. We avoid selecting P_i that is a cut point in the remaining parts-graph of R_{i-1} to keep the geometry of R_i connected. Once P_i is selected, our task is to select d_i from the edge vectors $\{e_i\}$ shared between P_i and its adjacent parts.

2. *Generating the key.* Generally, we select P_1 as the part with the most parallel edge vectors, and use this direction as P_1's removal direction d_1 to facilitate joint construction on the key; see Figure 4.10(b). This is because we create a halved joint for the edge that is parallel to d_1, a mortise-tenon joint for the edge that is nearly perpendicular to d_1 (angle within $[45°, 135°]$), and an empty joint for the other edges. After selecting P_1 and d_1, we plan a joint between P_1 and each part in $R_1^{'}$ (e.g., $R_1^{'} = \{P_2, P_4, P_5, P_7\}$ in Figure 4.10(b)) such that P_1 is only movable along d_1.

3. *Generating P_i and R_i ($i > 1$).* At the graph design stage, we need to select d_i from $\{e_i\}$ such that $G(d_i, A_i)$ is strongly connected, which can be classified into two cases. The first case is that $\pm d_i \notin \{d_1, ..., d_{i-1}\}$. For this case, we build $G(d_i, A_i)$ by converting each undirected edge among $\{P_1, ..., P_{i-1}, R_{i-1}\}$ in the parts-graph into two directed edges and adding a single directed edge between P_i and each part in $R_i^{'}$. The $G(d_i, A_i)$ should

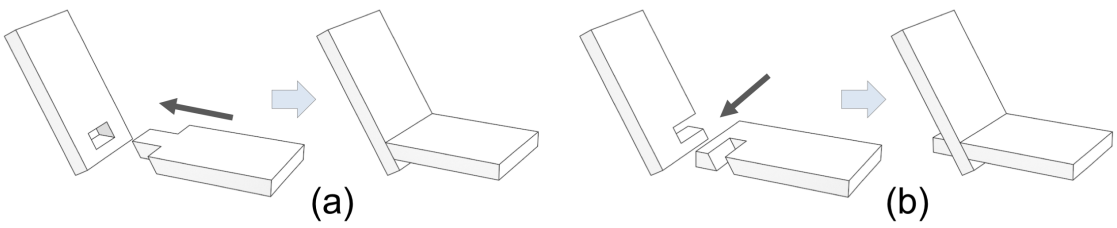


Figure 4.14 – Variants of (a) mortise-and-tenon joints and (b) halved joints that support non-orthogonal part connections with surface contact.

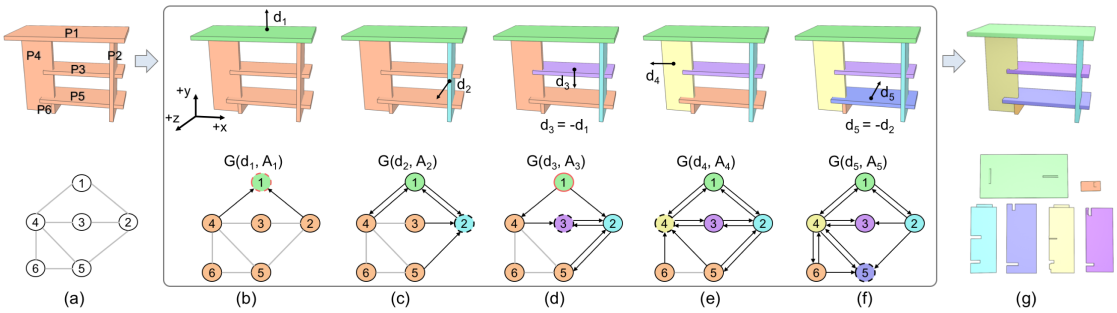


Figure 4.15 – Design of a 6-part interlocking TABLE with orthogonal joints. (a) Input design and parts-graph. (b-f) The iterative procedure to plan the joints. (g) Interlocking result and the corresponding parts.

be strongly connected by default; see Figure 4.10(c). The second case is that d_i or $-d_i$ $\in \{d_1, ..., d_{i-1}\}$, say $d_i = d_k$ $(1 \leq k \leq i-1)$. $G(d_i, A_i)$ inherits all blocking directions from $G(d_k, A_{i-1})$ and we add a single directed edge between P_i and each part in $R_i^{'}$. We try each of $\pm d_i$ and accept d_i $(-d_i)$ if $G(d_i, A_i)$ $(G(-d_i, A_i))$ is strongly connected; see Figure 4.10(g).

At the geometry realization stage, we use constructive solid geometry to create the joint geometry on P_i and each part in $R_i^{'}$ according to the joint type planned at the graph design stage. We rank the resulting candidates of $\mathbf{A}_i$ in ascending order of the number of empty joints in $\mathbf{A}_i$.

Figure 4.10(h) shows an interlocking CABINET designed by our approach. Besides furniture, plate structures also can be used to approximate a free form shape; see the 33-part LIZARD in Figure 4.1 for an example. Since the DBG-based approach is not sufficient to test interlocking for these plate structures with non-orthogonal part connections (see Subsection 3.5.1), we verify the two results by running the inequality-based interlocking test and find that both results pass the test.

In the special case of orthogonal joints, our approach generalizes the furniture design work of [Fu et al., 2015]; see Figure 4.15 for an example. In particular, Fu et al. [Fu et al., 2015] focus on furniture with 3- or 4-part cyclic substructures since their approach requires these substructures to construct LIGs. In contrast, our approach does not have such a limitation; see the BOOKSHELF with four 6-part cyclic substructures in Figure 4.1.

Lastly, inspired by our DBG-based representation, we find that a parts-graph with a cut point cannot be interlocking, no matter what kinds of joints are used; see Figure 4.16(left) for an example and supplementary material for a proof (Appendix A.2). This observation allows us to modify a given input to make it possible to be interlocking by adding a minimal number of new parts in the parts-graph in order to remove the cut point; see Figure 4.16(right) for an example.

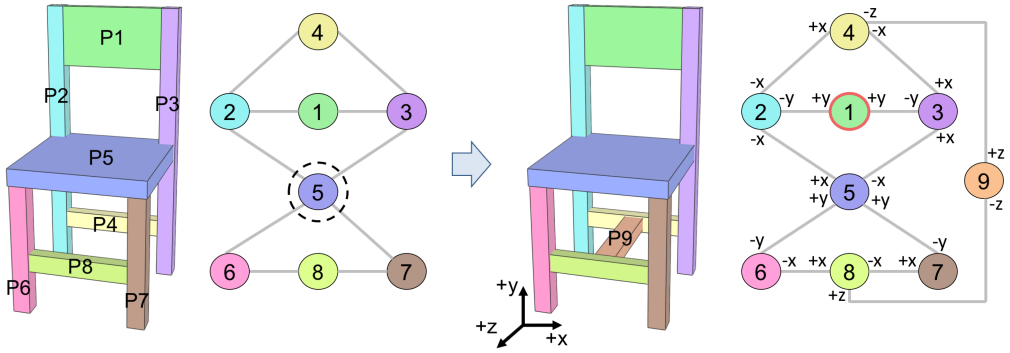


Figure 4.16 – Left: a Chair and its parts-graph, where a cut point (i.e., P_5) exists. Right: after adding a new part (i.e., P_9), our approach can generate an interlocking joint configuration, where the axial removal direction allowed by each joint is shown in the corresponding edge in the parts-graph.

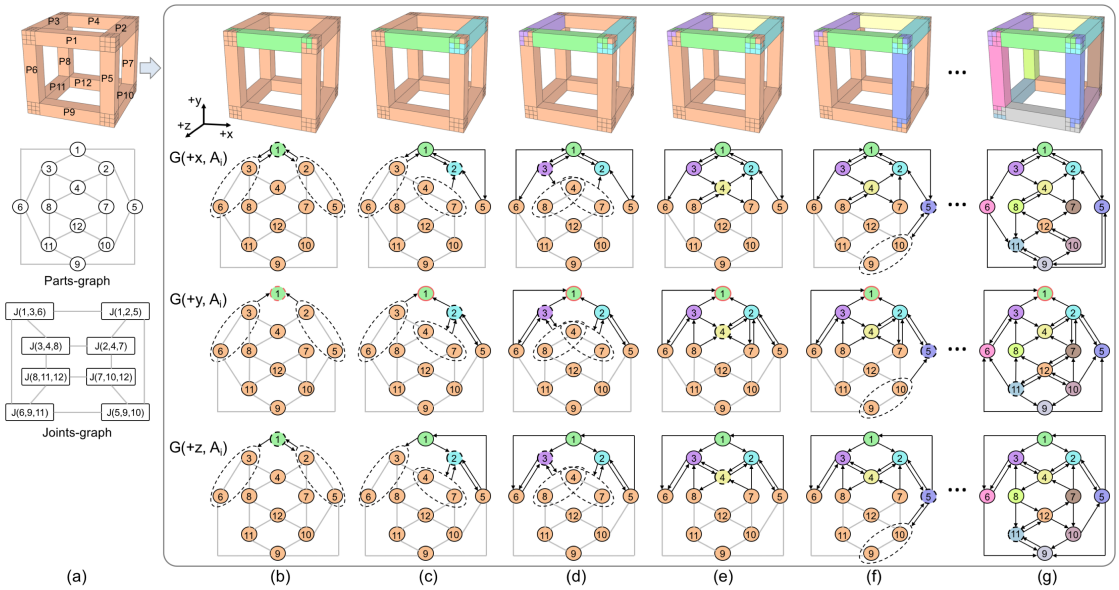


Figure 4.17 – Design of a 12-part interlocking FRAME CUBE. (a) Input frame design, parts-graph, and joints-graph. (b-f) The iterative procedure to construct the voxel joints, where the dashed ellipses highlight the set of parts in R_i that connect with P_j ($j \leq i$) using voxel joints. (g) The interlocking result.

4.3.3 Interlocking Frame Structures

As a new class of interlocking assembly, we propose *interlocking frame structures* that can be considered as a hybrid of the voxelized and the plate assemblies. A frame structure is a network of beams joined to represent a desired target shape. As input we assume a 3D polygonal mesh, where each edge represents a beam and each vertex represents a joint.

Compared with the plate assemblies, frame structures have two more challenges. First, frame structures require connecting more than two parts at a joint, making traditional woodworking

joints unsuitable; see the joints-graph in Figure 4.17(a). Thus, we propose to connect the beams with cube-shaped *voxel joints*. To make the problem tractable, we assume that each face of the joint connects to at most one beam at the center voxel, and thus each joint connects at most six beams (i.e., valence of the input mesh should be at most 6). Second, we need to individually optimize the geometry of each voxel joint. We place an axis-aligned $3 \times 3 \times 3$ cube at each joint location and partition the cubes into partial joints to restrict the relative movement of the connected beams; see the eight corners of the FRAME CUBE in Figure 4.17(a). Compared to the plate structures, these specifics require the following adaptations to our framework:

1. *Generating the key.* For the voxel joint at each end of P_1, we take the voxel preassigned to P_1 as a seed and include more voxels to P_1 such that it is movable along a singe axial direction following Subsection 4.2.2. Denote the subset of parts that connect with P_1 at each end as $S_1^k = \{P_l\}$, where $k \in \{1,2\}$; see the dashed circles in Figure 4.17(b). After constructing P_1, we draw directed edges between P_1 and each S_1^k in the DBGs accordingly.

2. *Generating P_i and R_i ($i > 1$).* At the graph design stage, we classify parts in each S_i^k into two groups: $\{\grave{P}_l\}$ where each part is from $\{P_1, ..., P_{i-1}\}$ and $\{\acute{P}_l\}$ where each part is from R_i. If $\{\acute{P}_l\}$ is empty, the blocking relations within this voxel joint are completely defined. So the graph design of P_i and R_i can be skipped for this joint; see joint $J(1,2,5)$ in Figure 4.17(f), where $P_i = P_5$, $\{\grave{P}_l\} = \{P_1, P_2\}$. If $\{\grave{P}_l\}$ is empty, we do not need to distribute external blocking relations in this joint; see joint $J(2,4,7)$ in Figure 4.17(c), where $P_i = P_2$, $\{\acute{P}_l\} = \{P_4, P_7\}$. Otherwise, we distribute external blocking relations associated with $\{\grave{P}_l\}$ to P_i and parts in $\{\acute{P}_l\}$ respectively; see joint $J(1,2,5)$ in Figure 4.17(c). Constructing internal blocking relations is restricted to P_i and parts in $\{\acute{P}_l\}$ for each S_i^k. To ensure that P_i is disassemblable in $[P_i, R_i]$, say along d_i, we construct a single directed edge between P_i and each part in $\{\acute{P}_l\}$ for both S_i^k in at least one DBG; see $G(+x, A_2)$ in Figure 4.17(c) for an example.

   The geometry realization of P_i and R_i is conducted within the voxel joint at each end of P_i following the approach in Subsection 4.2.3; see corners of the FRAME CUBE in Figure 4.17(c-f).

Figure 4.17(g) shows the resulting interlocking FRAME CUBE, in which every three beams joining at a corner are connected by a carefully constructed three-way voxel joint. Note that all joints are distinct even though all corners are symmetric. This illustrates how the assembly order dictates the geometry of the joints, more so than the geometry of the parts.

Our approach can generate frame structures with different joint valence, e.g., the valence in the FRAME CHAIR in Figure 4.18 can be 2, 3 or 4. We fabricate this result using wooden pillars and small wooden cubes to validate its steadiness; see video https://youtu.be/Pqwo3GbxBEo for the live demo. The FLOWER in Figure 4.1 shows another result, where curved beams are

Figure 4.18 – Interlocking $1.0m \times 0.5m \times 0.5m$ Frame Chair. The voxel joints are fabricated by gluing wooden cubes in the spatial arrangement computed by our algorithm. When attached to the corresponding wooden pillars, all pillars can be connected into a steady interlocking assembly.

connected by the voxel joints to form an appealing structure. Our approach can generate frame structures with a large number of parts; see Figure 4.19. To the best of our knowledge, these are the first *single-key* interlocking frame structures.

Table 4.1 – Statistics of results generated by our framework. The labels in 4rd to 6th columns refer to the number N of parts, the number M of base directions, and the time for generating the result.

	Fig.	Model	N	M	Time
Voxelized Structure	1	Cartoon Dog	14	3	1.06 hour
	10	4x4x4 Cube	9	3	1.13 hour
	12	Bunny	80	3	0.74 hour
	12	35x35x35 Cube	1500	3	3.33 hour
Plate Structure	1	Lizard	33	45	2.73 sec
	1	Bookshelf	12	3	0.30 sec
	13	Cabinet	7	5	0.09 sec
	17	Table	6	3	0.02 sec
	18	Chair	8	3	0.22 sec
Frame Structure	1	Flower	23	3	4.10 sec
	19	Frame Cube	12	3	0.53 sec
	20	Frame Chair	11	3	15.66 sec
	Inset	Scaffold	92	3	26.41 sec

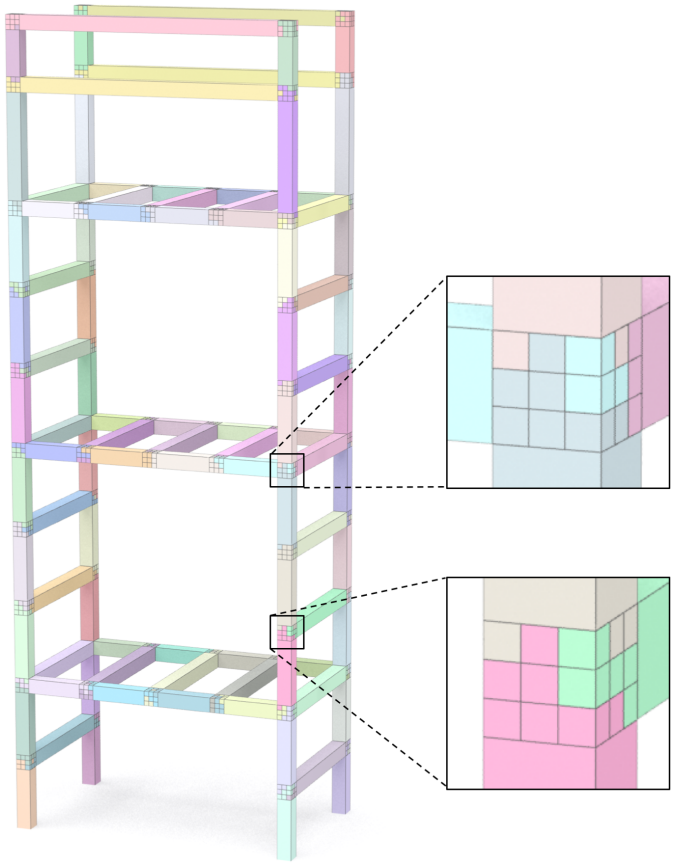


Figure 4.19 – A 92-part Scaffold connected with voxel joints.

4.3.4 Implementation and Performance

Our C++ implementation runs on an iMac with a 4.2GHz CPU and 32GB memory. In general, the timing performance depends on the input model, the number of parts N, and additional design requirements; see Table 4.1. The computation time to create interlocking voxelized structures highly depends on the number of desired parts. For example, the interlocking $4 \times 4 \times 4$ CUBES (Figure 4.7) with 7, 8, and 9 parts take 0.3 seconds, 12 seconds, and 1.13 hours respectively. For larger N, it becomes increasingly difficult to find an interlocking assembly since smaller parts have fewer potential blocking contacts. Enforcing additional design requirements can also increase the computation time substantially. For example, creating CARTOON DOG (Figure 4.1) without constraints takes 23.3 seconds, while incorporating the appearance constraints increases computation to 1.06 hours, since significantly more backtracking is required to ensure that the constructed parts align with the features. For more details on the implementation of our approach, we refer to the source code in our github https://github.com/KIKI007/DESIA.

Our approach creates interlocking plate structures very efficiently due to the relatively small search space. For example, it takes 0.07 seconds to compute CABINET in Figure 4.10. Due to

Figure 4.20 – The assembling sequence of a globally interlocking bench designed by Ulysse Martel using our software.

the non-orthogonal part connections in the CABINET, it further takes 0.02 seconds to verify interlocking by performing the inequality-based test. Hence, the total time to generate this result is 0.09 seconds; see Table 4.1. The other result with non-orthogonal part connections (i.e., LIZARD in Figure 4.1) takes 0.03 seconds to verify its interlocking. Designing frame structures is also fast since the $3 \times 3 \times 3$ cubes that are optimized at each joint location have only 27 voxels. In particular, it takes much longer time to generate the FRAME CHAIR than the FRAME CUBE although they have similar number of parts; see Table 4.1. This is mainly due to the lower number of cycles in the parts-graph of the FRAME CHAIR, which makes it harder to find interlocking configurations.

4.4 Limitations and Future Work

Our work has several limitations that open up interesting directions for future research. First, the DBG-based representation models only the infinitesimal translational motions of the parts. Finding a conceptual representation that supports extended translational motions (e.g., for avoiding global collision) would be helpful for testing interlocking and disassemblability in a unified manner. Currently we assume planar inter-part contact and translational assembly motion. While this simplifies the conceptual representation as well as the fabrication and construction of interlocking assemblies, generalizations to non-planar contact and more

complex assembly motions [Zhang et al., 2018] could lead to new types of assemblies. We currently do not analyze the structural implications of the way individual parts are connected. As a future work, it would be valuable to optimize the stress distribution to avoid local stress concentrations in the assembly. Another important aspect that is currently not covered in our work is tolerance handling. Fabrication imprecisions lead to deviations in the part geometries that can accumulate and negatively impact the stability of the assembly. How to design for robustness against such error accumulation is an exciting future research problem. The frame structure that we introduce in this chapter is just one instance of a broader class of possible assemblies where joint geometries are optimized together with the assembly, instead of being selected from a set of predefined joint types. Voxelized cube joints do not necessarily provide the most appropriate connection and novel joint typologies could be discovered in the future that are better suited for the kind of multi-part joints that we studied in this chapter. Other potential directions for future work include assemblies of deformable parts [Skouras et al., 2015] or reconfigurable assemblies.

4.5 Acknowledgments

We thank the reviewers for the valuable comments, Zhe Chen and Fengyuan Yang for their help in fabricating the frame chair, and Mina Konaković-Luković for proofreading the paper. This work was supported by the NCCR Digital Fabrication, funded by the Swiss National Science Foundation, NCCR Digital Fabrication Agreement #51NF40-141853.

5 Computational Design of Topological Interlocking Assemblies

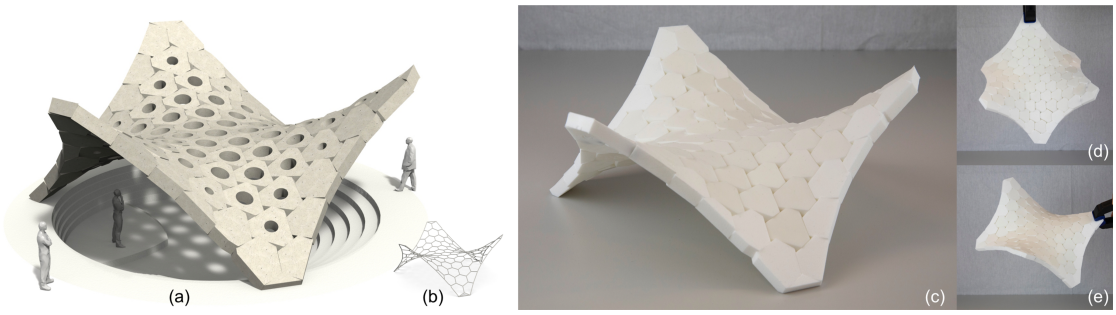


Figure 5.1 – A topological interlocking assembly (a) designed with our approach to conform to an input freeform design surface (b). The 3D printed prototype (c-e) is stable under different orientations.

We study assemblies of convex rigid blocks regularly arranged to approximate a given freeform surface. Our designs rely solely on the geometric arrangement of blocks to form a stable assembly, neither requiring explicit connectors or complex joints, nor relying on friction between blocks. The convexity of the blocks simplifies fabrication, as they can be easily cut from different materials such as stone, wood, or foam. However, designing stable assemblies is challenging, since adjacent pairs of blocks are restricted in their relative motion only in the direction orthogonal to a single common planar interface surface. We show that despite this weak interaction, structurally stable, and in some cases, globally interlocking assemblies can be found for a variety of freeform designs. Our optimization algorithm is based on a theoretical link between static equilibrium conditions and a geometric, global interlocking property of the assembly—that an assembly is globally interlocking if and only if the equilibrium conditions are satisfied for arbitrary external forces and torques. Inspired by this connection, we define a measure of stability that spans from single-load equilibrium to global interlocking, motivated by tilt analysis experiments used in structural engineering. We use this measure to optimize the geometry of blocks to achieve a static equilibrium for a maximal cone of directions, as opposed to considering only self-load scenarios with a single gravity direction. In the limit, this optimization can achieve globally interlocking structures. We show how different

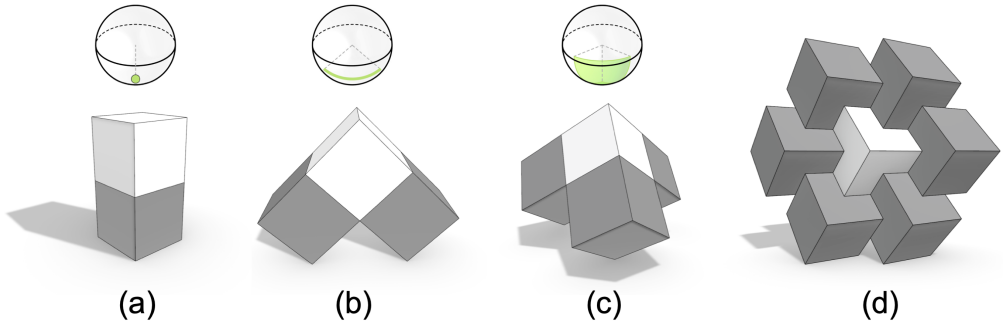


Figure 5.2 – Stability of cubes. Assuming the dark gray cubes are fixed and no friction forces act on the contact surfaces, the light gray cube is supported for a single gravity direction (a), an arc of directions (b), a patch of directions (c), and all directions (d).

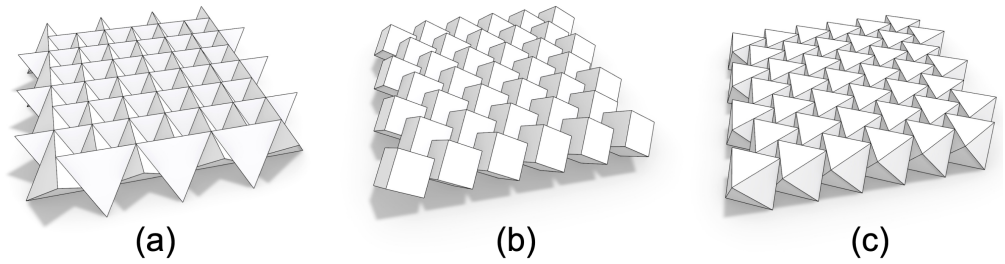


Figure 5.3 – Example planar TI assemblies described in [Dyskin et al., 2003a] composed of (a) tetrahedrons, (b) cubes, and (c) octahedrons.

geometric patterns give rise to a variety of design options and validate our results with physical prototypes.

5.1 Introduction

This chapter is about assemblies of convex rigid blocks. More specifically, we study how an ensemble of convex blocks, arranged in a regular topology to approximate a freeform surface, can form stable assemblies. Consider the two cubes of Figure 5.2-a. Assuming no friction forces act on the interface, a necessary condition for the cubes to form a stable stack is that the contact plane is orthogonal to the direction of gravity. In addition, we require that the center of gravity of the top block projects down into the contact polygon. This arrangement is in static equilibrium, but this equilibrium itself is not stable. Even the slightest tilt of the stack in any direction will cause the top block to slide and topple off.

Now let us consider three blocks (Figure 5.2-b). The light gray block on top has two contacts, each constraining its motion in the direction orthogonal to the contact plane. In this case, we can tilt the ensemble around the axis defined by the intersection of the contact planes and retain an equilibrium state. Effectively, the space of tilt directions under which the assembly is in equilibrium has been expanded from a single point to a 1D arc.

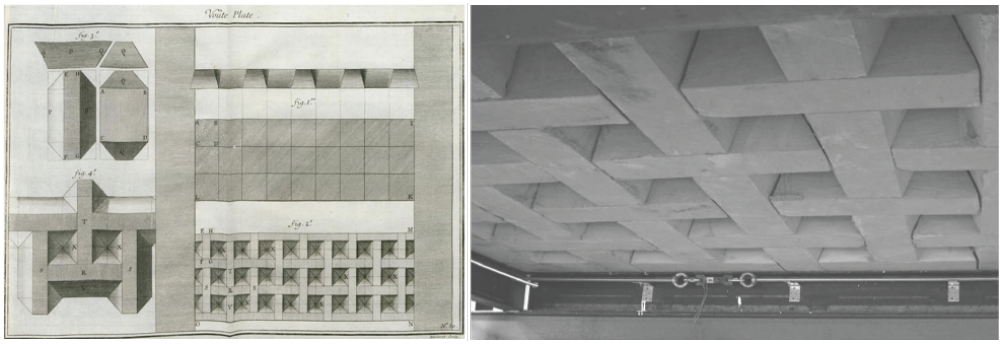


Figure 5.4 – The Abeille vault (sketch from 1734 on the left) is a globally interlocking assembly composed of identical convex blocks that form a planar roof structure (right).

Adding a third support block allows creating a 2D set of equilibrium directions (Figure 5.2-c). The top block is now in a stable configuration when tilting the ground plane around an arbitrary axis in some limited angle range. For a single cube to be completely immobilized no matter how the ensemble is rotated, we would need to constrain all six contact planes (Figure 5.2-d). In fact, this arrangement can be extended to form a regular assembly of cubes that cover the plane (see Figure 5.3-b). This specific pattern forms a so-called *topological interlocking* (TI) assembly [Dyskin et al., 2019]. Assuming the boundary is fixed, all interior cubes are mutually blocking each other. More specifically, this arrangement also forms a *globally interlocking* structure, where each part and each subset of parts is immobilized [Song et al., 2012].

Do we always need six neighbors to completely immobilize a convex element? The answer is no, because we can construct a non-empty polytope by intersecting four planes, hence a block constructed in this way can be completely immobilized by four neighbors. This construction is well known and has been used, for example, in the Abeille vault structure shown in Figure 5.4. These types of planar regular assemblies composed of identical blocks have been extensively studied in material science and mechanical engineering, where they have proven to have superior structural properties; see detailed discussion in Section 2.2.3.

So can we extend this concept from planar assemblies to curved freeform surfaces? And can we retain the advantageous structural properties of planar TI assemblies, in particular the global interlocking property? It is clear that we cannot use identical elements if we want to closely approximate a double curved surface. However, we can easily modify the shapes of the blocks so that the assembly conforms well to a given design surface [Fallacara et al., 2019]. For example, the assembly shown in Figure 5.5, created by a constructive method detailed later, well approximates a spherical design surface. Each block in this assembly is immobilized by its neighbors, i.e., no block can move if we assume each adjacent block is fixed. With the global peripheral constraint given by a complete ring of fixed boundary blocks shown in dark gray, the assembly should then be globally interlocking.

Unfortunately, this reasoning is flawed. While it is true that no block can move individually,

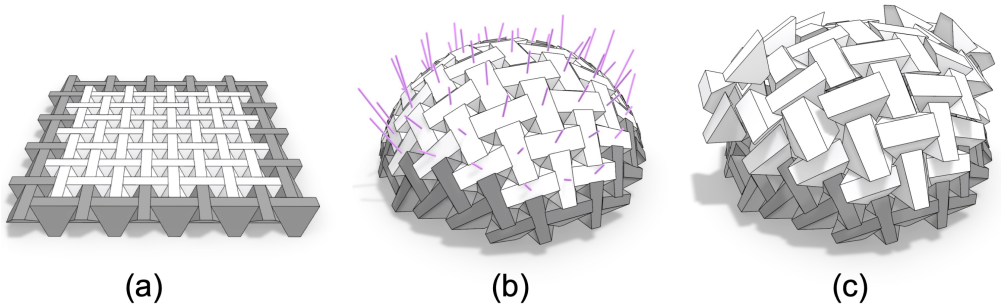


Figure 5.5 – A globally interlocking assembly following Abeille's construction (a) is lifted onto a spherical surface (b). While each block is still immobilized with respect to its neighbors, a subset of blocks can move simultaneously along different trajectories (c). The purple lines in (b) indicate the instantaneous velocity of each block for this disassembly motion. As a consequence, the assembly is *not* globally interlocking.

sub-groups of blocks can move *simultaneously* along different trajectories. This kind of multi-part instability is not captured by existing methods and provides a first indication that globally interlocking TI assemblies are difficult to obtain for curved surfaces. Motivated by this observation, we focus on designing TI assemblies that are as close to global interlocking as possible.

Contributions. Our goal is to design structurally stable assemblies with convex rigid blocks that closely conform to a given design surface. We focus in this chapter on convex elements with planar faces because they can easily be fabricated by blade or wire cutting, but also because of their superior structural integrity [Alexandrov, 2005]. Geometrically, however, the convexity of blocks poses the biggest challenge in terms of creating stable assemblies, because two adjacent parts are only constrained in one direction by their mutual contact. To address this challenge, we make the following contributions:

1. We introduce a new, general algorithm to test for global interlocking that considers not only part translation, but also rotation, thus avoiding false positives that can occur with existing methods.

2. We formalize a theoretical link between static equilibrium conditions and a global interlocking property with a mathematical proof.

3. We propose a quantitative measure for structural stability of assemblies and present a gradient-based method that optimizes the geometry of blocks to maximize this measure.

4. We develop an interactive design tool that allows a real-time preview and efficient exploration of a wide range of design parameters of TI assemblies.

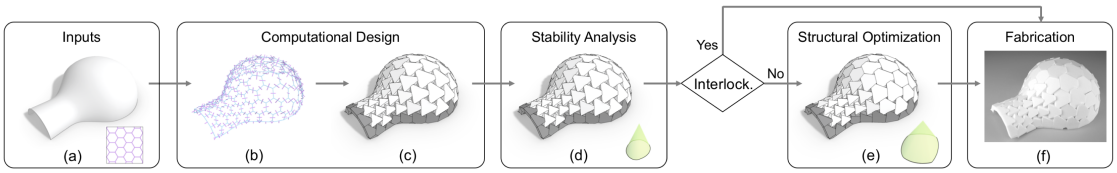


Figure 5.6 – Overview of our approach. (a) Input reference surface and 2D tessellation. (b) 3D surface tessellation with augmented vectors. (c) Initial TI assembly. (d) Stability analysis computes the cone of stable directions. (e) Structural optimization improves stability, i.e., the cone becomes larger. (f) 3D printed prototype.

Overview. The rest of the chapter is organized as follows: Section 5.2 introduces a parametric model for TI assemblies that facilitates a constructive approach for design exploration. Section 5.3 presents a gradient-based optimization to improve the structural stability of an assembly with respect to the measure. In Section 5.4 we show and discuss a variety of TI assemblies designed by our approach. We conclude with a discussion of limitations of our approach and identify opportunities for future research.

5.2 Computational Design of TI Assemblies

Given a reference surface **S** as input, our goal is to design a structurally stable TI assembly **P** that closely conforms to **S**. To make this problem tractable, we first define a parametric model that facilitates a constructive approach for design exploration of TI assemblies. We then show in Section 5.3 how to optimize for the structural stability of a designed assembly. Figure 5.6 gives a high-level overview of our computational design pipeline.

5.2.1 Parametric Model

Dyskin et al. [Dyskin et al., 2013] proposed a parametric model for planar TI assemblies based on a 2D polygonal tessellation **T** in which the edges are augmented with normalized vectors. We extend this model to parameterize 3D free-form TI assemblies using a 3D surface tessellation **T** with augmented vectors.

Parameter space Specifically, we represent **T** as a polygon mesh using a half-edge data structure, where each face is denoted as T_i and each pair of half-edges shared by T_i and T_j is denoted as e_{ij} on T_i and e_{ji} on T_j. The 3D directional vector defined by each half-edge e_{ij} is denoted as $\mathbf{e}_{ij}$ with $\|\mathbf{e}_{ij}\| = 1$. We assign to each T_i a normal vector $\mathbf{N}_i$ defined by the least-squares plane of the polygon's vertices. We further augment each half-edge e_{ij} with a 3D vector $\mathbf{n}_{ij}$ where $\|\mathbf{n}_{ij}\| = 1$ and $\mathbf{n}_{ij} \perp \mathbf{e}_{ij}$; see Figure 5.7-a. Each half-edge e_{ij} together with the augmented vector $\mathbf{n}_{ij}$ defines a 3D plane.

Block geometry We intersect all 3D planes associated with the half-edges of each T_i to construct the (convex) geometry of the corresponding block P_i in **P**; see Figure 5.7-b. Sometimes,

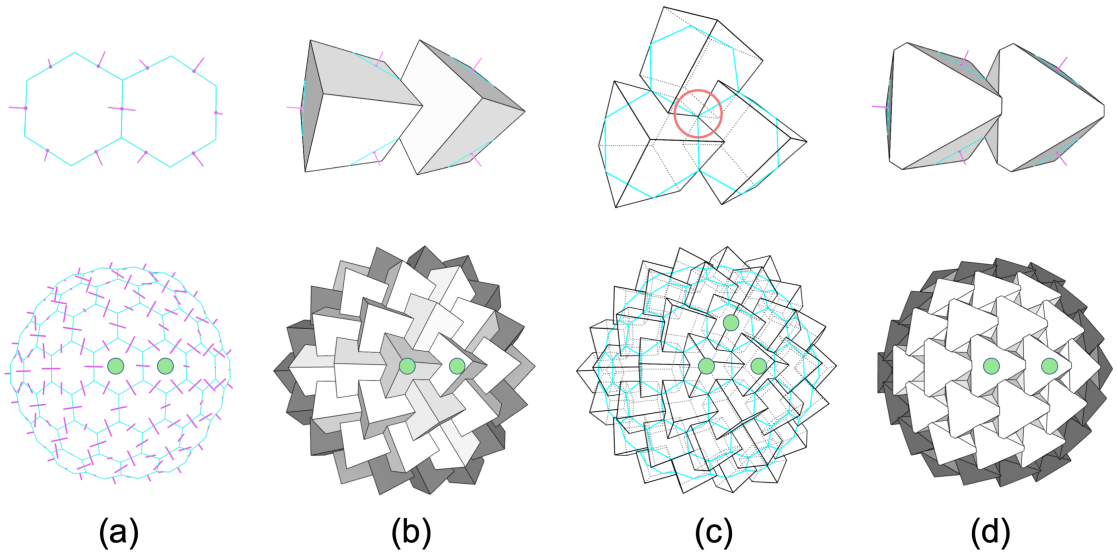


Figure 5.7 – A TI assembly is created from a 3D surface tessellation and a set of augmented vectors (a) by intersecting the half-spaces defined by each tessellation polygon (b). Each vertex of the tessellation corresponds to the joining point of neighboring blocks; see the red circle (c). Blocks can be additionally trimmed with surface offset planes (d). Zooming views of the faces/blocks highlighted with green dots are shown on the top row.

the intersected block geometry could be infinite (or simply too bulky), so we optionally trim the blocks using offset planes with normal $\pm\mathbf{N}_i$; see Figure 5.7-d. Blocks corresponding to faces T_i in $\mathbf{T}$ that contain a boundary edge will be merged to form the boundary frame, shown in darker shading in the figures. The resulting boundary frame can also be clamped to a smooth outline; see for example Figure 5.11.

Valid assemblies For the above construction method to produce a geometrically and structurally valid assembly, we restrict $\mathbf{T}$ to only contain convex faces that are not triangles as these would produce pyramid-shaped elements that cannot properly "interlock" with other blocks. We require $\mathbf{n}_{ij} = -\mathbf{n}_{ji}$ to ensure a proper planar contact face between adjacent blocks. We further require that each face T_i is in the half-space $(\mathbf{v} - \mathbf{v}_{ij}) \cdot \mathbf{n}_{ij} \leq 0$ defined by each augmented half-edge of T_i, where $\mathbf{v}$ is an arbitrary point and $\mathbf{v}_{ij}$ is a point on the edge e_{ij}. This will ensure that the intersected geometry of P_i defined by the 3D planes $\{e_{ij}, \mathbf{n}_{ij}\}$ is not empty and encompasses the face T_i; see the zooming views in Figure 5.7-b&c.

5.2.2 Interactive Design

To initialize a design, the user selects a tessellation pattern, adapts global alignment and scaling, and assigns initial augmented vectors. An automatic procedure that checks the above geometric requirements then provides immediate feedback on the assembly's validity. Specifically, our computational approach proceeds as follows:

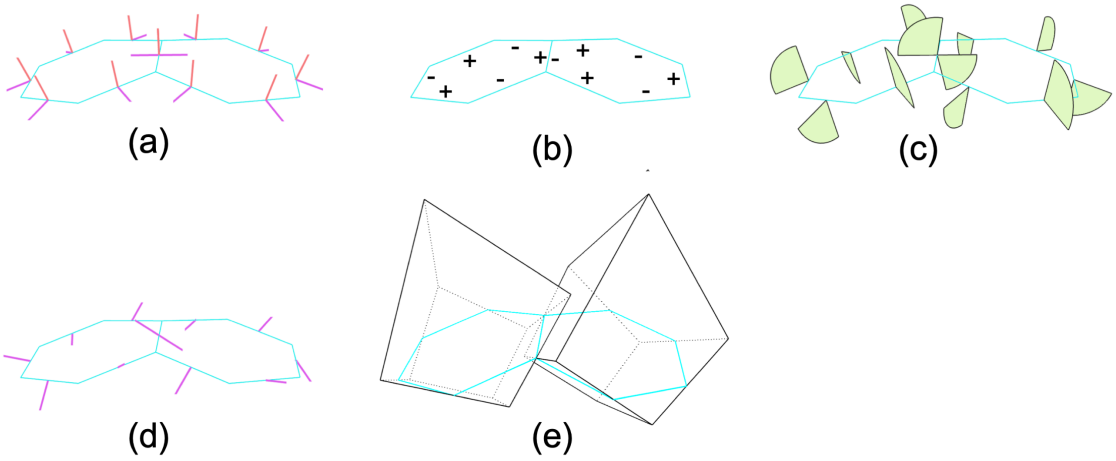


Figure 5.8 – (a) Initialize $\mathbf{n}_{ij}$ (in purple), where the red vector is $(\mathbf{N}_i + \mathbf{N}_j)/\|\mathbf{N}_i + \mathbf{N}_j\|$. (b) Determine orientations of $\mathbf{n}_{ij}$, where + (−) indicates clockwise (counterclockwise) rotation around $\mathbf{e}_{ij}$. (c) Compute range for each $\mathbf{n}_{ij}$ (visualized as green sectors). (d) Example $\{\mathbf{n}_{ij}\}$ generated with user-specified $\alpha = 35°$. (e) Two resulting blocks.

Initialize tessellation Given a reference surface **S**, there are many different ways to create a surface tessellation **T**, including remeshing, surface Voronoi diagrams, or parameterization approaches. Our tool mainly uses conformal maps to lift a planar tessellation onto the surface; see Figure 5.11 for examples. The user can interactively adjust the location, orientation, and scale of the tessellation. We further optimize the vertex positions using a projection-based optimization [Bouaziz et al., 2012, Deuss et al., 2015] to improve planarity and regularity of the 3D polygons and to ensure proper contacts among blocks by avoiding small dihedral angles. Please refer to the supplementary material (Appendix B.1) for details about this optimization.

Assign vectors $\{\mathbf{n}_{ij}\}$ For each half-edge e_{ij} in the tessellation **T**, we initialize $\mathbf{n}_{ij}$ as $\mathbf{e}_{ij} \times (\mathbf{N}_i + \mathbf{N}_j)$ after normalization (see Figure 5.8-a). We then rotate $\mathbf{n}_{ij}$ around $\mathbf{e}_{ij}$ by an angle $x_{ij}\alpha_{ij}$, where each α_{ij} is initialized as a user-specified rotation angle α and $x_{ij} \in \{-1, 1\}$ specifies rotational direction (i.e., clockwise or counterclockwise). The goal here is to obtain alternating directions for adjacent edges of a polygon to improve the interlocking capabilities of blocks [Dyskin et al., 2019]. For this purpose, we use a simple flood-fill algorithm that starts with a random edge and traverses the half-edge data structure to assign $\{x_{ij}\}$ that locally maximize adjacent sign alternations. If all polygons have an even number of edges, this strategy can achieve global alternation (see Figure 5.8-b), which cannot be guaranteed in general, i.e., when the tessellation contains polygons with odd number of edges.

Select rotation angle α The global parameter α can be interactively controlled by the user. For each edge we compute an allowable range $[\alpha_{ij}^{\min}, \alpha_{ij}^{\max}]$ that ensures a valid block geometry as defined above and clamp the applied rotation accordingly. Figure 5.9 shows 3D tessellations generated with different values of α. Due to the efficiency of this construction approach, the user can interactively create and preview TI assemblies while adjusting the design parameters, i.e. the 2D tessellation and its mapping onto the reference surface, the rotation angle α, and

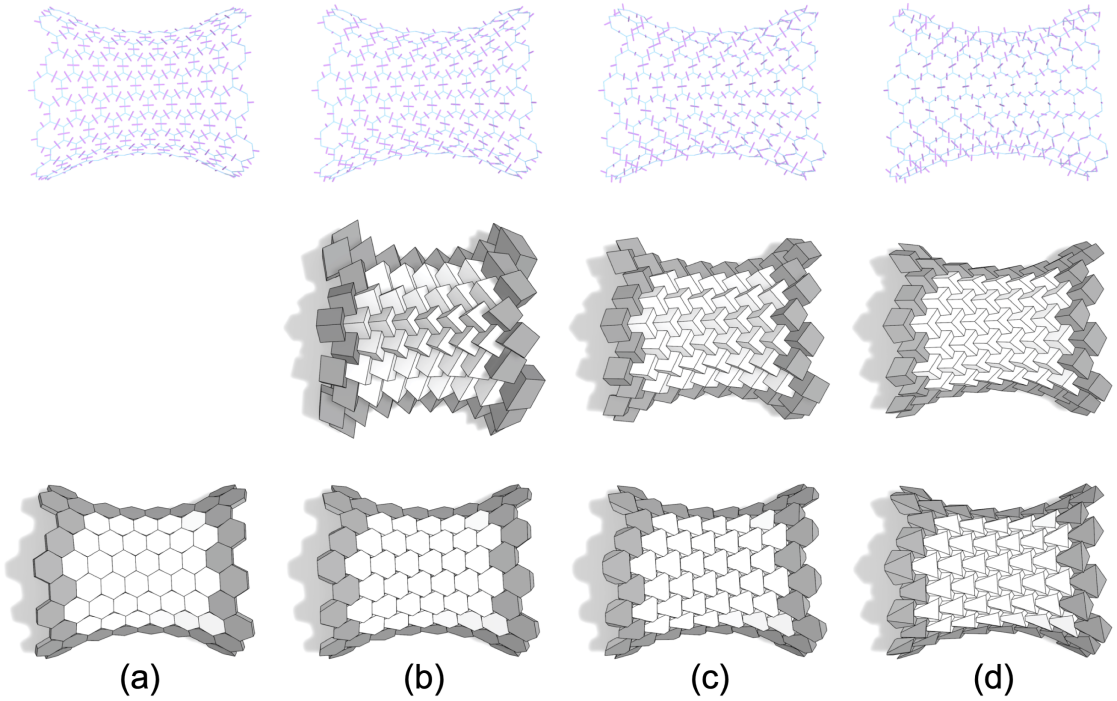


Figure 5.9 – TI assemblies generated with α equals to (a) 0°, (b) 25°, (c) 45°, and 65°. From top to bottom: 3D surface tessellation with augmented vectors, TI assemblies with originally constructed blocks, and with trimmed blocks. Note that the originally constructed blocks in the TI assembly with $\alpha = 0°$ have infinite geometry and thus are not shown.

the thickness of the blocks.

5.3 Structural Optimization of TI Assemblies

The interactive design stage generates a TI assembly **P** as input to the subsequent stages of our computational pipeline (Figure 5.6-d&e). If our analysis algorithm of Section 3.19 reveals that **P** is globally interlocking, no further optimization is required. Otherwise, we run a structural optimization that distinguishes several cases; see Algorithm 2. To make these computations tractable, we only optimize the augmented vectors $\{\mathbf{n}_{ij}\}$ while fixing the tessellation **T**.

In detail, if the initial design is not in static equilibrium under gravity, we first run an optimization to find a stable state (Section 5.3.2). If a stable configuration is found, we evaluate its stability score as discussed in Section 3.6. If $\Phi = 180°$, then no further optimization is required. If $\Phi = 90°$, then finding a static equilibrium for any force direction in the upper hemisphere without breaking equilibrium for the directions in the lower hemisphere will result in $\Phi = 180°$ due to convexity of the feasible set $\mathbb{G}(\mathbf{P})$. We therefore simply run our optimization for all the six axial directions. Finally, if $\Phi \in [0°, 90°)$, we optimize stability for an incrementally growing cone of directions (Section 5.3.1).

Algorithm 2 Algorithm of structural optimization on a TI assembly **P** to improve its stability.

function StructuralOptimization(**P**)
  $S \leftarrow$ StabilityAnalysis(**P**) ▷ See Section 3.5.1
  if S = Interlocking **then**
    return
  else if S = NonEquilibrium **then**
    if OptimizeAssembly(**P**, $-z$) $\neq$ Success **then**
      return
    end if
  end if
  $\Phi \leftarrow$ ComputeStabilityMeasure(**P**) ▷ See Section 3.6
  if $\Phi = 180^\circ$ **then**
    return
  else if $\Phi = 90^\circ$ **then**
    if OptimizeAssembly(**P**, $\{\pm\mathbf{x}, \pm\mathbf{y}, \pm\mathbf{z}\}$) $\neq$ Success **then**
      return
    end if
  else
    $\omega \leftarrow \omega_c$ ▷ $\omega_c = 1.2$ in our experiments
    while $\omega > \omega_t$ **do** ▷ $\omega_t = 1.01$ in our experiments
      $\Phi_{\text{tagt}} \leftarrow \omega\,\Phi$
      $\{\mathbf{d}_k\} \leftarrow$ ComputeTargetDirections(Φ_{tagt}, **P**)
      if OptimizeAssembly(**P**, $\{\mathbf{d}_k\}$) = Success **then**
        $\omega \leftarrow \omega_c$
        $\mathbf{P} \leftarrow \mathbf{P}^*$ ▷ $\mathbf{P}^*$ is the optimized assembly
      else
        $\omega \leftarrow \delta\,\omega$ ▷ $\delta = 0.95$ in our experiments
      end if
    end while
    return
  end if
end function

5.3.1 Compute Target Force Directions

Given a TI assembly **P** with stability measure $\Phi(\mathbf{P}) \in [0^\circ, 90^\circ)$, the radius of the largest inner circle (centered at the origin) of the feasible section $\mathbb{S}(\mathbf{P})$ is $\tan(\Phi)$; see again Figure 3.9(c). To improve the stability measure from Φ to $\Phi_{\text{tagt}} = \omega\Phi$ ($\omega > 1$), we need to modify the feasible section and enlarge the radius of its largest inner circle from $\tan(\Phi)$ to $\tan(\Phi_{\text{tagt}})$. To this end, we approximate the target feasible section $\mathbb{S}_{\text{tagt}}(\mathbf{P})$ with a convex polygon that completely contains the target circle with radius $\tan(\Phi_{\text{tagt}})$. The vertices $\{\mathbf{v}_k\}, 1 \le k \le K$, of the polygon should be as close as possible to the current feasible section $\mathbb{S}(\mathbf{P})$ to require minimal change

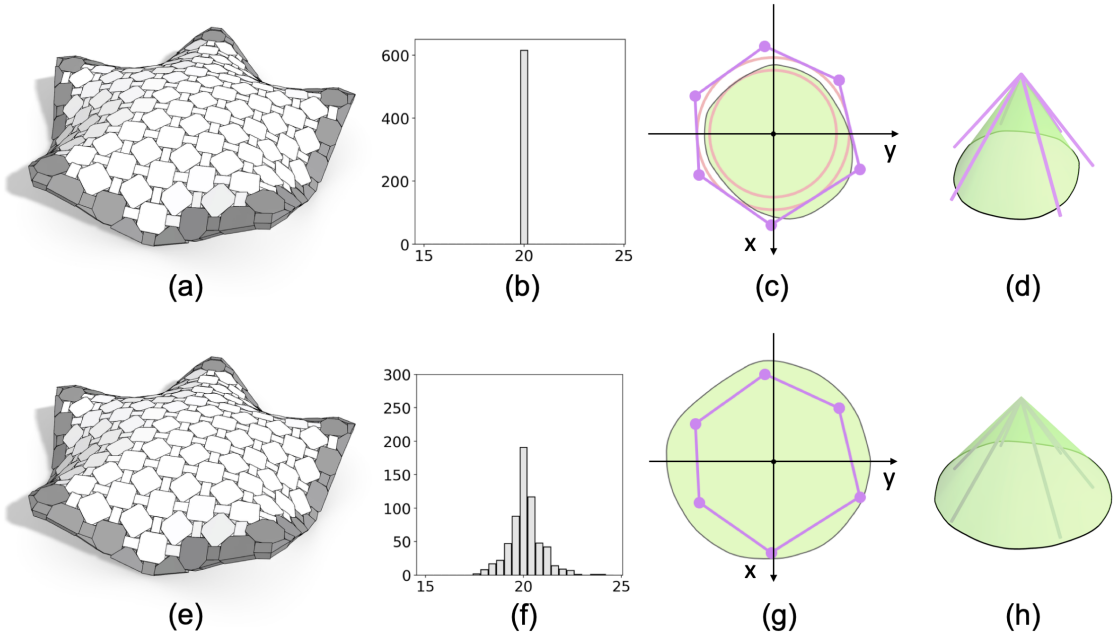


Figure 5.10 – An example TI assembly before (top) and after (bottom) one step of our optimization. (b&f) Histograms of vector rotation angles $\{\alpha_{ij}\}$, where each $\alpha_{ij} = \alpha$ (i.e., 20°) before the optimization. (c&d) Target force directions (in purple) in the feasible section and cone, where current and target circles are colored in red. (g&h) The optimization goal is achieved by including the target force directions in the optimized feasible section and cone.

to the geometry of the blocks. In our experiments, we choose $K = 6$ as a trade-off between computation efficiency and approximation accuracy.

We initialize the vertices as a regular K-sided polygon that encloses the target circle and optimize their positions to minimize their distances to the current feasible section $\mathbb{S}(\mathbf{P})$; see supplementary material for details about the optimization (Appendix B.2). Figure 5.10-c shows an example target feasible section $\mathbb{S}_{\text{tagt}}(\mathbf{P})$ approximated with a hexagon using our approach, together with the current and target inner circles. Each vertex of the target feasible section (i.e., the hexagon) corresponds to a 3D force direction that is usually outside of the current feasible cone $\mathbb{G}(\mathbf{P})$; see the purple lines in Figure 5.10-d. We denote these target force directions corresponding to $\{\mathbf{v}_k\}$ as $\{\mathbf{d}_k\}$.

5.3.2 Optimize TI Assembly

Given the set of target force directions $\{\mathbf{d}_k\}$, the goal of our optimization is to include each direction $\mathbf{d}_k$ in the feasible cone $\mathbb{G}(\mathbf{P}^*)$ of the optimized assembly $\mathbf{P}^*$. If this optimization succeeds, we enlarge Φ_{tagt}, recompute the target force directions $\{\mathbf{d}_k\}$, and repeat the optimization. Otherwise, we have to lower the optimization goal by decreasing Φ_{tagt}, and repeat the optimization. Our optimization terminates when the stability measure Φ cannot be improved any more; see again Algorithm 2.

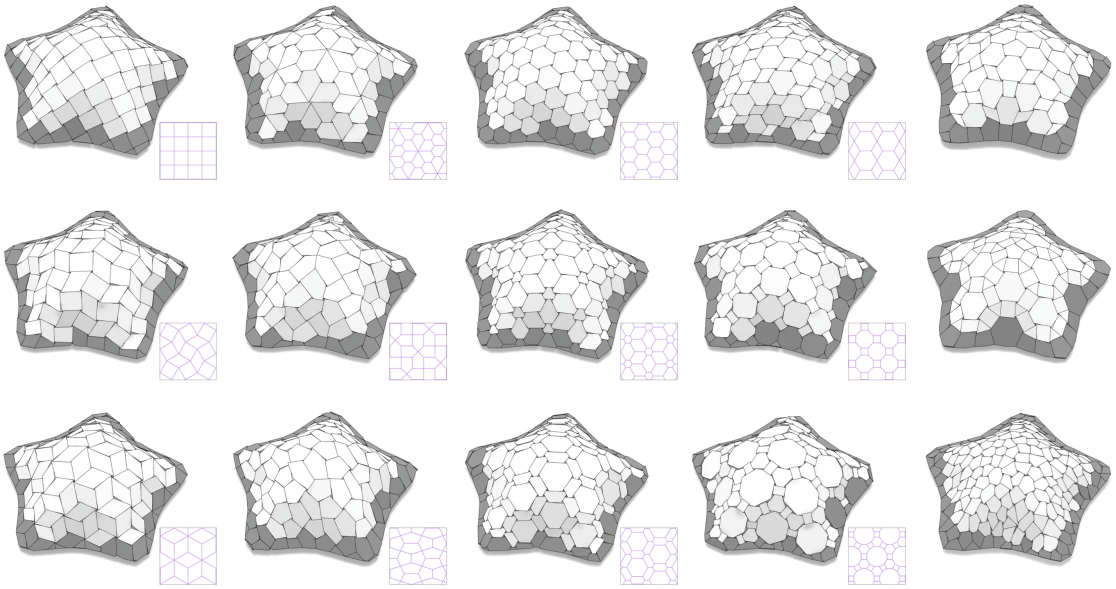


Figure 5.11 – A variety of patterns supported by our tool for designing TI assemblies. The surface tessellations can be generated by lifting 2D tessellations (see the boxed images) using conformal maps (the left four columns), manually designed by users (top two patterns in the rightmost column), or created as a surface Voronoi diagram (bottom pattern in the rightmost column).

Problem Formulation. Our optimization can be formulated as:

$$\{\alpha_{ij}^*\} = \arg\min_{\{\alpha_{ij}\}} \sum_{k=1}^{K} E(\,\mathbf{P},\,\mathbf{d}_k\,) \tag{5.1}$$

$$\begin{aligned} \text{s.t.} \qquad & \text{Area}(\,C_l\,) \geq A_{\text{thres}}, \quad \forall \text{ contact } C_l \\ & \max(\alpha^{\min}, \alpha_{ij}^{\min}) \leq \alpha_{ij} \leq \min(\alpha^{\max}, \alpha_{ij}^{\max}) \end{aligned}$$

where $E(\mathbf{P}, \mathbf{d}_k)$ quantifies the assembly's infeasibility to be in equilibrium under external forces along direction $\mathbf{d}_k$; A_{thres} is the minimum allowable contact size among the blocks (for face-face contacts only); and $[\alpha^{\min}, \alpha^{\max}]$ is the user-specified range for every α_{ij} to preserve the assembly appearance while $[\alpha_{ij}^{\min}, \alpha_{ij}^{\max}]$ is the range for constructing blocks with valid geometry (see again Figure 5.8-c).

We compute the energy $E(\mathbf{P},\,\mathbf{d}_k)$ following the approach in [Whiting et al., 2012], where the key idea is to allow tension forces to act as "glue" at block interfaces to hold the assembly together, to penalize the tension forces, and to use their magnitude to quantify the infeasibility to be in static equilibrium.

In detail, we first express each contact force f_l^c at vertex c of contact C_l in terms of compression

and tension forces using the difference of two non-negative variables:

$$f_l^c = f_l^{c+} - f_l^{c-} \qquad f_l^{c+},\ f_l^{c-} \geq 0 \tag{5.2}$$

where f_l^{c+} and f_l^{c-} are the positive and negative parts of f_l^c, representing the compression and tension forces, respectively. Our objective is to minimize tension forces between the blocks in the assembly $\mathbf{P}$ under external forces along direction $\mathbf{d}_k$, subject to the equilibrium constraint:

$$E(\,\mathbf{P},\,\mathbf{d}_k\,) \;=\; \min_{\{\mathbf{f}_k\}} \mathbf{f}_k^T \mathbf{H} \mathbf{f}_k \tag{5.3}$$

$$\text{s.t.} \quad \mathbf{A}_{\text{eq}} \cdot \mathbf{F}_k = \mathbf{W}_k$$

where $\mathbf{f}_k$ is the vector of contact forces represented as $\{f_l^{c+}, f_l^{c-}\}$, $\mathbf{H}$ is a diagonal weighting matrix for the tension (large weights) and compression (small weights) forces, $\mathbf{W}_k$ is the vector of external forces acting on each block along direction $\mathbf{d}_k$, and $\mathbf{F}_k$ is the vector of contact forces.

Optimization Solver. Our optimization in Equation 5.1 is very similar to the equilibrium optimization of 3D masonry structures [Whiting et al., 2012]. Hence, we solve our optimization following the gradient-based approach in [Whiting et al., 2012] with several important differences:

1. Our optimization aims to achieve static equilibrium under forces along each target force direction in $\{\mathbf{d}_k\}$ respectively, rather than along a single gravitational direction.

2. Our assemblies do not rely on friction, so we eliminate the friction constraints in the equilibrium condition in Equation 5.3.

3. We compute the gradient of the energy $E(\mathbf{P}, \mathbf{d}_k)$ with respect to the vector rotation angles $\{\alpha_{ij}\}$, while [Whiting et al., 2012] computes it with respect to the positions of the block vertices; see the supplementary material for derivations of our gradients (Appendix B.3).

Figure 5.10 shows example TI assemblies before and after our optimization for a set of fixed target force directions $\{\mathbf{d}_k\}$. The histograms of the vector rotation angles $\{\alpha_{ij}\}$ show that our optimization adaptively adjusts these angles to make the assembly in static equilibrium for each of these force directions.

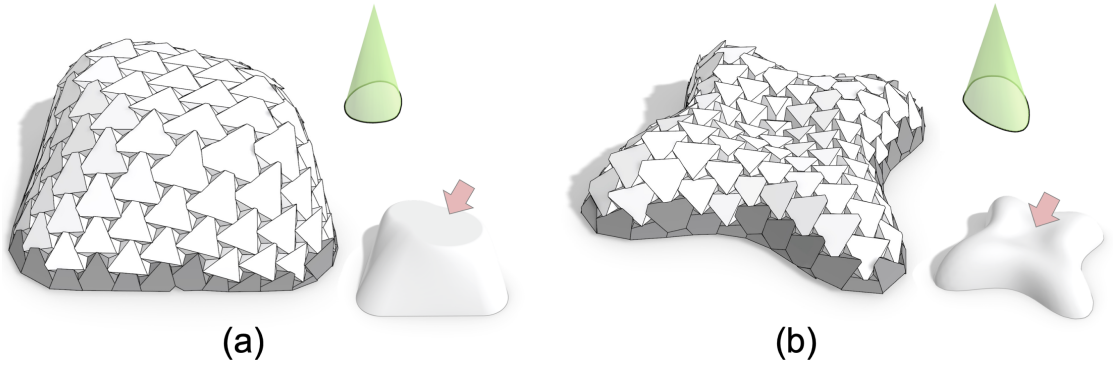


Figure 5.12 – Our method allows creating stable TI assemblies, indicated by the green feasible cones, even for design surfaces that are not self-supporting.

5.4 Results and Discussion

We implemented our tool in C++ and OpenGL, and employed MOSEK [ApS, 2019] and Knitro [Artelys, 2019] for solving our optimizations. We conducted all experiments on an iMac with a 4.2GHz CPU and 32GB memory. Our tool supports a variety of patterns as illustrated in Figure 5.11. We tested our design and optimization pipeline on a wide range of surfaces in Figure 5.13, e.g., FREE HOLES with high genus, FLOWER with zero mean curvature (i.e., minimal surface), and SURFACE VOUGA with both positive and negative Gaussian curvature. Figure 5.12 shows that our tool allows generating structurally stable TI assemblies from non-self-supporting surfaces, i.e. with a flat part or even an inverted bump on the top.

Table 5.1 summarizes the statistics of all the results presented in this chapter. The third to sixth columns list the total number of parts, the total number of contacts, and the number of contacts for each specific type, respectively. As can be seen, face-face contacts are dominant in all the results. The seventh to ninth columns show the stability measure Φ before and after the optimization, and the interlocking test result on the assembly. In general, stability improves significantly; e.g., from non-equilibrium to $\Phi = 53.8°$ for SURFACE VOUGA, from $\Phi = 33.8°$ to global interlocking for HYPERBOLIC. One interesting observation in our experiments is that TI assemblies constructed from a minimal surface are easy to be globally interlocking, even without structural optimization, e.g., ROOF in Figure 5.1 and FLOWER in Figure 5.13. The tenth and eleventh columns show timing statistics of the TI geometry initialization from given parameters (in milliseconds) and the complete structural optimization on the geometry (in minutes), respectively.

Fabricated Prototypes. We fabricated two TI assemblies designed by our tool, ROOF in Figure 5.1 and IGLOO in Figure 5.6, using an SLS 3D printer with PA 2200 polyamide material. To ensure assemblability of the structure, we break the boundary frame into two separated components. We compute the assembly sequence based on disassembly. Starting from one boundary component as the key, we iteratively identify blocks that can be taken out from the structure, until the remaining boundary component. Figure 5.15 shows the assembly

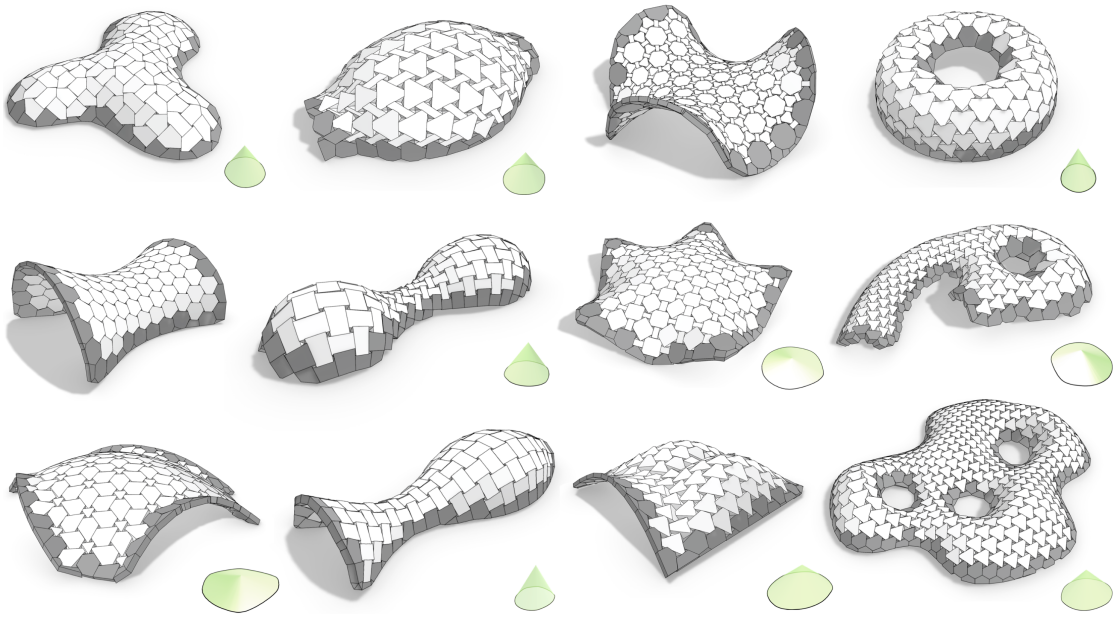


Figure 5.13 – TI assemblies of various shapes and their corresponding feasible cones (except those that are globally interlocking). From left to right and then top to bottom: BLOB, SPINDLE, FLOWER, TORUS, HYPERBOLIC, PEANUT, PENTAGON, SIX, BUGA PAVILION, VASE, SURFACE VOUGA, and FREE HOLES.

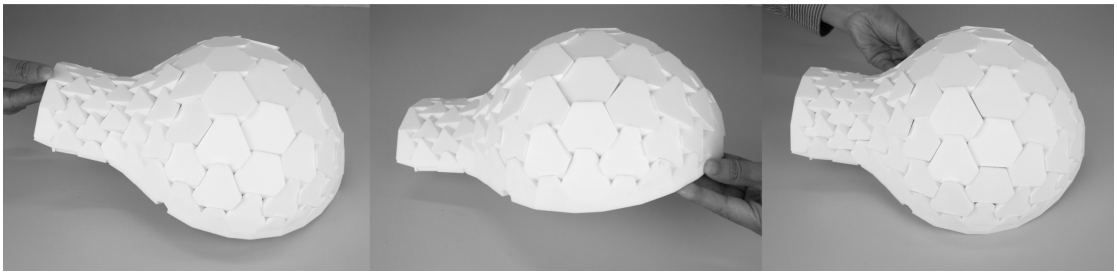


Figure 5.14 – Tilt analysis experiments on the 3D printed IGLOO to validate its stability.

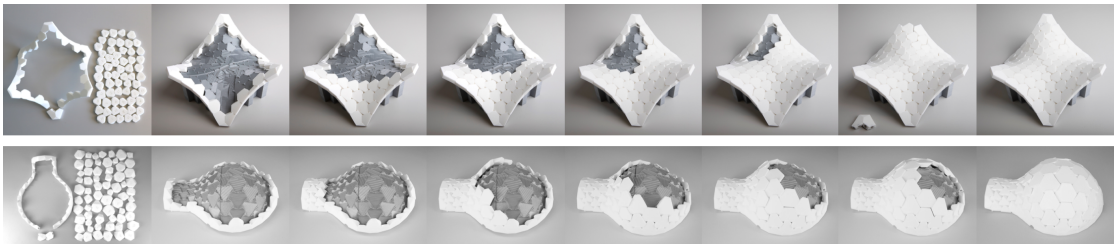


Figure 5.15 – Assembly sequence of (top) ROOF and (bottom) IGLOO, where the blocks and the boundary frame (in two components) are shown on the left. 3D printed formworks are used to support incomplete structures during the assembly.

Table 5.1 – Statistics of the resulting TI assemblies in this chapter.

Fig	Surface	# Part	# Contact	# Face-face contact	# Edge-edge contact	Before Optim. (degree)	After Optim. (degree)	Global Interlock	Init. Time (ms)	Optim. Time (min)
1	Roof	62	264	264	0	180.0	180.0	Yes	4.4	-
10	Igloo	64	274	274	0	22.2	29.4	No	4.7	30.7
16	Bump	110	408	408	0	0.8	13.4	No	2.3	474.4
	Lilium	105	432	432	0	-	16.2	No	5.6	259.8
17	Blob	113	522	381	141	-	31.7	No	5.1	798.7
	Spindle	133	589	426	163	1.4	31.8	No	6.3	78.1
	Flower	346	1192	1192	0	180.0	180.0	Yes	16.2	-
	Torus	120	507	507	0	10.6	25.7	No	6.7	54.5
	Hyperbolic	110	428	428	0	33.8	180.0	Yes	5.8	37.8
	Peanut	122	685	351	334	5.2	35.3	No	5.3	31.6
	Pentagon	190	740	740	0	31.9	78.5	No	12.7	471.1
	Six	138	580	580	0	33.1	65.0	No	7.7	1141.1
	Buga Pavilion	136	636	636	0	26.2	77.1	No	11.4	239.1
	Vase	95	565	291	274	14.6	36.5	No	4.7	24.1
	Surface Vouga	104	429	429	0	-	53.8	No	6.1	165.8
	Free Holes	345	1371	1371	0	18.3	46.6	No	17.4	340.0

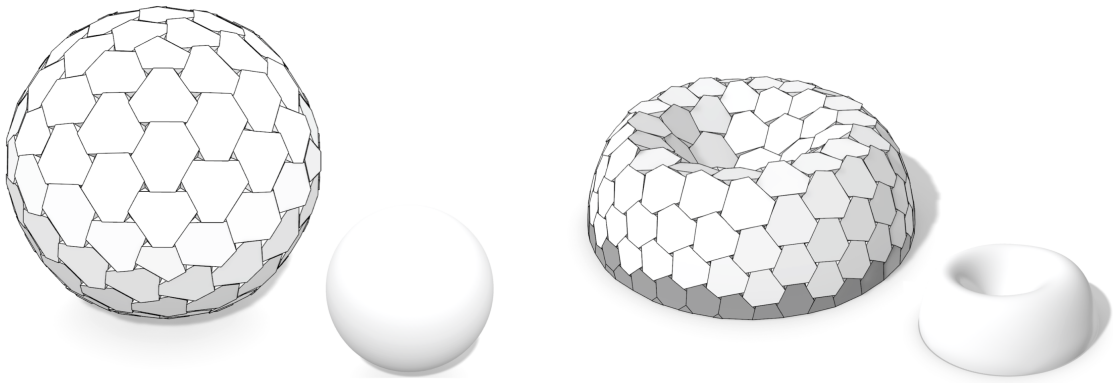


Figure 5.16 – Example shapes for which the optimization does not find an equilibrium under gravity.

sequence of the two structures. Once all blocks are assembled, we close the boundary frame by adding the key boundary component, which is connected to its counterpart using integrated magnets. Figure 5.1 and 5.14 show physical experiments, including the tilt analysis, conducted

on the two fabricated models, which validate their stability under different gravity directions. In particular, ROOF in Figure 5.1 is stable under arbitrary orientations as predicted by its global interlocking property. Please watch video https://youtu.be/EUm6IIdk2XM for demos.

5.5 Limitations and Future Work

Our work has several limitations that open up interesting directions for future research.

First, not all input designs are suitable for our method. For certain models, for example, closed surfaces or surfaces with a significant concave cavity, our method does not find an equilibrium under gravity; see Figure 5.16 for two examples. In addition, we make a number of idealized assumptions. We model the boundary frame as a single part yet break the frame into two subparts for fabrication and assembly, which might affect prediction accuracy of our computational method. We also assume rigidity and perfect accuracy of the assembly blocks. For practical applications, questions related to fabrication tolerances and their effect on the global assembly are important, as accumulation of fabrication errors could lead to structural failure of a supposedly stable assembly.

Our stability measure considers load-induced external forces acting on the center of gravity of each block. As a consequence, our optimization only aims for static equilibrium under all possible gravity directions yet not under all possible force and torque configurations. This also means that we currently do not directly optimize for globally interlocking assemblies (see also Figure 3.8). Formulating a stability measure and corresponding optimization to support all possible force and torque configurations would be an interesting future work.

Our structural optimization adapts the rotation angles $\{\alpha_{ij}\}$ of the augmented vectors while keeping the surface tessellation fixed. Extending the method to also optimize the tessellation is an interesting research challenge. Another possible extension is to also consider non-convex blocks or curved contact surfaces. Other interesting future work is on finding optimal assembly sequences, e.g. with respect to the required support structure, improving the computational performance of the optimization, or analyzing and optimizing structural stability under element failure. Finally, interesting theoretical questions emerge. For example, what is the class of surfaces for which a globally interlocking assembly with convex blocks is possible? We expect minimal surfaces to be in this class, but we do not yet have a proof of this conjecture.

5.6 Acknowledgments

We thank the reviewers for their valuable comments, Martin Kilian for providing surface models of BLOB, PENTAGON, and FREE HOLES, Daniele Panozzo for providing the SURFACE VOUGA model, and Julian Panetta, Mina Konaković-Luković for proofreading the paper. This work was supported by the Swiss National Science Foundation (NCCR Digital Fabrication

Agreement #51NF40-141853) and the SUTD Start-up Research Grant (Award Number: SRG ISTD 2019 148).

6 Modeling and Optimizing Cone-joints for Complex Assemblies

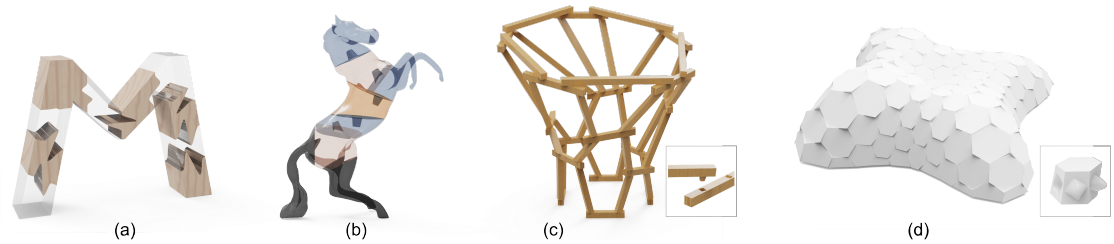


Figure 6.1 – Our computational framework optimizes cone joints for designing assemblable and stable structures with a variety of geometric forms: (a) planar, (b) volumetric, (c) frame, and (d) shell structures.

We present a computational framework for modeling and optimizing complex assemblies using cone joints. Cone joints are integral joints that generalize traditional single-direction joints such as mortise and tenon joints to support a general cone of directions for assembly. This additional motion flexibility not just reduces the risk of deadlocking for complex joint arrangements, but also simplifies the assembly process, in particular for automatic assembly by robots. On the other hand, compared to planar contacts, cone joints restrict relative part movement for improved structural stability. Cone joints can be realized in the form of curved contacts between associated parts, which have demonstrated good mechanical properties such as reduced stress concentration. To find the best trade-off between assemblability and stability, we propose an optimization approach that first determines the optimal motion cone for each part contact and subsequently derives a geometric realization of each joint to match this motion cone. We demonstrate that our approach can optimize cone joints for assemblies with a variety of geometric forms, and highlight several application examples.

6.1 Introduction

An assembly is a collection of parts that are deliberately arranged to have a specific functionality and/or form. The majority of man-made objects are designed as assemblies to accomplish a certain task (machines, vehicles), to make fabricating large objects feasible or cheaper (build-

ings, furniture), or simply to entertain (puzzles, toys). A necessary condition for an assembly to be practically used is structural stability. To this end, adjacent parts in an assembly have to be properly joined such that no unwanted relative part motions will happen under external forces.

Parts in an assembly are typically joined by glue, nails, screws, or some standard connectors. However, these joining methods do not encourage disassembly and re-assembly, and sometimes harm the external appearance of the assembly. With the advance of digital fabrication techniques, integral joints are more and more widely used for designing and making assemblies with intricate geometry. Integral joints are implicitly defined as the portion of each individual part that is in contact with adjacent parts. These joints can simplify the assembly process significantly as a sequence of operations to insert individual parts, without the need of installing external connectors with tools [Fairham, 2013].

Integral joints are typically designed in a way that two parts can be separated by translating one part along a single direction, e.g., mortise and tenon joints and dovetail joints. We call these joints *single-direction joints*; see Figure 6.2(a). Single-direction joints are widely used in furniture, timber structures, and 3D printed assemblies due to their strong capacity to strengthen structural stability. However, complex arrangements of single-direction joints could lead to deadlocking, making the assembly physically unrealizable. Moreover, these joints may complicate the assembly process as inserting a part precisely along a certain direction to fit the other could be a challenging task, especially in robotic assembly.

On the other end of the spectrum are integral joints with planar contacts (see Figure 6.2(b)), which are very common in unreinforced masonry structures [Whiting et al., 2009, Panozzo et al., 2013]. Advantages of *planar contacts* include simple geometry, ease of fabrication, avoidance of local stress concentration. Yet, these joints have the weakest capacity to restrict relative part motion. Incomplete assemblies with planar contacts usually require additional supports for being stable (i.e., in equilibrium) [Deuss et al., 2014].

In this chapter we study integral joints that generalize single-direction joints in terms of restricting relative part motion. We focus on making use of these joints for designing structures that are assemblable and stable. We call these joints *cone joints* since they allow one part to be separated/inserted relative to the other by translation along any direction within a motion cone. A cone joint is a single-direction joint if its translational motion cone contains a single direction. And a cone joint degenerates into a planar contact if its translational motion cone becomes a half sphere. Cone joints are typically realized in the form of curved or piecewise-planar contacts between two parts (see Figure 6.2(c&d)), which have been demonstrated to have good mechanical properties such as reduced stress concentration in building structurally stable assemblies [Dyskin et al., 2003b, Javan et al., 2016]. Parts with cone joints can be easily fabricated with 3D printing, CNC milling, and even hot-wire cutting for large-scale objects [Duenser et al., 2020].

Although cone joints have been successfully used in stable planar structures (e.g., brick-based

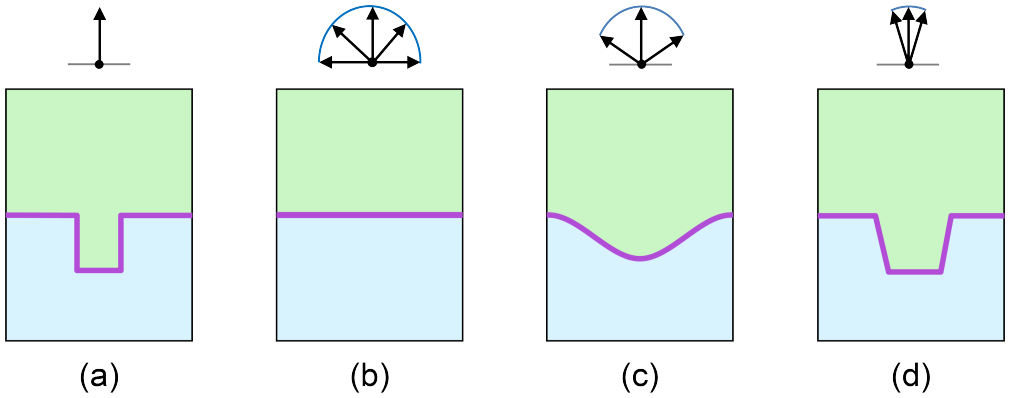


Figure 6.2 – Two parts joined in different ways using: (a) a single-direction joint, (a) a planar contact, and cone joints with (c) a curved or (d) a piecewise-planar contact (contacts are shown in purple). The four assemblies are all in equilibrium under gravity yet the translational motion space of taking out the green part varies (see cones on the top).

flooring systems [Weizmann et al., 2017]), assembly-based 3D printing [Araújo et al., 2019], and space-filling blocks [Akleman et al., 2020], little is known about how the variation in geometry of cone joints affects a structure's assemblability and stability, not to mention optimizing the geometry of cone joints for these two design goals. In this chapter, assemblability has a two-fold meaning: 1) parts can be assembled into the final structure without collision; and 2) each part can be inserted by translating along any direction within a sufficiently large circular cone, aiming to simplify the assembly process. Stability means a structure is in equilibrium under known external forces such as gravity. To address the challenge of modeling and optimizing complex assemblies with cone joints, we make the following technical contributions:

1. We establish a connection between the geometry of a joint and its motion space based on convexity theory. We show that the joint motion space is always convex and present a sampling-based approach to compute the motion space of curved-contact joints.

2. We present a motion-based method for static analysis of assemblies with cone joints, which is dual to existing force-based methods. The strength of this new method is to quantify structural stability and assemblability coherently in motion space.

3. We develop an optimization approach to construct cone joints for designing structures that are assemblable and stable, assuming the assembly sequence is given. Our framework iterates between a kinematic design stage that determines the required motion cone for each part contact and a geometric realization stage that finds the geometry of each joint to match this motion cone.

In this chapter, we model the geometry of cone joints using a simple parametric model to demonstrate the core functionality of our computational framework. Other parametric models can be easily integrated into our framework since they affect only the geometric realization

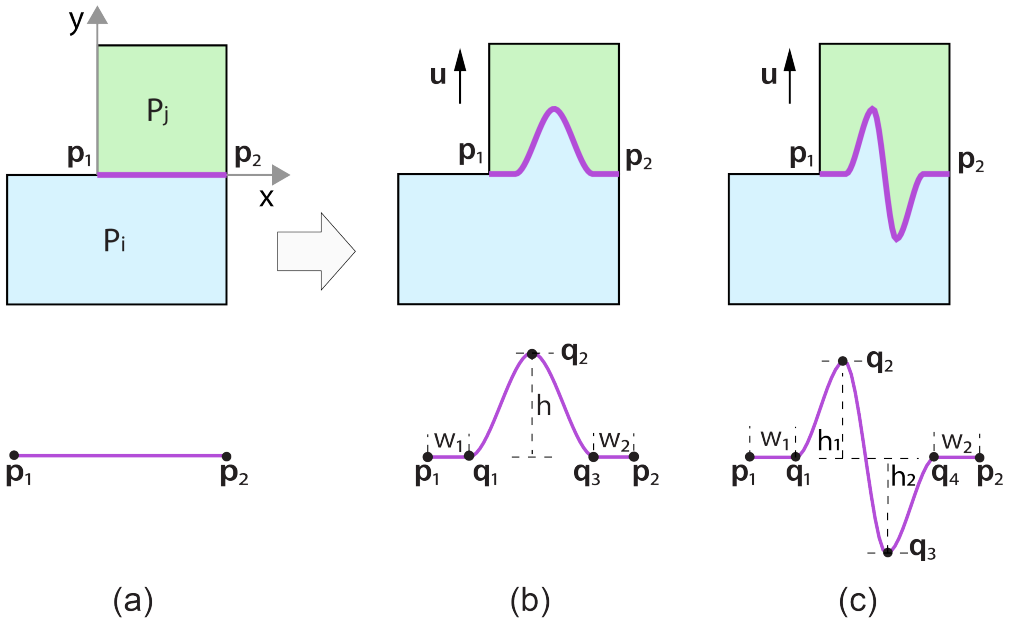


Figure 6.3 – Model cone joints, where the principal direction $\mathbf{u} = +y$. (a) An initial linear contact between two parts P_i and P_j; (b) an n-type cone joint; and (c) a z-type cone joint. The curved contact of each cone joint is modeled as a cubic spline defined by a few parameters (e.g., w_1, h, and w_2 in (b)).

stage, but not the motion space computation, motion-based static analysis, and kinematic design stage.

6.2 Modeling Assemblies with Cone Joints

In this section, we first present a parametric model to represent the geometry of cone joints with curved contacts. Next, we introduce a method to model assemblies with cone joints.

6.2.1 Modeling Geometry of Cone Joints

We define a contact as a curve segment (in 2D) or surface patch (in 3D) that lies exactly on two adjacent parts in an assembly. Each contact is initialized as a linear segment (in 2D) or a planar surface (in 3D). Starting from each initial contact, a cone joint can be modeled by modifying the geometry of the contact (i.e., making it curved). We propose a parametric model to represent the geometry of these cone joints. To facilitate understanding, we illustrate our parametric model mainly on 2D examples, and briefly explain how it can be extended to 3D.

One joint between two parts Given two 2D parts, P_i and P_j, with an initial linear contact defined by two endpoints $\mathbf{p}_1$ and $\mathbf{p}_2$, a local coordinate system is defined in a way that the

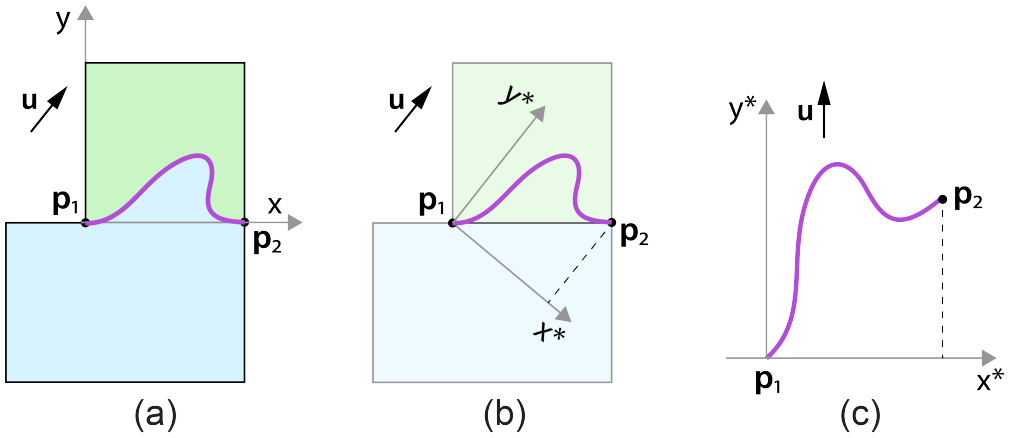


Figure 6.4 – Model cone joints when the principal direction $\mathbf{u}$ deviates from the y-axis. (a&b) Define a new local coordinate system such that $+\mathbf{y}^* = \mathbf{u}$. (c) Model the joint geometry in the new local coordinate system following the same method as $\mathbf{u} = +\mathbf{y}$.

origin is at $\mathbf{p}_1$ and the $+x$-axis is along vector $\mathbf{p}_2 - \mathbf{p}_1$; see Figure 6.3(a). We model the geometry of the cone joint between the two parts as a continuous curved contact. We parametrize the curved contact by using a cubic spline that interpolates the two endpoints $\mathbf{p}_1$ and $\mathbf{p}_2$ (see Figure 6.3(b&c)). According to the shape of the curved contact, joints can be classified into two classes:

1. *n-type joint* involves a concavity on one part and an extrusion on the other part, similar to the mortise-and-tenon joint. The joint shape is adjustable by three control points $\mathbf{q}_1$, $\mathbf{q}_2$, and $\mathbf{q}_3$ inserted between the two endpoints $\mathbf{p}_1$ and $\mathbf{p}_2$; see Figure 6.3(b).

2. *z-type joint* can be considered a combination of two n-type joints with opposite orientations. The joint shape is adjustable by four control points $\mathbf{q}_1$, $\mathbf{q}_2$, $\mathbf{q}_3$, and $\mathbf{q}_4$ inserted between the two endpoints $\mathbf{p}_1$ and $\mathbf{p}_2$; see Figure 6.3(c).

Compared with n-type joints, z-type joints generally have stronger capacity to restrict relative part motion, at the cost of more complex joint geometry and higher chance of stress concentration. In our experiments, we set bounds for the joint parameters to preserve their appearance and structural soundness.

To ensure that P_j can translate along a principal direction $\mathbf{u}$ while P_i is fixed, the contact curve has to be a height field along $\mathbf{u}$, taking the initial contact $\mathbf{p}_1\mathbf{p}_2$ as the base. Figure 6.3(b&c) shows example cone joints where P_j is removable along $\mathbf{u} = +\mathbf{y}$. When the principal direction $\mathbf{u}$ deviates from the y-axis, we need to define a new local coordinate system where the origin is still at $\mathbf{p}_1$ yet $+\mathbf{y}^* = \mathbf{u}$; see Figure 6.4(a&b). Next, we transform the endpoint $\mathbf{p}_1$ and $\mathbf{p}_2$ into this new coordinate system, and follow the same method as $\mathbf{u} = +\mathbf{y}$ to model the joint geometry as a cubic spline; see Figure 6.4(c).

Multiple joints between two parts Two initial parts may have more than one linear contact when the parts have non-convex shape (see Figure 6.5(a-c)). Ideally, we construct a cone joint for each linear contact. To this end, we assign a principal direction $\mathbf{u}^k$ to each contact C^k, and follow the above method to model the joint. To ensure that the two parts always can be separated, all assigned $\mathbf{u}^k$'s should have the same value; see Figure 6.5(d&e). Sometimes, some initial contact could be unsuitable for modeling a joint, e.g., when the contact is too small. In this case, we simply skip the joint modeling for that contact; see Figure 6.5(c&f) for an example.

Cone joints in 3D The parametric model for cone joints in 2D can be easily extended to 3D. In the following, we take a 3D n-type joint as an example to illustrate the extension, for which the initial contact is a 3D polygon and the principal direction is $\mathbf{u} = +z$. We model the 3D joint geometry as a parametric surface based on bivariate polynomials of degree 3 defined on a 3D rectangular region ($l_1 \times l_2$) within the initial contact. Hence, we first find a large rectangular region inside the initial contact (Figure 6.6(a)). Next, we divide the rectangular region into 9 rectangles with 16 interpolation points $\{\mathbf{p}_{ij}\}$ in total (Figure 6.6(b)). We assign the middle four points a height value h and the remaining points zero height, and apply bicubic interpolation to find the joint contact surface passing through these control points (Figure 6.6(c)). We show in Section 6.3 how our simple parametric model is well suited for optimization.

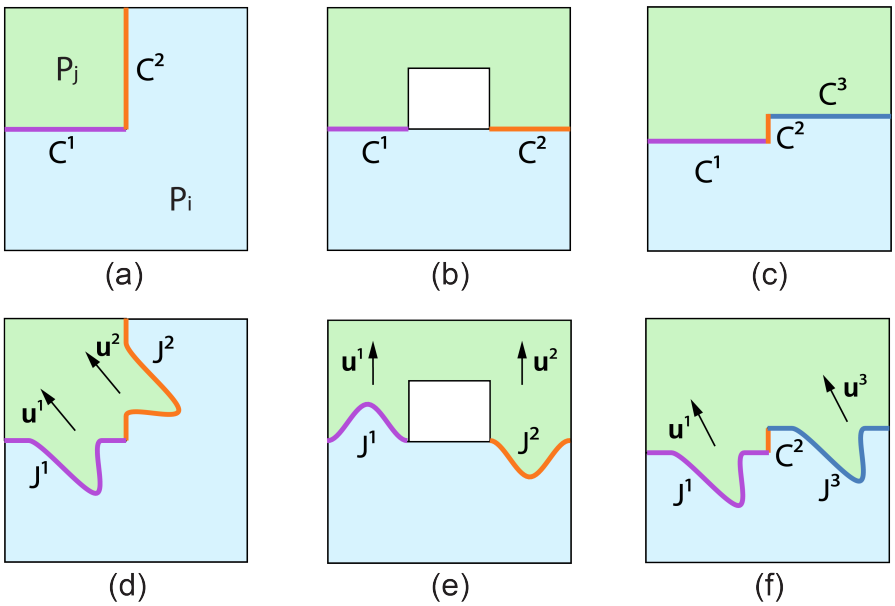


Figure 6.5 – Model cone joints when (a) P_i, (b) P_j, or (c) both P_i and P_j have non-convex shape, resulting multiple contacts between the two parts. (d&e) The principal directions of the two contacts should be the same to ensure P_1 and P_2 can be separated. (f) It is possible to skip joint modeling for some initial contacts that are too small, e.g., the orange one.

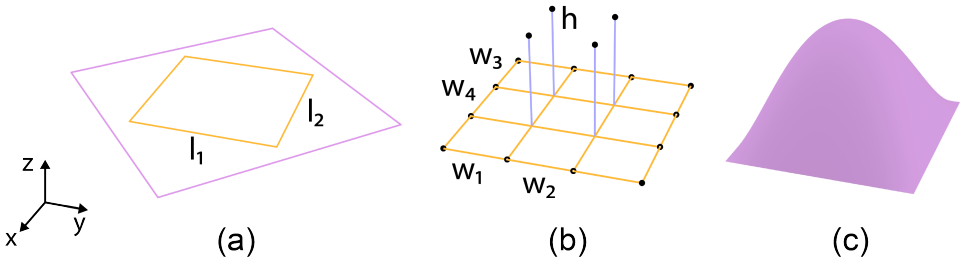


Figure 6.6 – Model n-type cone joints in 3D. (a) Identify a large rectangular region inside an initial 3D planar contact (i.e., the dashed polygon). (b) Compute sixteen control points, where h is the height of the four points in the middle. (c) Joint geometry modeled as a bicubic surface.

6.2.2 Modeling Assemblies with Cone Joints

In an assembly with $n \geq 2$ parts, we denote the parts as $\{P_i\}$, $1 \leq i \leq n$. We assume a user-specified assembly sequence, and name each part according to its assembly order (i.e., P_i is the ith assembled part). In the initial assembly with planar contacts, we model each part as a polyhedron. Faces in each part can be classified as *contact* and *non-contact* faces, according to whether the part is in contact with its neighbors through that face. Non-contact faces are allowed to be augmented with geometric features to enrich the assembly's appearance.

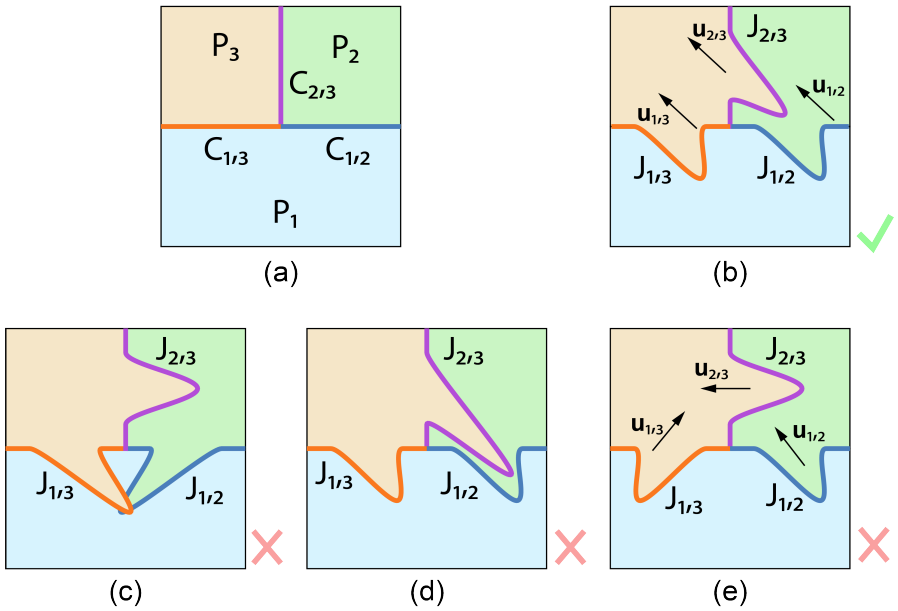


Figure 6.7 – Model cone joints in an assembly. (a) An initial assembly with three parts. (b) A modeled assembly that is disassemblable by taking out P_3 first. Examples of three cases that should be avoided: (c) cone joints intersect with one another; (d) cone joints are too close to one another, leading to structurally weak parts; and (e) motion cones of the joints are not well planned, leading to a deadlocking assembly.

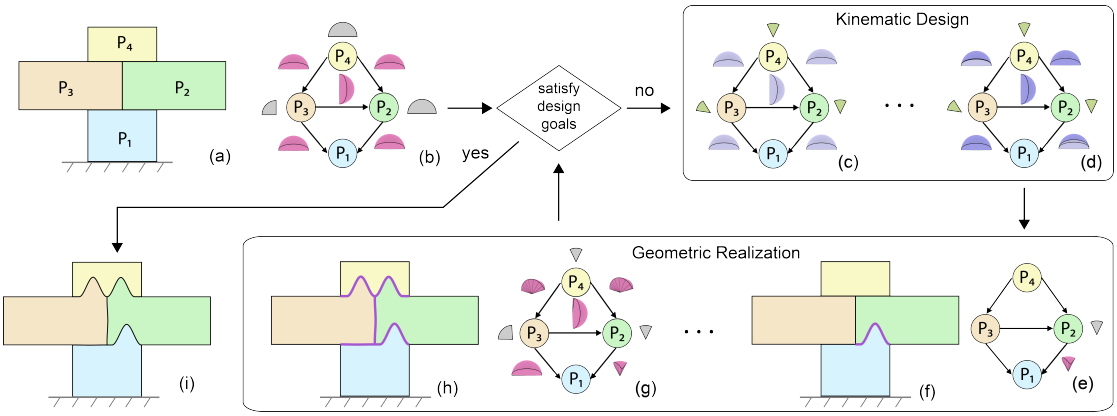


Figure 6.8 – Our computational design framework. (a) Our input is an initial assembly with planar contacts. (b) Motion-based representation of the initial assembly, where the joint motion cones (3D) and part motion cones (2D) are colored in magenta and gray respectively. (c&d) Kinematic design satisfies the two design goals by iteratively finding the required joint motion cones (colored in blue). (e-h) Geometric realization finds joints geometry independently to satisfy the corresponding joint motion cone. (i) The resulting assembly.

The initial planar contacts between each pair of adjacent parts, say P_i and P_j, are denoted as $\{C^k_{i,j}\}$. Each $\{C^k_{i,j}\}$ typically contains one contact, especially when initial parts are convex; see Figure 6.7(a) for an example. The cone joints modeled from the initial contacts $\{C^k_{i,j}\}$ are denoted as $\{J^k_{i,j}\}$. The geometry of each joint $J^k_{i,j}$ is parameterized by a small set of parameters $\Phi^k_{i,j}$, which include the joint $J^k_{i,j}$'s principal direction $\mathbf{u}^k_{i,j}$ (see Figure 6.7(b)) and the geometric parameters illustrated in Figure 6.3(b&c).

To ensure that parts (with cone joints) are fabricable and assemblable, several cases of joint configurations should be avoided by a careful selection of the joint parameters $\{\Phi^k_{i,j}\}$. First, intersection among the cone joints should be strictly avoided to ensure that the assembly is physically realizable. For example, the extrusion portions of $J_{1,2}$ and $J_{1,3}$ in P_1 should not intersect with each other like in Figure 6.7(c). Second, when a part has multiple cone joints, those cone joints should maintain a certain distance to one another to ensure structural soundness of the part. For example, part P_2 in Figure 6.7(d) is structurally weak since joints $J_{2,3}$ and $J_{1,2}$ are too close. Lastly, deadlocking should be avoided to ensure that parts can be physically assembled; compare Figure 6.7(b&e).

6.3 Designing Assemblies with Cone Joints

Taking an assembly with planar contacts as an input, our goal is to make it structurally stable and assemblable by constructing cone joints among the parts; see Figure 6.8(a&i). Here, structural stability means that the assembly is in equilibrium under known external forces $\{\mathbf{w}_i\}$ (e.g., gravity of each part) while assemblability means that each individual part can be inserted without colliding with assembled parts, by translation along any direction within

a sufficiently large cone. We model this required assemblability cone for each part P_i as a circular cone $K(\mathbf{d}_i, \alpha)$ for simplicity, where $\mathbf{d}_i$ is the cone's axis and α is the cone angle.

To obtain a structurally stable assembly, we prefer that each constructed cone joint can restrict the relative motion between the associated parts as much as possible. In the limit, each joint becomes a single-direction joint. However, to satisfy the goal of assemblability, single-direction joints should be strictly avoided and each constructed joint should have enough tolerance in the insertion direction of associated parts. Hence, our challenge is to find the geometry of cone joints that satisfy these two conflicting goals (i.e., stability and assemblability) simultaneously.

6.3.1 Overview of our approach

To address this problem, one straightforward approach is to directly search parameters that define the geometry of cone joints, e.g., by using a gradient-based method similar to [Whiting et al., 2012]. However, this approach has several limitations. First, the test of equilibrium could be very slow due to the dense sampling of cone joints; see Section 3.2. Second, the approach heavily relies on the initial geometry of the joints. Changing joint geometry significantly or even joint type requires restarting the whole search process.

Inspired by our motion-based equilibrium method in Section 3.3, we propose a new computational approach that is able to search the joint parameters efficiently and flexibly; see Figure 6.8. The key idea is to separate the search process into two stages: *kinematic design* and *geometric realization*, by introducing an intermediate motion-based representation of the assembly. In the kinematic design stage, we aim to satisfy the two design goals kinematically, by searching the motion cone required for each cone joint; see Figure 6.8(c&d). In the geometric realization stage, we compute for each joint suitable geometric parameters to satisfy the required motion cone; see Figure 6.8(e-h). The strength of our approach is that the kinematic design stage converts the two high-level design goals into a set of local requirements on the geometry of each individual joint. This allows us not just to search geometric parameters independently for each joint, but also to try as many initial joint parameters/types as possible to avoid local minima. Moreover, the kinematic design stage itself is very efficient since it purely works in motion space and focuses on the required motion cones with very few faces.

6.3.2 Kinematic Design

To facilitate discussion, we assume a single planar contact $C_{i,j}$ between each pair of adjacent parts P_i and P_j in an assembly, from which our approach will construct a cone joint $J_{i,j}$ by optimizing its parameters $\Phi_{i,j}$ defined in Section 6.2.2. The kinematic design stage aims at finding a required motion cone $\bar{\mathbf{V}}_{i,j}$ for each cone joint $J_{i,j}$ such that the two design goals can be satisfied kinematically. In the following, we first present a motion-based representation of a given assembly and then describe an optimization to search $\{\bar{\mathbf{V}}_{i,j}\}$ based on this representation.

Motion-based representation An assembly $\{P_i\}$ can be represented by a parts-graph, where each node represents a part and each edge represents the contact/joint between the two associated parts. The known part assembly order can be easily encoded in the parts-graph by adding a direction for each edge. More precisely, each directed edge from P_j to P_i indicates that P_i will be assembled before P_j (i.e., $i < j$ according to our notation); see Figure 6.8(b).

Our motion-based representation augments this directed parts-graph with two pieces of information. We associate to each edge from P_j to P_i a motion cone $\mathbf{V}_{i,j}$ allowed by the contact $C_{i,j}$ or joint $J_{i,j}$. The geometry of the motion cone $\mathbf{V}_{i,j}$ is defined by all possible infinitesimal rigid motions of P_j to separate it from a fixed P_i; see again Figure 3.4(a&c). For each node P_j, we define a motion cone $\mathbf{V}_j$ of all possible infinitesimal translational motion to take out P_j from the partial assembly $\{P_1, ..., P_{j-1}\}$. Denote the indices of P_j's adjacent parts in the given assembly as $A(j)$. The motion cone $\mathbf{V}_j$ for taking out P_j can be represented as:

$$\mathbf{V}_j = \bigcap_{i<j,\ i\in A(j)} T(\mathbf{V}_{i,j}) \tag{6.1}$$

where $T(\cdot)$ is an operator that converts a rigid motion cone into a translational motion cone by ignoring the rotational component. This is because we assume translational motion for assembling the parts due to its simplicity of execution. $\mathbf{V}_j$ is called the part motion cone and used for checking (dis)assemblability while $\mathbf{V}_{i,j}$ is called the joint motion cone and used for static analysis.

Given the motion-based representation, we are able to easily test whether the two design goals are satisfied. For equilibrium, we run the optimization in Equation 3.17, in which each joint motion cone $\mathbf{V}_{i,j}$ is defined in a global coordinate system associated with the assembly rather than the joint local coordinate system. To verify assemblability, we search whether each part motion cone $\mathbf{V}_j$ contains a circular cone $K(-\mathbf{d}_j, \alpha)$, where the cone angle α is a constant and the cone direction $\mathbf{d}_j$ is unknown. Note that the circular cone $K(-\mathbf{d}_j, \alpha)$ is a disassemblability cone, which is actually a reflection of the assemblability cone $K(\mathbf{d}_j, \alpha)$ in the motion space.

Search for required joint motion cones $\{\bar{\mathbf{V}}_{i,j}\}$ We model each required joint motion cone $\bar{\mathbf{V}}_{i,j}$ as a polyhedral cone with a small number of faces to speed up the static analysis process. In particular, the conic section of each required joint motion cone $\bar{\mathbf{V}}_{i,j}$ is a rectangle (4 faces) for 2D joints and 5D cuboid (10 faces) for 3D joints, defined in the joint local coordinate system; see Figure 6.9(top). When transformed to the global coordinate system, these conic sections will not be rectangle/cuboid any more; see Figure 6.9(bottom). We parameterize the geometry of each motion cone $\bar{\mathbf{V}}_{i,j}$ in 2D (3D) with a few variables denoted as $\Psi_{i,j} = \{\psi_{i,j}\}$. Based on the required joint motion cones $\{\bar{\mathbf{V}}_{i,j}\}$, we compute the corresponding part motion cones $\{\bar{\mathbf{V}}_j\}$ according to Equation 6.1.

Given the required joint motion cones $\{\bar{\mathbf{V}}_{i,j}\}$ as well as the known external forces $\{\mathbf{w}_i\}$, we compute the assembly's infeasibility measure for equilibrium by solving Equation 3.17. Based

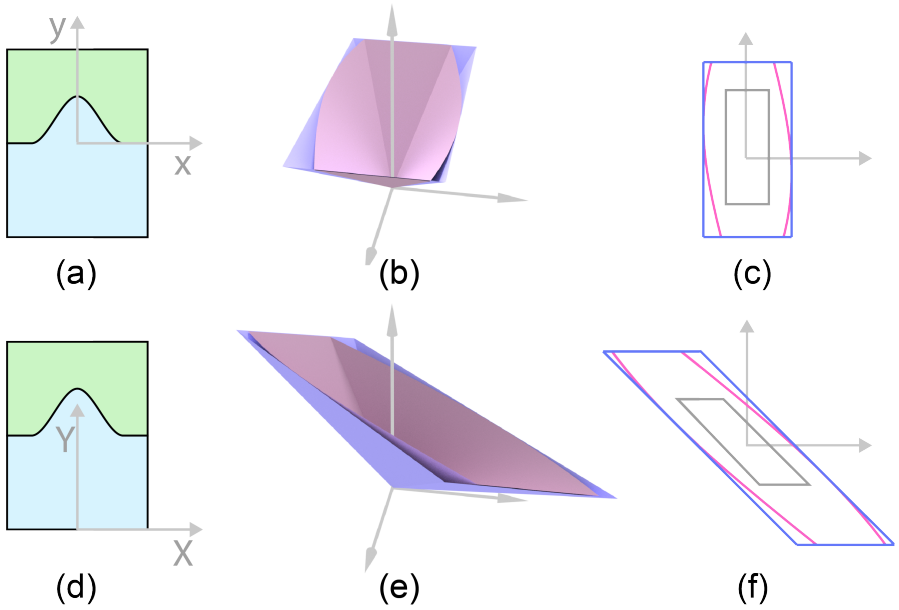


Figure 6.9 – (a&d) Given a joint, (b&e) its actual and required joint motion cones as well as (c&f) their conic sections are colored in magenta and blue, respectively. When changing from (top) the joint local coordinate system to (bottom) the assembly global coordinate system, the motion cones change accordingly. An example bound of the required joint motion cone is colored in grey in (c&f), for the joint's principal direction $\mathbf{u} = +y$.

on our modeling, each required joint motion cone $\bar{\mathbf{V}}_{i,j}$ can be converted back to 4 (10) linear constraints in our infeasibility energy optimization for each 2D (3D) joint. With very few constraints, the optimization can be performed efficiently. We also satisfy the goal of assemblability by requiring the following constraint for each part motion cone $\bar{\mathbf{V}}_j$:

$$K(-\mathbf{d}_j, \alpha) \subseteq \bar{\mathbf{V}}_j \tag{6.2}$$

To this end, the kinematic design can be formulated as a problem to search for the required joint motion cones $\{\bar{\mathbf{V}}_{i,j}\}$ by solving the following optimization:

$$\begin{aligned} \min_{\Psi_{i,j}, \mathbf{d}_j} \quad & E(\mathbf{w}, \{\bar{\mathbf{V}}_{i,j}\}) \\ \text{s.t.} \quad & K(-\mathbf{d}_j, \alpha) \subseteq \bar{V}_j, \end{aligned} \tag{6.3}$$

where E is the motion-based infeasibility measure in Equation 3.17. In practice, we avoid too small motion cones for the cone joints such that the motion cones can be realized by joint geometry later. Hence, we set a bound for the parameters $\Psi_{i,j}$ of each required joint motion cone $\bar{\mathbf{V}}_{i,j}$; see Figure 6.9(c&f). The above optimization can be solved by using an off-the-shelf interior-point method. Please refer to the supplementary material for details (Appendix C).

Once we have obtained the required joint motion cones $\{\bar{\mathbf{V}}_{i,j}\}$, we should require $\mathbf{V}_{i,j} \subseteq \bar{\mathbf{V}}_{i,j}$,

where $\mathbf{V}_{i,j}$ is the actual joint motion cone, to ensure the resulting assembly is in equilibrium. To satisfy the goal of assemblability, we should require $\hat{K}(-\mathbf{d}_j,\alpha) \subseteq \mathbf{V}_j \subseteq \mathbf{V}_{i,j}$, where $\hat{K}(-\mathbf{d}_j,\alpha) = K(-\mathbf{d}_j,\alpha) \times \{\mathbf{0}\}^{2m-3}$, $m = 2,3$ is the circular cone $K(\mathbf{d}_j,\alpha)$ of a mD joint represented in a higher dimensional space (i.e., represent translational motion cone in the rigid motion space). In summary, the kinematic design stage converts the two design goals into a set of local requirements for each cone joint $J_{i,j}$ to be constructed at the geometric realization stage:

$$\hat{K}(-\mathbf{d}_j,\alpha) \subseteq \mathbf{V}_{i,j} \subseteq \bar{\mathbf{V}}_{i,j}, \quad \text{for each } J_{i,j} \tag{6.4}$$

6.3.3 Geometric Realization

Geometric realization aims to construct the geometry of each cone joint to satisfy the constraint on the joint motion cone $\mathbf{V}_{i,j}$ in Equation 6.4. According to Theorem 2, $\mathbf{V}_{i,j}$ can be approximated by sampling the joint's generalized normal space $\tilde{\mathbf{N}}_{i,j} = \{\mathbf{n}_l(\Phi_{i,j})\}$ and computing the dual cone, i.e., $\mathbf{V}_{i,j} \approx \tilde{\mathbf{N}}^*_{i,j}$.

To achieve our goal, we could optimize the joint shape (i.e., search the joint design parameters $\Phi_{i,j}$) such that the joint motion cone $\mathbf{V}_{i,j}$ satisfies the constraint in Equation 6.4. However, computing the joint motion cone is already very time-consuming, let alone doing shape optimization for it. A better way is to perform the shape optimization in the generalized normal space $\tilde{\mathbf{N}}_{i,j}$, which is directly controlled by the joint parameters $\Phi_{i,j}$. Lemma 1 provides a strategy to transform the constraints from the motion space (see Equation 6.4) to the generalized normal space:

$$\bar{\mathbf{V}}^*_{i,j} \subseteq \texttt{cone}(\mathbf{N}_{i,j}) \subseteq \hat{K}(-\mathbf{d}_j,\alpha)^* \tag{6.5}$$

in which the direction of the subset symbol "$\subseteq$" is reversed due to the dual operator.

In particular, the dual of the required joint motion cone $\bar{\mathbf{V}}_{i,j}$, a polyhedral cone, is just the minimum convex cone envelope of the polyhedral cone's face normals $\{\mathbf{f}_k(\Psi_{i,j})\}$:

$$\bar{\mathbf{V}}^*_{i,j} = \texttt{cone}(\{\mathbf{f}_k(\Psi_{i,j})\}) \tag{6.6}$$

And the dual of the circular cone $\hat{K}(-\mathbf{d}_j,\alpha)$ is still a circular cone, just with a cone angle $\frac{\pi}{2}-\alpha$:

$$\begin{aligned} \hat{K}(-\mathbf{d}_j,\alpha)^* &= (K(-\mathbf{d}_j,\alpha) \times \{\mathbf{0}\}^{2m-3})^* \\ &= K(-\mathbf{d}_j, \frac{\pi}{2}-\alpha) \times \mathbb{R}^{2m-3} \end{aligned} \tag{6.7}$$

We illustrate the relation between the dual cones using their conic sections for 2D assemblies in Figure 6.10. The required disassemblability cone is an infinitely long strip, and the required joint motion cone is the minimum convex cone envelope of face normals $\{\mathbf{f}_k\}$. The minimum convex cone envelope of $\mathbf{N}_{i,j}$ has to stay in between the conic sections of these two required cones. To this end, we formulate our joint shape optimization as an energy minimization

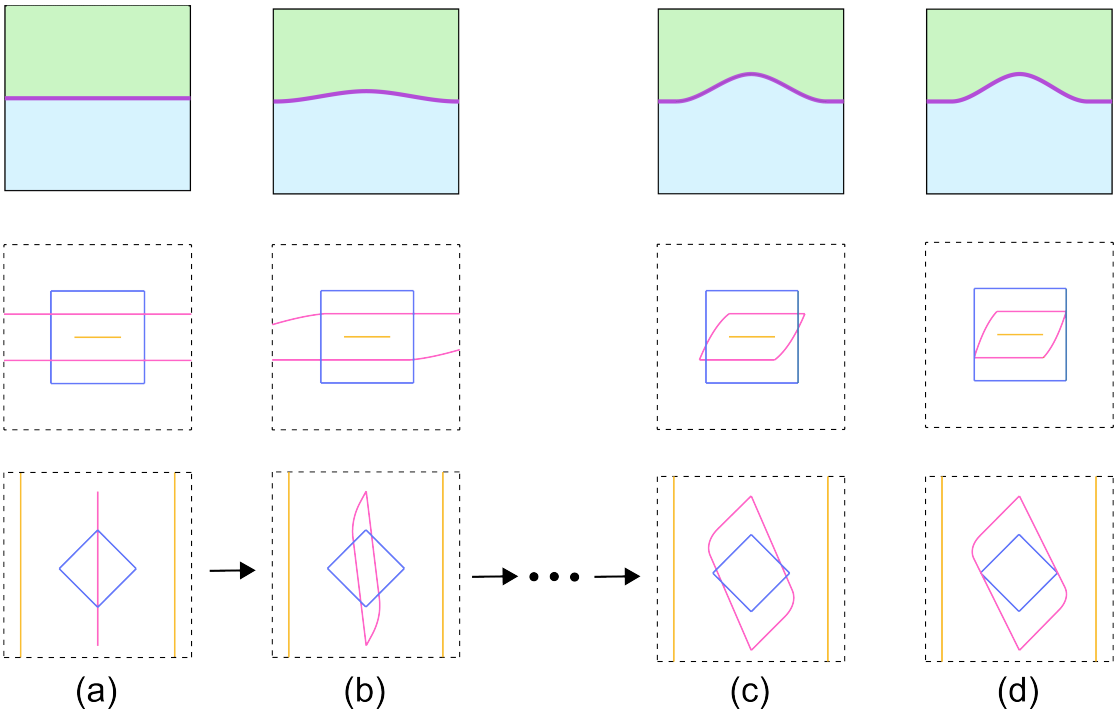


Figure 6.10 – Geometric realization process for a single joint. From left to right, initial planar contact, intermediate joints, and resulting joint. From top to bottom, joints represented in geometric space, motion space, and generalized normal space, respectively. The conic sections of the required joint motion cones are shown in blue, the required disassemblability cones in orange, and the actual joint motion cones in magenta.

problem:

$$\begin{aligned} \min_{\Phi_{i,j}} \quad & \sum_{l,k} \text{dist}(\texttt{cone}(\{\mathbf{n}_l\}), \mathbf{f}_k) \\ \text{s.t.} \quad & \mathbf{n}_l \in K(-\mathbf{d}_j, \frac{\pi}{2} - \alpha) \times \mathbb{R}^{2m-3} \end{aligned} \tag{6.8}$$

where $\text{dist}(\texttt{cone}(\{\mathbf{n}_l\}), \mathbf{f}_k)$ is the distance between the minimum convex cone envelope $\texttt{cone}(\{\mathbf{n}_l\})$ and the face normal $\mathbf{f}_k$. In particular, $\text{dist}(\texttt{cone}(\{\mathbf{n}_l\}), \mathbf{f}_k) = 0$ when the face normal $\mathbf{f}_k$ is inside the $\texttt{cone}(\{\mathbf{n}_l\})$. We compute the distance using quadratic programming:

$$\text{dist}(\texttt{cone}(\{\mathbf{n}_l\}), \mathbf{f}_k) = \min_{\lambda_l \geq 0} \quad \|\mathbf{f}_k - \sum_l \lambda_l \mathbf{n}_l\|^2 \tag{6.9}$$

One additional constraint is to maintain a certain distance between each joint $J_{i,j}$ and the boundary of parts P_i and P_j to ensure validity and structural soundness of these two parts; see again Figure 6.7(c&d). To this end, we combine the shape of parts P_i and P_j, and compute a signed distance function to the shape boundary; see Figure 6.11(b&d). The constraint is satisfied by requiring the distance value of each point on the joint $J_{i,j}$ to be larger than a threshold; see Figure 6.11.

The initial values of joint parameters $\Phi_{i,j}$ can profoundly influence the result. Fortunately, the

geometric realization is performed independently among the joints (see again Figure 6.11), and each joint optimization problem only has a few variables. Hence, for each joint, we try as many initial values as possible to avoid local minima by uniformly sampling the variables. Similar to the optimization at the kinematic design stage, the joint optimization at the geometric realization stage can be solved by an off-the-shelf interior-point method.

In practice, we need to iterate between the kinematic design and geometric realization stages several times. This is because each part's centroid and weight (i.e., volume) are assumed to be fixed at the kinematic design stage, yet are subject to change after joints are constructed in the geometric realization stage. To alleviate this issue, our approach prefers not to have dramatic changes on the joint geometry at each iteration. Occasionally, the geometric realization stage may not be able to find any joint geometry that satisfies the required motion cone computed at the kinematic design stage. To this end, updating the lower bound for the required joint motion cones according to the current joint geometry (i.e., the new bound is the same as the joint motion cone) would facilitate the search of joint geometry in the next iteration.

6.4 Results

We implemented our tool in C++ and libigl [Jacobson et al., 2018], and employed Knitro [Artelys, 2020] for solving our optimizations. We conducted all experiments on a Linux workstation

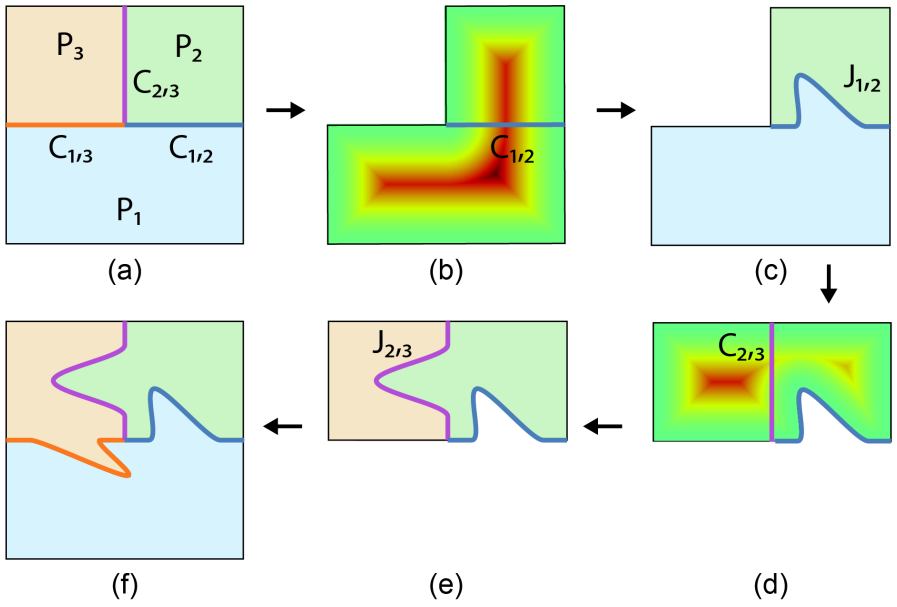


Figure 6.11 – Geometric realization process on a 3-part assembly. Starting from (a) an initial assembly, we (b) compute a distance function for the boundary of parts P_1 and P_2 to (c) construct joint $J_{1,2}$. Next, we (d) compute a distance function for the boundary of parts P_2 and P_3 to (e) construct joint $J_{2,3}$. (f) The resulting assembly. In (b&d) the distance functions, green and red indicate small and large distance values, respectively.

with an AMD Ryzen Threadripper 3990X 64-Core Processor and 128GB of RAM. We show that our approach can handle assemblies with a variety of geometric forms, including planar, volumetric, frame, and shell structures; see Figure 6.1. Thanks to the flexibility offered by our cone joints, our approach can compute assemblies that are stable to different degrees, e.g., *equilibrium* under gravity, stable under *lateral* forces, and single-key *interlocking*. Our approach can also consider stability not just for the final assembly but also for intermediate stages of the assembly process. By this, we can generate *support-free* structures that can be assembled without using any support.

Statistics Table 6.1 summarizes the statistics of all the results presented in this chapter. The third to seventh columns show if the result is 2D or 3D, the total number of parts, the total number of joints, angle α of the required assemblability cone, one of the four features mentioned above, and optimization time, respectively. The angle α is typically set as 5 degrees, which is sufficient for inserting parts easily. We set a larger α for making the cone joints less sharp (Figures 6.12(right) and 6.15) and a smaller α for making the assembly more stable (Figure 6.13(c)). We can also not consider assemblability as a design goal by not setting any value for angle α (Figure 6.14). Our approach typically takes less than 1 minute to generate 2D results yet may take hours for 3D results due to the larger number of joints as well as more variables to define the geometry of each joint.

Table 6.1 – Statistics of the results shown in this chapter.

Fig	Result	2D/3D	# Part	# Joint	α	Feature	Optim. Time (min)
1	M	2D	10	9	5	Equilibrium	1.00
	Horse	3D	13	9	5	Equilibrium	9.82
	Pavilion	3D	48	62	5	Equilibrium	2.19
	Lilium Tower	3D	139	325	5	Equilibrium	203.78
15(rig)	Scarecrow	2D	4	5	25	Equilibrium	0.08
16(c)	Sphere	3D	6	12	3	Interlocking	2.98
17	Stack	2D	10	10	-	Equilibrium	0.04
18	Tree	2D	7	9	10	Equilibrium	0.37
19	Leaning Tower (lef)	2D	14	28	5	Equilibrium	0.23
	Leaning Tower (mid)	2D	14	28	5	Equilibrium	0.45
	Leaning Tower (rig)	2D	13	27	5	Equilibrium	0.38
20	Leaning Tower	2D	14	28	5	Support-free	0.68
21	Deer	2D	14	23	5	Support-free	0.70
22	Igloo	3D	139	326	5	Laterally stable	61.11

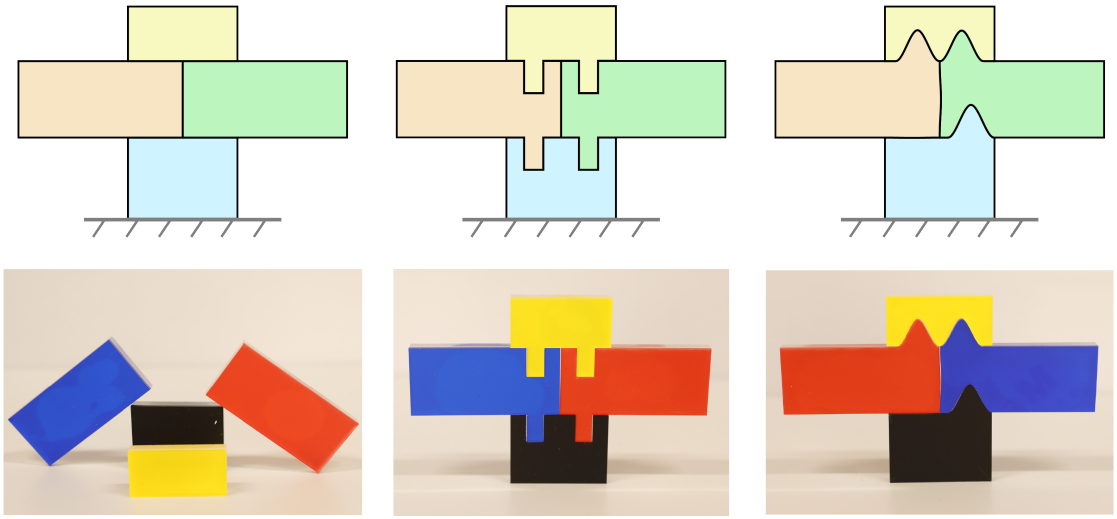


Figure 6.12 – Comparison of stability and assemblability of 4-part SCARECROWs with (left) planar contacts, (middle) single-direction joints, and (right) our optimized cone joints.

Evaluation of cone joints We compare our optimized cone joints with planar contacts and single-direction joints by designing three 2D puzzles; see Figure 6.12. To verify stability and assemblability of these puzzles, we fabricate them with laser-cutting. Our experiment shows that the puzzle with planar contacts is not in equilibrium under gravity; see Figure 6.12(left). To assemble the puzzle with single-direction joints, we need to align each part with the partially completed puzzle carefully before it can be successfully inserted; see Figure 6.12(middle). Compared with these two puzzles, our puzzle with cone joints avoids the two issues by achieving a good balance between stability and assemblability; see Figure 6.12(right). Please watch the accompanying video for demos.

We compare our cone joints with planar contacts and standard mortise-and-tenon joints by designing four 6-part SPHERES. The sphere with planar contacts cannot be in equilibrium under gravity, and the sphere with standard mortise-and-tenon joints are deadlocking; see

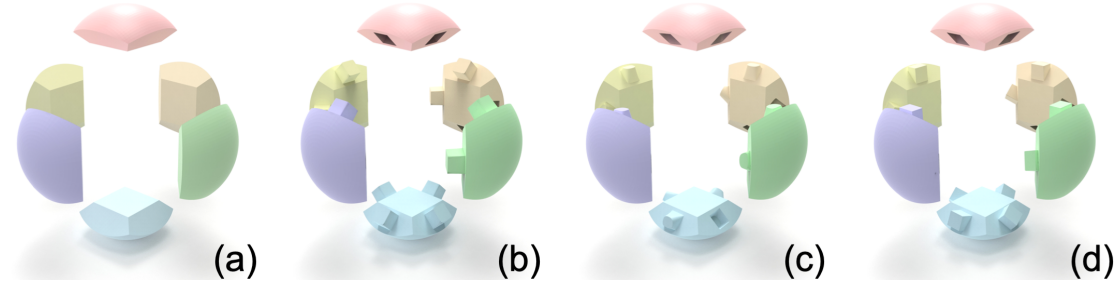


Figure 6.13 – Comparison of stability and assemblability of 6-part SPHERES with (a) planar contacts, (b) standard mortise-and-tenon joints, (c) our optimized cone joints, and (d) tilted mortise-and-tenon joints derived from our result. The assembly in (a) is not stable while the one in (b) is deadlocking.

Figure 6.13(a&b). We show that our approach can make the sphere single-key interlocking by constructing cone joints; see Figure 6.13(c). This is achieved by applying an external force configuration on the assembly, where at each contact the external force tries to push the two associated parts and separate them. The assembly is single-key interlocking if it passes our equilibrium test under this external force configuration, assuming the key is held by other means; see the supplementary material (Appendix C for a proof. Our design result can be easily modified to derive some variants, e.g., to facilitate fabrication. For example, we can replace each curved-contact joint with a tilted mortise-and-tenon joint based on the principal direction **u** of the joint; see Figures 6.13(d) and 6.4. The resulting assembly is guaranteed to be interlocking since a tilted mortise-and-tenon joint has a smaller motion cone than any curved-contact joint.

Evaluation of our optimization To show the benefits of our design approach, we compare it with a baseline approach that directly searches parameters of cone joints using a gradient-based method, similar to [Whiting et al., 2012]. Both approaches take the same assembly with planar contacts as input (Figure 6.14(a)), and generate curved-contact joints to make the assembly be in equilibrium (Figure 6.14(b&c)). Figure 6.14(d) shows the computation time of the two approaches with respect to the number of parts. When the number of parts is small, the computation time of the two approaches are comparable. However, the computation time of the baseline increases dramatically when the number of parts is more than 16 since the computation cost becomes dominated by the equilibrium test, which is expensive for assemblies with many curved-contact joints. Thanks to the kinematic design stage, our approach avoids executing the expensive equilibrium test frequently whenever changing joints geometry. During the experiment, we find that the baseline approach sometimes can find better solutions than ours, especially when the number of parts is small. For example, when there are only two parts where the bottom one is fixed, the baseline creates a cone joint that makes the assembly in equilibrium yet our approach fails to do so; see Figure 6.14(e). This is because our kinematic design stage assumes the parts' centroids are fixed, which actually may change after joints have been constructed in the geometric realization stage.

Equilibrium puzzles We have used our approach to generate 2D/3D equilibrium puzzles. Figure 6.1(a&b) shows two puzzles, M and HORSE, which cannot be in equilibrium if we simply use planar contacts. This is because planar contacts cannot prevent the motion (e.g., sliding) of some parts caused by gravity such as the two parts at the middle of M and the head part of HORSE. Figures 6.15 shows a puzzle TREE as well as the fabricated result. Our designed cone joints not only make the puzzle stable under gravity, but also provide a hint to find puzzle pieces that should match with one another.

Given input assemblies that are far from an equilibrium state, our approach can still make it stable. To demonstrate this, we create a sequence of input assemblies, where parts have the same shape (except bottom ones) but have been tilted for a certain angle; see Figure 6.16(left)

for an example input. Due to the parts arrangement, the larger the tilt angle is, the further the assembly is from an equilibrium state. We construct cone joints for each input assembly to make it be in equilibrium using our approach; see Figure 6.16 for three example results. We observe that our optimized cone joints become more and more sharp when the tilt angle increases since a sharp joint corresponds to a small motion cone. A drawback of these sharp cone joints is that they may fail due to stress concentration. To alleviate this issue, additional constraints on the joint shape can be enforced during the geometric realization stage.

Our approach can be extended to design support-free puzzles; see Figures 6.17 and 6.18. These puzzles are in equilibrium for each intermediate assembly state, and thus can be assembled

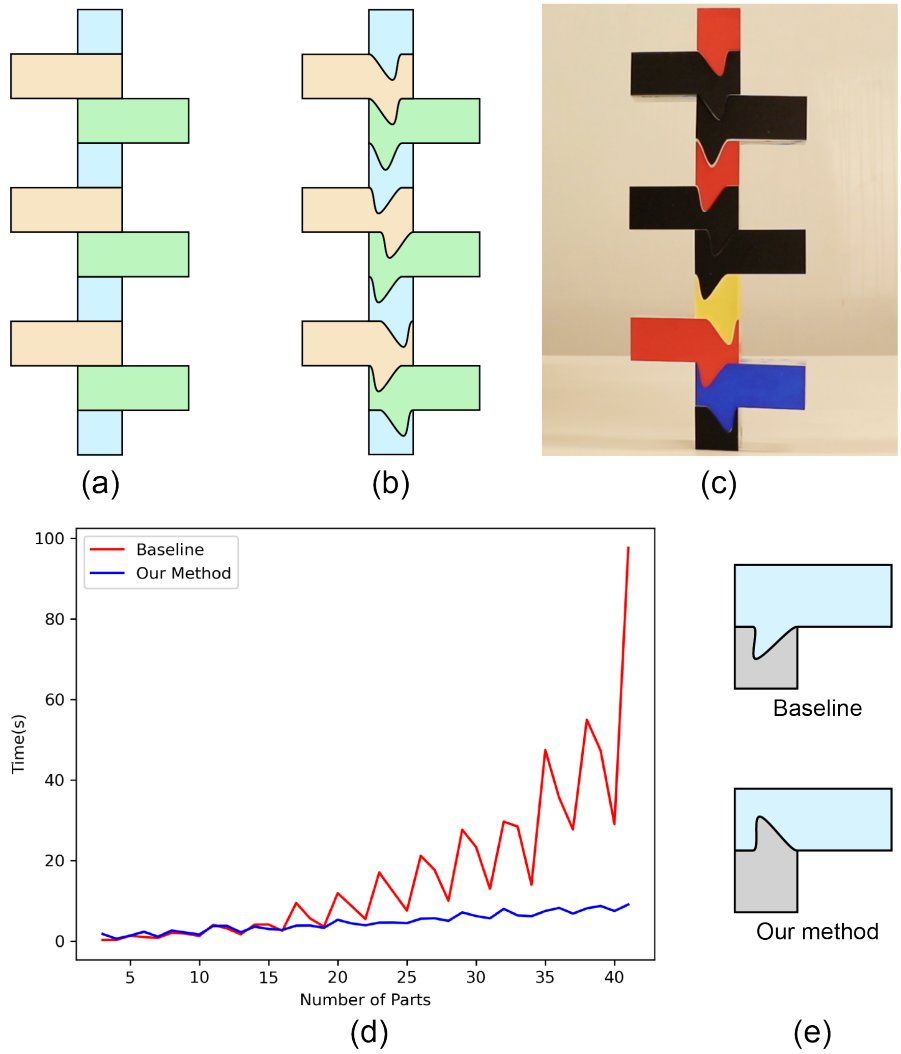


Figure 6.14 – Comparison of our design approach with a baseline approach: (a) an example input assembly; (b&c) a result generated by our approach; (d) computation time of the two approaches with respect to the number of parts; and (e) an example case where the baseline performs better.

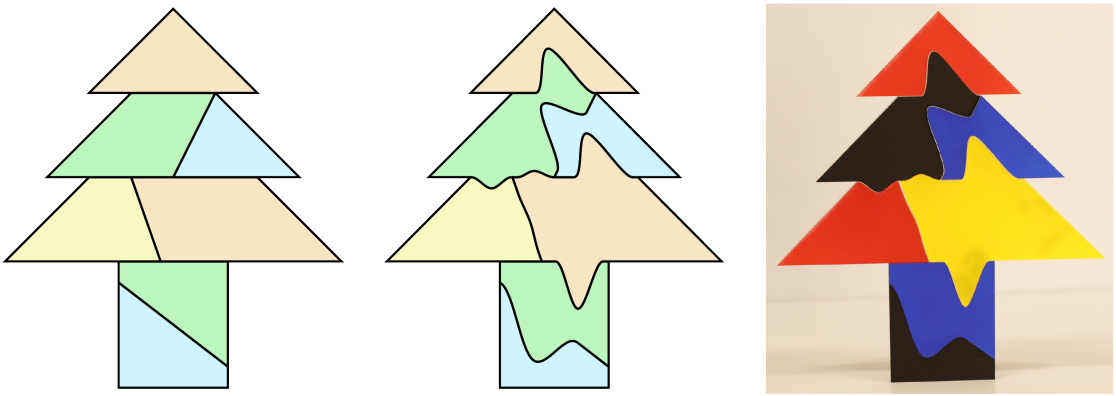


Figure 6.15 – Equilibrium puzzle TREE generated by our approach. From left to right: input assembly, our result, and laser-cut puzzle.

without using any support. To generate these puzzles, we assume a predefined assembly sequence, and take all the intermediate assemblies and the final assembly as input of our kinematic design. We obtain the required joint motion cones by summing the infeasibility energy in Equation 6.3 for all the input assemblies and solving the optimization. The geometric realization is performed without any change.

Shell structures Shell structures with planar contacts can be used as masonries in architecture. These structures are *self-supporting* if they can be in equilibrium under gravity [Panozzo et al., 2013]. Shell structures with an inverted bump on the top usually cannot be self-supporting; see LILIUM TOWER in Figure 6.1(d). Taking such a structure as input, our approach can make it self-supporting by creating cone joints for some of the planar contacts. Our approach only modifies a small subset of the planar contacts since our gradient-based optimization changes contact geometry only if it can reduce the infeasibility energy in Equa-

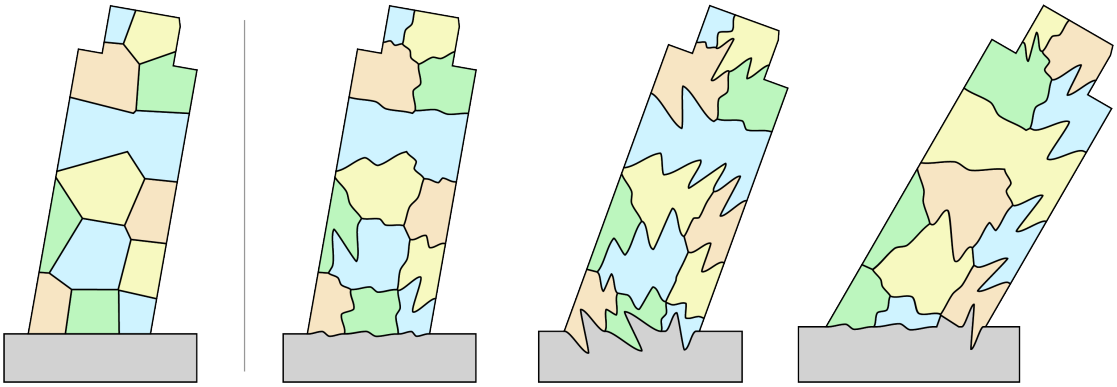


Figure 6.16 – Tilt experiment on (left) an input LEANING TOWER to verify the ability of our designed cone joints to make an assembly stable. From left to right: the tilt angles of the three results are 10, 20, and 30 degrees, respectively.

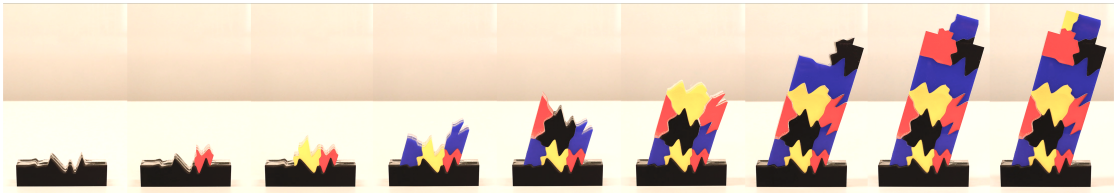


Figure 6.17 – Support-free equilibrium puzzle LEANING TOWER with 15 degrees tilt angle (see also Figure 6.16).

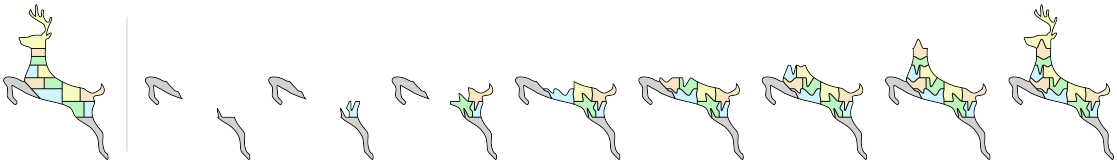


Figure 6.18 – Support-free equilibrium puzzle DEER. The input assembly is shown on the left and parts that are fixed during the assembly process are colored in gray.

Figure 6.19 – An IGLOO shell with lateral stability designed by our approach.

tion 6.3.

In some applications (e.g., architecture), a shell structure should be in equilibrium under not just gravity but also lateral forces (e.g., from wind). This lateral stability can be evaluated by simulating a tilt analysis experiment, where the ground plane of the structure is tilted to apply a lateral force to the structure caused by gravity. The lateral stability is measured by the largest angle of tilting the ground plane without collapse of the structure [Wang et al., 2019]. Our approach can generate shell structures with lateral stability by a slight change on the kinematic design stage. In detail, we sum the infeasibility energy E in Equation 6.3 for multiple external force configurations (i.e., gravity, and lateral forces from different directions). The assembly is considered as laterally stable if it is in equilibrium under any of the external force configurations. Figure 6.19 shows a shell structure IGLOO that cannot be in equilibrium under gravity if only planar contacts are used. By using our cone joints, we show that the structure can be tilted for at most 35 degrees.

Frame structures Frame structures composed of rod-like parts are widely used in architecture and furniture. In these structures, each part contacts others at its two ends and each contact area is usually quite small. Our approach can optimize cone joints to make these structures stable. The resulting cone joints look similar to single-direction joints; see PAVILION in Figure 6.1(c). This is perhaps because these structures with planar contacts are far away from an equilibrium state, and thus joints with a small motion cone are needed to make them stable.

6.5 Limitations and Future Work

Our work has several limitations that open up interesting directions for future research. First, the geometry of 2D/3D cone joints that can be supported by our current parametric model is quite limited. Exploring more complex parametric models and studying their impact on the motion cones would be an interesting future work. Second, we plan to include friction in our motion-based equilibrium method as well as the design approach. Third, our geometric realization stage focuses on generating geometry of cone joints to match optimized motion cones. Taking other aspects such as appearance and structural soundness of joints into consideration would make the designed cone joints more practical for use. Lastly, we approximate the required motion cones in the kinematic design stage as a pyramid with 4 (10) faces for 2D (3D) joints to speed up the static analysis. In the future, we may try more accurate approximations to see if they are helpful to improve the design performance.

6.6 Acknowledgments

We thank the reviewers for their valuable comments, Tian Chen for fabricating the physical models, Héloïse Dupont de Dinechin and Mingyang Li for implementing some experiment code. This research was supported by the NCCR Digital Fabrication, funded by the Swiss National Science Foundation (NCCR Digital Fabrication Agreement #51NF40-141853), the TL@SUTD Seed Research Project Grant (Number: RTDS S 1907 011), and the SUTD Start-up Research Grant (Number: SRG ISTD 2019 148).

7 Conclusion & Discussion

7.1 Summary

When solving complicated problems, humans tend to decompose them into small sub-problems and tackle them separately. The assembly perfectly exploits this divide-and-conquer strategy making them so ubiquitous in our society. However, most assembly design problems have multiple objectives. Some objectives, such as structural stability and assemblability, though essential for assemblies to be practical, can distract designers from other important factors like aesthetics and functionality. We formulate the assembly design problems as multi-objective optimization and propose interactive design tools that benefit both experts and DIY enthusiasts.

This thesis has demonstrated several computational methods for analyzing and designing structurally stable assemblies with rigid parts. We have shown that kinematics is the key element to establish the connection between parts' geometry and structural stability measurement. With the help of the joint motion cones, we can directly measure the structural stability of assemblies in the kinematic domain without knowing detailed geometric features. The separation of geometric and kinematic design stages makes our design framework capable of handling complex inputs and adaptable to various applications.

In Chapter 3, we summarize the structural stability analysis for rigid body assemblies. We review the forced-based stability measurement and establish an equivalent kinematic-based stability measurement using the duality between statics and kinematics. We show that the non-penetration constraints in our new stability measurement are the key components to construct the based directional blocking graphs and the linear inequality system for testing global interlocking. We prove that globally interlocking assemblies are the most stable assemblies which can withstand arbitrary forces and torques. For assemblies with lateral stability, we measure their stability by using tilting analysis.

In Chapter 4, we present *DESIA*, a general framework for designing interlocking assemblies. The foundation is the conceptual representation of interlocking assemblies using a set of

directed graph called the base DBGs that encode blocking relations among the parts along with a finite number of directions. Guided by these graphs, DESIA constructs interlocking assemblies iteratively using a two-stage scheme, where the first stage specifies desired blocking relations among the parts while the second stage realizes them in the embedded geometry. Rather than focusing on each specific kind of assembly as previous works, DESIA supports designing interlocking assemblies of various forms and allows generating interlocking results according to users' desires. Compared with previous works, DESIA significantly enhances the flexibility of designing interlocking assemblies by exploring a significantly enlarged search space. The achievements of DESIA are evidenced by the various results presented in the thesis that cannot be designed without using DESIA.

In Chapter 5, we studied how to approximate freeform design surfaces with structurally stable assemblies of convex rigid pieces. When analyzing their interlocking behavior, we observed that simultaneous multi-part motions and part rotations need to be considered when formulating a general algorithm to test for global interlocking. Our new formulation then utilizes the link between the geometric property of global interlocking and force-based equilibrium conditions defined in Section 3 . This provided the key insight to formulate a general stability metric that is suitable for optimization. Our experiments show how this optimization allows the creation of structurally stable assemblies of convex elements that can approximate a wide variety of double-curved freeform surfaces.

In Chapter 6, we show that assemblability and stability are two necessary conditions for using assemblies in the physical world. However, they could be in conflict with each other when restricting relative part motion with joints. Finding a trade-off between these two conditions is a challenging task for conventional approaches based on planar contacts or single-direction joints. We propose cone joints to address this challenge, which interpolate between planar contacts and single-direction joints in terms of capacity to restrict relative part motion. We quantify this capacity as a motion cone and present an approach to optimize cone joints for designing structures that are assemblable and stable. We found the separation into kinematic and geometric stages essential to make the optimization computationally tractable. We show the versatility of our approach by designing a variety of 2D/3D assemblies that are in equilibrium under gravity, stable under lateral forces, interlocking, or support-free for the assembly process.

7.2 Future Work

Motivated by the advance in technologies of digital fabrication, automatic construction, and material science, future research effort could be devoted to designing and fabricating advanced assemblies that are not accessible before such as architectural structures that can be constructed automatically by using robotic assembly [Eversmann et al., 2017], modular robots that can be reconfigured autonomously or manually to form different assemblies [Brunete et al., 2017], and self-assembling systems based on shape memory material [Sun et al., 2021].

To design these advanced assemblies, significant challenges remain in this field and should be addressed in future research:

Analysis with high prediction accuracy. To simplify the computational analysis of assemblies, a common assumption is that the fabrication material is rigid and manufacturing precision is perfect. Yet in practice, materials are not infinitely rigid and fabrication devices have limited manufacturing precision. As a consequence, structural issues can arise when the relative motion of two parts is restricted by a small contact or a thin joint. Internal stress concentrations at the contact or joint can lead to material failure. Similarly, a geometrically stable design, e.g., a furniture assembly, could become instable when fabricated, due to gaps among the parts caused by machining tolerances. To improve prediction accuracy, one possible way is to take these imperfections into consideration in the analysis methods. For example, tolerance analysis [Chen et al., 2014] can provide insights into the structural performance of an assembly. Such analysis tasks are particularly challenging as imperfections of each individual part will propagate within the whole assembly.

Another challenge is functionality analysis of assemblies, which is critical for designing assemblies that function as expected by the user. Functionality analysis of given shapes is a relatively new topic in computer graphics [Hu et al., 2018], and developing methods for analyzing functionalities of assemblies could be a promising research direction.

Design for multiple objectives. This report has discussed a number of objectives for designing assemblies. To date, however, most publications address and solve a specific objective and do not consider all aspects relevant for an assembly. For example, for large-scale assemblies in architecture, combining the objectives of parts fabricability, structural stability of the final and intermediate assemblies, and automated assembly process planning, would provide a more complete framework for design. Future research effort could address such design problems with multiple objectives, which might require a fundamentally new formulation of the problem, a combination of different representations of assemblies, and new strategies to search over a rather constrained solution space. Another challenge is to find a good trade-off among multiple objectives to satisfy users' high-level preference.

Advanced design methods can be studied by exploring several aspects of assemblies in a novel way, in particular, assembly plans. Designing assemblies with (dis)assembly plans that are beyond sequential, monotone, and linear is a promising research topic, as these complex plans are possible to be precisely realized by robots nowadays. Such (dis)assembly plans could be useful in terms of entertaining people (e.g., disassembly puzzles) or enhancing structural stability (e.g., complex coordinated motion to take out parts).

Machine learning for searching the design space. Machine learning methods offer potential benefits for solving non-linear and non-convex problems. In recent years, tremendous research efforts have been devoted to solve traditional graphics problems such as shape synthesis and mesh segmentation using machine learning methods [Mitra et al., 2019]. However, developing learning methods to solve design and fabrication problems is challenging due to

various hard constraints (e.g., fabricability) and/or global constraints (e.g., assemblability) involved in these problems. Another challenge is the lack of datasets to train the learning algorithms, as the design problem could be too specific or there is no existing way to generate desired designs. One possible way to address this challenge is to use reinforcement learning, which has been shown to be good at adaptively sampling the search space. We see a high potential for reinforcement learning to help solve challenging problems of designing assemblies.

A Supplementary Material for *Interlocking Assemblies*

This supplementary material presents comparisons with [Song et al., 2012] and [Fu et al., 2015] and more details about results generated by our approach (Section 4.3), and provides a proof of the statement that an assembly whose parts-graph has a cut point cannot be interlocking (Section A.2), as we as a proof of the statement that our DBG-based approach is sufficient to test interlocking for 3D assemblies with orthogonal part connections (Section A.3).

A.1 Comparisons and Results

Comparison with [Song et al. 2012]. Given the same voxelized model, designing an N-piece interlocking puzzle becomes significantly harder when we increase N. This is because the fewer number of voxels each piece has, the more difficult is to construct blocking relations among the pieces. Thanks to our framework that allows to achieve interlocking with fewer blocking relations, we can create interlocking puzzles with a much larger N than [Song et al., 2012]. Figure A.1 shows several interlocking CUBEs designed by [Song et al., 2012] and our approach, from which we can see that the more voxels the input model has, the larger difference is on N between our approach and [Song et al., 2012].

We fabricate the 9-part $4 \times 4 \times 4$ CUBE with Lego bricks to validate its steadiness; see Figure A.2(top) and the video https://youtu.be/Pqwo3GbxBEo. Besides solid objects, our approach works also on hollowed objects; see the 7-part $4 \times 4 \times 4$ HOLLOWED CUBE in Figure A.2(bottom).

Compared with [Song et al., 2012], our approach allows imposing additional constraint when designing interlocking assemblies, e.g., avoid cutting seams on geometric features of the final assembly. This ability is demonstrated on the 14-part CARTOON DOG shown in Figure A.4, which avoids cutting seams on its ears, nose and tail (i.e., P_{10}, P_{13}, P_9, and P_2). In particular, the two eyes of the model is actually from the same part (i.e., P_{12}).

Comparison with [Fu et al. 2015]. Fu et al. [Fu et al., 2015] design interlocking furniture by breaking all furniture parts into a set of small groups and achieving global interlocking by

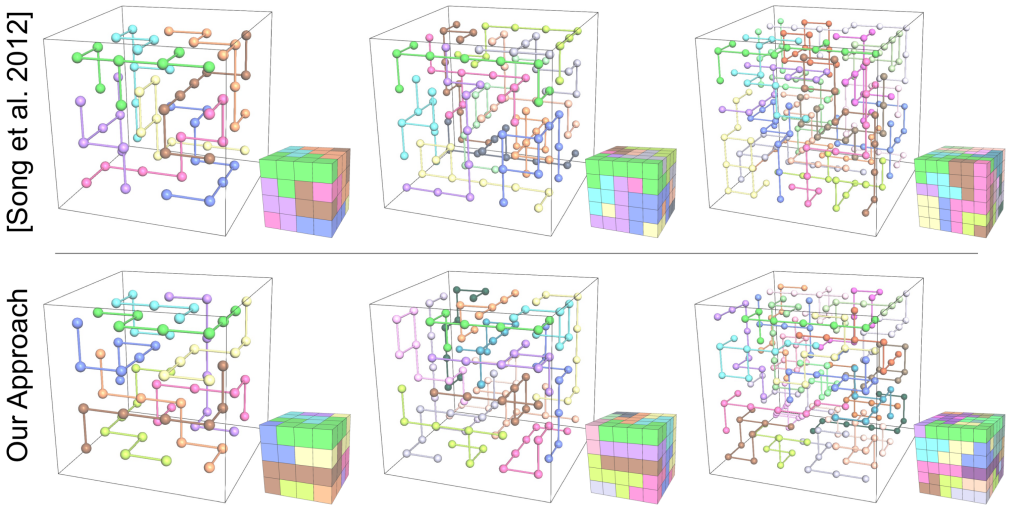


Figure A.1 – Interlocking Cubes created by [Song et al., 2012] (top) and our approach (bottom). From left to right: the Cubes are $4 \times 4 \times 4$ (8 vs 9 parts), $5 \times 5 \times 5$ (13 vs 16 parts), and $6 \times 6 \times 6$ (20 vs 27 parts).

making each group interlocking (i.e., LIG) and enforcing dependency across the LIGs; see Figure A.3(top). Here, a LIG is a connected sub-graph (i.e., 3- or 4-part cycles) in the parts-graph, such that only a specific part (local key) is mobile in the LIG. The dependency between LIGs, say G_j depending on G_i, means that G_i immobilizes G_j if the key of G_j is a non-key

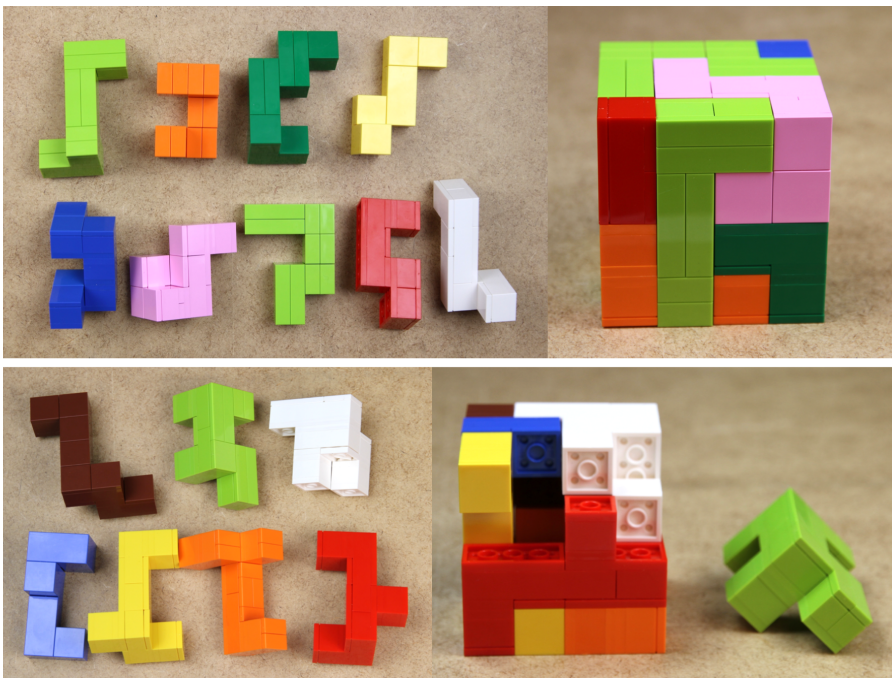


Figure A.2 – (Top) 9-part CUBE and (bottom) 7-part HOLLOWED CUBE designed by our approach and made with Lego bricks.

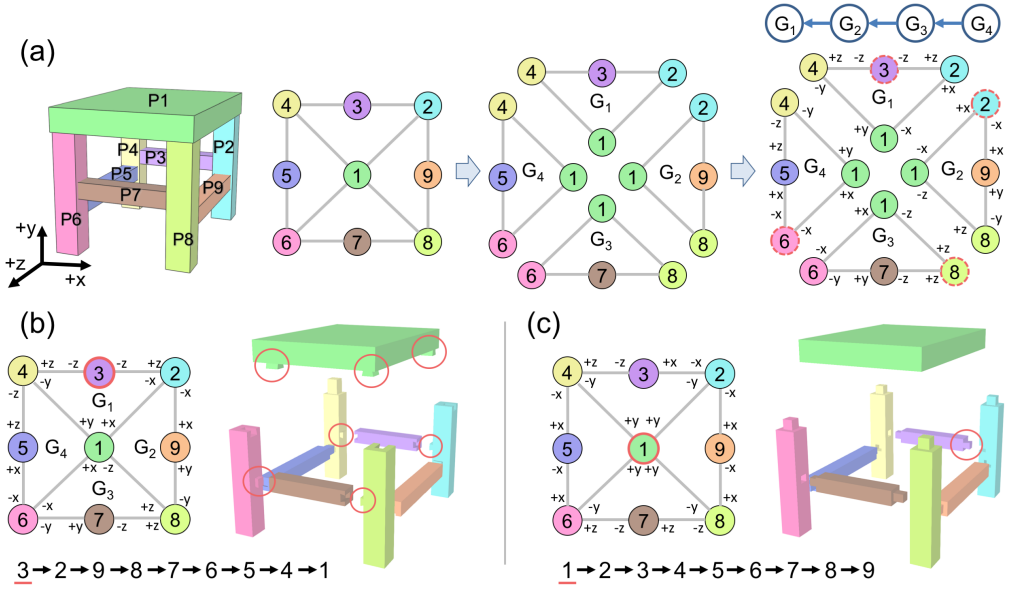


Figure A.3 – (a) The LIG-based approach of [Fu et al., 2015] to design a 9-piece interlocking Bench, where the LIGs are denoted as G_i and the local key in each LIG is highlighted with a dashed circle. Interlocking results generated by [Fu et al., 2015] (b) and our approach (c), where the part disassembly sequence is shown at the bottom and the number of dovetail joints are 10 and 1 respectively (see examples in red circles). Note that dovetail joint is generally weaker than mortise-and-tenon due to smaller contact surface between the joint and the part body.

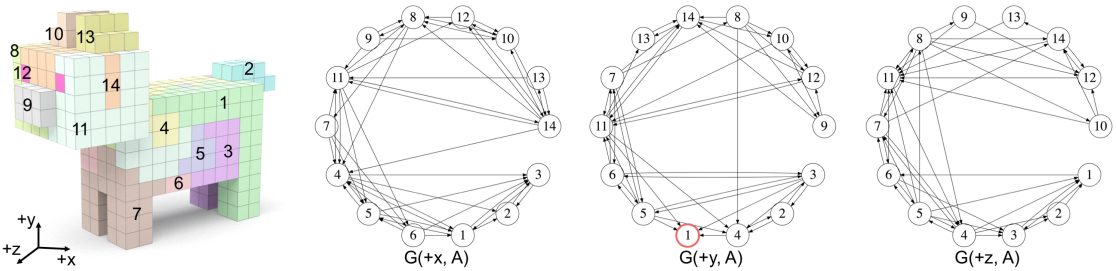


Figure A.4 – 14-part CARTOON DOG and its base DBGs. The key part P_1 (in green) is movable along $+y$; see the red circle in $G(+y, A)$.

part of G_i. By successively forming LIGs in a parts-graph with carefully-planned joints, a dependency tree can be built over all the LIGs (see Figure A.3(top right)), thus interlocking all the parts in the entire assembly.

Although this LIG-based approach can generate promising results, it restricts the generation of interlocking configurations by constraining the part construction order; i.e., successive parts need to be from the same LIG or neighboring LIGs in the dependency tree (see Figure A.3(b)). In sharp contrast, our approach has much more flexibility on selecting P_i for construction, as long as parts in R_i remain connected in the parts-graph. For example, our approach can

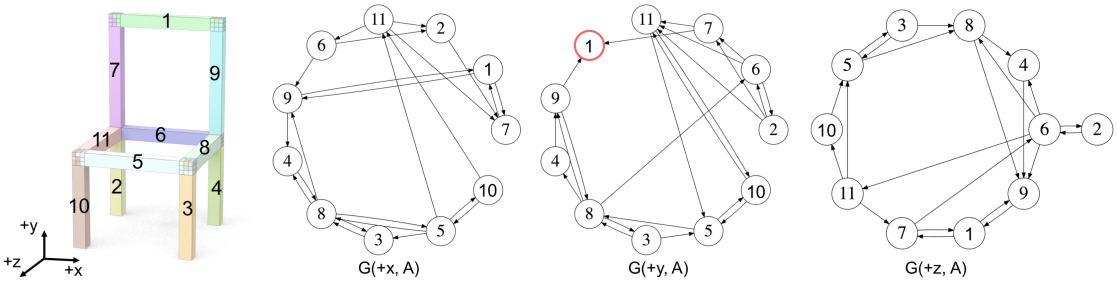


Figure A.5 – 11-part FRAME CHAIR and its base DBGs. The key part P_1 (in green) is movable along $+y$; see the red circle in $G(+y, A)$.

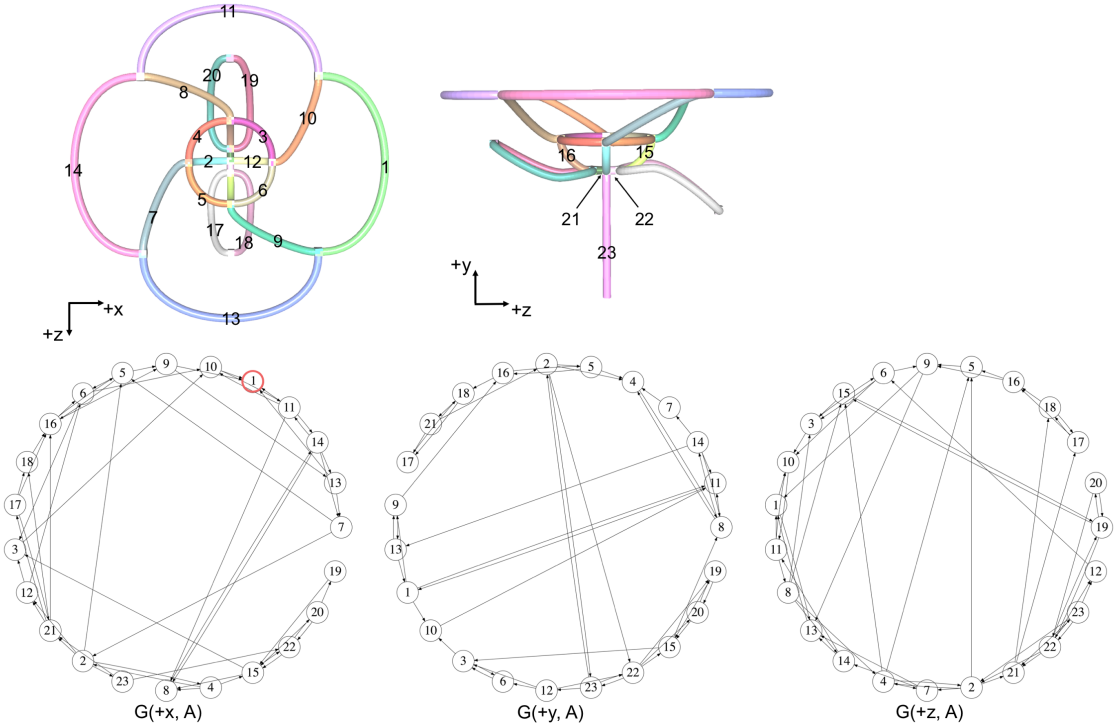


Figure A.6 – 23-part FLOWER and its base DBGs. The key part P_1 (in green) is movable along $+x$; see the red circle in $G(+x, A)$.

generate a result with P_1 as the key (see Figure A.3(c)), which cannot be achieved by [Fu et al., 2015]. This is because after taking out P_1, there is no more 3- or 4-part cycle for [Fu et al., 2015] to construct LIGs. This advantage enables us to explore a larger space of interlocking joint configurations and generate results that satisfy additional requirement(s), e.g., an interlocking solution with more structurally strong joints such as mortise-and-tenon; compare the joints in Figure A.3(b&c).

Moreover, this LIG-based approach focuses on furniture with 3- or 4-part cyclic substructures

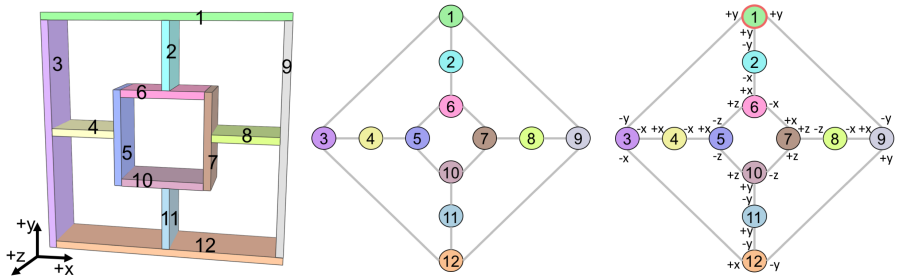


Figure A.7 – From left to right, a BOOKSHELF, its parts-graph, and an interlocking joint configuration generated by our approach.

only since they require these substructures to make LIGs. Thus, this approach does not suitable for furniture that has very few such substructures; e.g., the BOOKSHELF shown in Figure A.7(left) has only two 4-part substructures (i.e., $[P_6, P_7, P_{10}, P_5]$ and $[P_1, P_9, P_{12}, P_3]$) but four 6-part substructures (e.g., $[P_1, P_2, P_6, P_5, P_4, P_3]$). In contrast, our approach does not have such a limitation and can generate an interlocking joint configuration; see Figure A.7(right).

Interlocking Frame Structures. Figure A.5 shows the 11-part FRAME CHAIR designed by our approach. The interlocking property of this result can be easily verified since the three base DBGs are strongly connected except the key. This validates the efficiency of our interlocking testing approach. Figure A.6 shows the 23-part FLOWER designed by our approach, along with its three base DBGs.

A.2 Proof of Statement on the Parts-graph

To prove the statement about the requirement on an assembly's parts-graph to make it interlocking, we first present a lemma and then prove the statement by contradiction.

Lemma. For any single-key interlocking assembly $\mathbf{A}_n$, there always exists a disassembly sequence, say $P_1 \rightarrow P_2 \rightarrow \cdots \rightarrow P_n$, such that $\{P_1, .., P_k\}$ and $\{P_{k+1}, .., P_n\}$ are both connected in the parts-graph G respectively.

Statement. If the parts-graph of an assembly $\mathbf{A}_n$ has a cut point, then $\mathbf{A}_n$ cannot be interlocking, no matter what kinds of joints are constructed between the parts in $\mathbf{A}_n$.

Proof. Suppose $\mathbf{A}_n$ is an interlocking assembly and $P_1 \rightarrow P_2 \rightarrow \cdots \rightarrow P_n$ is a sequence to disassemble $\mathbf{A}_n$ that satisfies the condition in the lemma.

Case 1: Cut point is P_1. $\{P_1, P_2, \ldots, P_n\}$ must be connected in G according to the lemma, which contradicts with the condition that P_1 is a cut point.

Case 2: Cut point is P_n. $\{P_1, P_2, \ldots, P_{n-1}\}$ must be connected in G according to the lemma, which contradicts with the condition that P_n is a cut point.

Case 3: Cut point is $P_k (1 < k < n)$. P_k and $P_j, (j > k)$ must be in the same strongly connected component in all the DBGs. Thus, simple paths from P_k to P_j and P_j to P_k exist in all DBGs. These paths could not go through $\{P_1, P_2, \ldots, P_{k-1}\}$ in any DBG since P_k is a cut point of the parts-graph G. Therefore, in any DBG, there exists an in-edge from P_l $(l > k)$ to P_k and an out-edge from P_k to P_m $(m > k)$. This makes P_k immobilized in all base directions in $[P_k, \ldots, P_n]$, and thus P_k should not be the first part to be taken out in $[P_k, \ldots, P_n]$. This leads to a contradiction with the assumption on the disassembly order.

By summarizing above three cases, we prove that the parts-graph of an interlocking assembly cannot have a cut point.

A.3 Proof of Statement on the DBG-based Test

To prove the statement that the DBG-based approach is sufficient for testing interlocking of 3D assemblies where parts are orthogonally connected, we only need to prove the following statement:

Statement. For 3D assemblies with orthogonal part connections, if there exists a part group in which parts can translate along different directions simultaneously, there must exist a subset of the group in which part(s) can translate along the same direction simultaneously.

Proof. Since parts are orthogonally connected, the contact normal n_{ij} from P_i to P_j must be one of $\{\pm e_x, \pm e_y, \pm e_z\}$. Denote the velocity of part P_i as $v_i = (v_i^x, v_i^y, v_i^z)$ and assume $n_{ij} = e_x = (1,0,0)$ without loss of generality. The inequality $(v_j - v_i) \cdot n_{ij} \geq 0$, which is used to test interlocking in Section 3.2, is simplified to:

$$v_j^x - v_i^x \geq 0 \tag{A.1}$$

By summarizing this simplified constraint for every $n_{ij} = \pm e_x$, we have the following system of linear inequalities:

$$A_X X \geq 0 \tag{A.2}$$

where X is a vector of x-component of part velocities, and A_X is the matrix specifying the coefficients given by the normals of all interfaces that are $\pm e_x$.

Similarly, we can have another two systems of linear inequalities:

$$\begin{aligned} A_Y Y &\geq 0 \\ A_Z Z &\geq 0 \end{aligned} \tag{A.3}$$

Therefore, the original system of linear inequalities $AV \geq 0$ can be separated into three inde-

pendent linear systems:

$$\begin{aligned} A_X X &\geq 0 \\ A_Y Y &\geq 0 \\ A_Z Z &\geq 0 \end{aligned} \tag{A.4}$$

where $V = [X^T, Y^T, Z^T]^T$.

For any valid solution $V = [X^T, Y^T, Z^T]^T$ that allows a group of parts translating along different directions, one of the vectors X, Y, Z have to be none-zero. Here, we assume $Y \neq 0$ without loss of generality. $Y = [y_1, y_2, \cdots, y_n]^T$ can be further separated into Y^+ and Y^- in which every element $y_i^+ (y_i^-)$ is positive (negative) or zero. Hence,

$$\begin{aligned} & AV \geq 0 \\ \Longrightarrow\ & A_Y Y \geq 0 \\ \Longrightarrow\ & y_j - y_i \geq 0 \\ \Longrightarrow\ & y_j - y_i \geq |y_j - y_i| \geq |y_i| - |y_j| \\ \Longrightarrow\ & \frac{1}{2}[(y_j + |y_j|) - (y_i + |y_i|)] \geq 0 \\ \Longrightarrow\ & y_j^+ - y_i^+ \geq 0 \\ \Longrightarrow\ & A_Y Y^+ \geq 0 \end{aligned} \tag{A.5}$$

Denoting $V' = [0, (Y^+)^T, 0]^T$, it is obvious that V' satisfies the three independent linear systems. Hence, V' is a valid solution of $AV \geq 0$. Here, the validity of V' implies that a subset of parts can move along the same direction $+e_y$ (i.e., $+y$ axial direction) simultaneously.

B Supplementary Material for *TI Assemblies*

This supplementary material is composed of three parts. The second part formulates the optimization on the 3D surface tessellation (Section 5.2). The third part describes the optimization to find the vertices of the target feasible section (Section 5.3.1). The last part presents the gradient-based approach to solve the structural optimization on the topological interlocking (TI) assemblies (Section 5.3.2).

B.1 Optimization of 3D Surface Tessellation

Taking a surface tessellation **T** using the conformal maps as a good initialization, we further optimize its vertices while fixing their connectivity for desirable properties described in the paper (i.e., planarity and regularity of each face, and small dihedral angles between adjacent

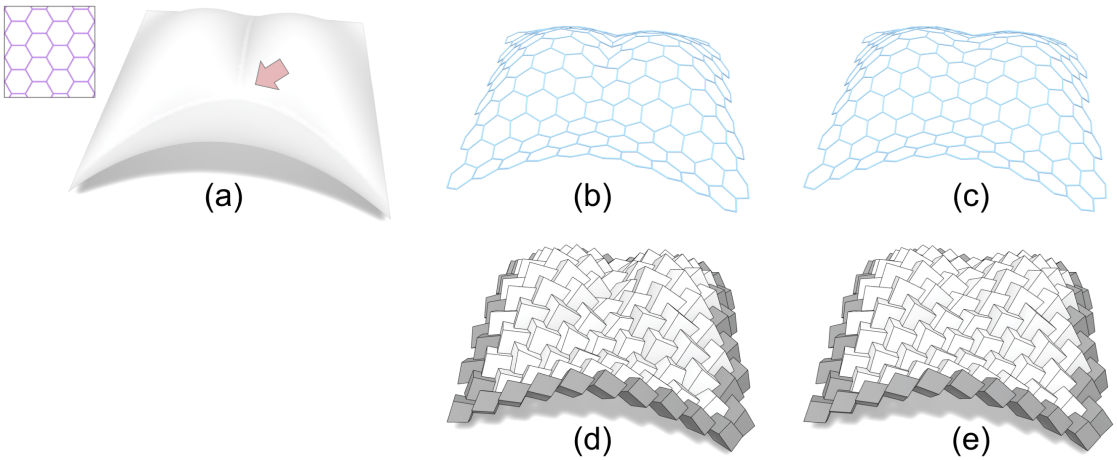


Figure B.1 – (a) Input reference surface and 2D tessellation. (b) Initial surface tessellation using conformal maps. (c) Optimized surface tessellation. (d&e) Corresponding TI assemblies with the same α.

faces). We formulate this optimization as an energy minimization problem:

$$E = \omega_1 E_{\text{surf}} + \omega_2 E_{\text{bound}} + \omega_3 E_{\text{planar}} + \omega_4 E_{\text{regular}} + \omega_5 E_{\text{coplanar}} \tag{B.1}$$

where E_{surf} aims to make the tessellation $\mathbf{T}$ resemble the reference surface $\mathbf{S}$; E_{bound} aims to preserve boundary of the tessellation $\mathbf{T}$; E_{planar} and E_{regular} aim to make faces in $\mathbf{T}$ planar and regular, respectively; and E_{coplanar} aims to ensure sufficient dihedral angles among adjacent faces in the tessellation $\mathbf{T}$.

In particular, the surface term E_{surf} is a summation of the distance from each vertex$\mathbf{v}_i$ of the tessellation $\mathbf{T}$ to the surface $\mathbf{S}$:

$$E_{\text{surf}} = \sum_{\mathbf{v}_i \in \mathbf{T}} d^2(\mathbf{v}_i, \mathbf{S}) \tag{B.2}$$

The boundary term E_{bound} evaluates the distance from each boundary vertex $\mathbf{v}_{b_i}$ of $\mathbf{T}$ to its initial position $\mathbf{v}_{b_i}^{\text{init}}$:

$$E_{\text{bound}} = \sum ||\mathbf{v}_{b_i} - \mathbf{v}_{b_i}^{\text{init}}||^2 \tag{B.3}$$

The E_{planar} and E_{regular} terms for each face and E_{coplanar} term for each pair of adjacent faces in the tessellation $\mathbf{T}$ are computed following the projection-based approach in [Bouaziz et al., 2012].

Figure B.1 shows an example surface tessellation $\mathbf{T}$ before and after our optimization, as well as the corresponding TI assemblies. Thanks to the desirable properties of the optimized tessellation, the resulting TI assembly have more coherent arrangement of the blocks; compare Figure B.1-d&e.

B.2 Compute Target Force Directions

Given a feasible section $\mathbb{S}(\mathbf{P})$ of an assembly $\mathbf{P}$, the radius r_0 of its largest inner circle centered at the origin is equal to $\tan(\Phi)$, where Φ is the stability measure of the assembly $\mathbf{P}$. The distance between a 2D point $\mathbf{v}_k$ and $\mathbb{S}(\mathbf{P})$ is denoted as:

$$\text{dist}(\mathbf{v}_k, \mathbb{S}(\mathbf{P})) = \begin{cases} \min\limits_{\mathbf{s} \in \partial\mathbb{S}} ||\mathbf{v}_k - \mathbf{s}|| & \mathbf{v} \notin \mathbb{S} \\ 0 & \mathbf{v} \in \mathbb{S} \end{cases} \tag{B.4}$$

where $\partial\mathbb{S}$ denotes all vertices on the boundary of $\mathbb{S}(\mathbf{P})$.

Our goal is to compute a K-gon with vertices $\{\mathbf{v}_k\}$ that contains the target circle with $r_1 = \tan(\omega\Phi)$ and is close to the feasible section $\mathbb{S}(\mathbf{P})$. We formulate this as an optimization problem:

$$\min_{\{\mathbf{v}_k\}} \sum_{k=1}^{K} \text{dist}(\mathbf{v}_k, \mathbb{S}(\mathbf{P})) \tag{B.5}$$

$$\text{s.t.} \qquad \text{dist}(\overline{\mathbf{v}_k\mathbf{v}_{k+1}}, \mathbf{0}) >= r_1, \forall k \in 1, \cdots, K$$

where the line segment $\overline{\mathbf{v}_k\mathbf{v}_{k+1}}$ is kth edge of the K-gon and $\mathbf{v}_{K+1} = \mathbf{v}_0$. The constraints ensure that the target feasible polygon always contains the target circle. Since this optimization has non-convex constraints, we solve it by using the interior conjugate gradient algorithm from `knitro`.

B.3 Gradient-based Structural Optimization

In the paper, structural optimization on the TI assemblies is formulated as:

$$\{\alpha_{ij}^*\} = \arg\min_{\{\alpha_{ij}\}} \sum_{k=1}^{K} \mathbf{f}_k^T \mathbf{H} \mathbf{f}_k \tag{B.6}$$

$$\begin{aligned} \text{s.t.} \qquad & \text{Area}(\ C_l\ ) \geq A_{\text{thres}},\ \forall \text{contact } C_l \\ & \max(\alpha^{\min}, \alpha_{ij}^{\min}) \leq \alpha_{ij} \leq \min(\alpha^{\max}, \alpha_{ij}^{\max}) \\ & \mathbf{A}_{\text{eq}} \cdot \mathbf{F}_k = \mathbf{W}_k \end{aligned}$$

We solve above optimization following the gradient-based approach in [Whiting et al., 2012]. Given the optimization problem formulated as a quadratic program (QP) with inequality constraints, it can be transformed into QP with only equality constraints by considering active constraints at a local intermediate solution $\mathbf{f}_k^*$, leading to a closed-form force solution:

$$\mathbf{f}_k^* = \mathbf{H}^{-1}\mathbf{C}^T(\mathbf{C}\mathbf{H}^{-1}\mathbf{C}^T)^{-1}\mathbf{b}_k \tag{B.7}$$

where

$$\mathbf{C} = \begin{bmatrix} \mathbf{A}_{\text{eq}} \\ \tilde{\mathbf{I}}_{lb} \end{bmatrix}, \quad \mathbf{b}_k = \begin{bmatrix} \mathbf{W}_k \\ \mathbf{0} \end{bmatrix}$$

and the active lower bounds are denoted as $\tilde{\mathbf{I}}_{lb} \cdot \mathbf{f}_k^* = \mathbf{0}$, which contains the contact forces in $\mathbf{f}_k^*$ that are exactly at zero.

We assume that the topology of each block and that of the block contact polygons remain the same for the differential movement of $\{\alpha_{ij}\}$, and derive an analytic gradient for $\mathbf{f}_k^*$ with respect to the vector rotation angles $\{\alpha_{ij}\}$. Among the components of $\mathbf{f}_k^*$ in Equation B.7, only $\mathbf{A}_{\text{eq}}$ and $\mathbf{W}_k$ are the functions of $\{\alpha_{ij}\}$ while $\mathbf{H}$ and $\tilde{\mathbf{I}}_{lb}$ are constants. Hence, we compute the gradient of each element in $\mathbf{A}_{\text{eq}}$, $\mathbf{W}_k$, as well as C_l with respect to $\{\alpha_{ij}\}$ using chain rule. In particular, the variables in $\mathbf{A}_{\text{eq}}$ include block centroid, block contact normal and contact vertex, while the variables in $\mathbf{W}_k$ are only about block volume. In the followings, we describe the derivations of each variable's gradient with respect to the vector rotation angles $\{\alpha_{ij}\}$.

B.3.1 Definition

Following the constructive procedure to generate TI blocks, our differentiation pipeline for the variables in $\mathbf{A}_{\text{eq}}$, $\mathbf{W}_k$, and C_l is shown as follows

$$\begin{array}{ccccc} & & \nearrow & \nabla \mathbf{n}_{ij}^E & \\ \alpha_{ij} \rightarrow & \nabla \mathbf{n}_{ij}^F & & \nearrow & \nabla \mathbf{o}_i \\ & & \searrow & \nabla \mathbf{v}_i^h \rightarrow & \nabla V_i \\ & & & \searrow & \nabla \mathbf{c}_{ij}^g \rightarrow \nabla A_{ij} \end{array}$$

The meaning of each variable in the pipeline is described as below and our goal is to calculate the derivatives of each variable with respect to α_{ij}:

1. $\mathbf{n}_{ij}^F$: normal of a face-face contact between P_i and P_j (pointing towards P_j).
2. $\mathbf{n}_{ij}^E$: normal of an edge-edge contact between P_i and P_j (pointing towards P_j).
3. $\mathbf{v}_i^h$: h th vertex of P_i.
4. $\mathbf{c}_{ij}^g$: g th vertex of the contact interface between P_i and P_j.
5. A_{ij}: area of contact interface between P_i and P_j.
6. $\mathbf{o}_i$: center of mass of P_i.
7. V_i: volume of P_i.

In our differentiation, most elements of gradients $\nabla \mathbf{X}$ ($\mathbf{X} = \mathbf{n}_{ij}, \mathbf{v}_i^k, ...$) are zero due to the fact that $\nabla \mathbf{X}$ only depends on a few vector angles $\{\alpha_{ij}\}$. Hence, we only need to compute the potentially non-zero partial derivatives. We use the notation $\frac{\partial \mathbf{X}}{\partial \alpha}$ to represent one of the non-zero derivatives $\frac{\partial \mathbf{X}}{\partial \alpha_{ij}}$ for writing simplicity.

B.3.2 Face-Face Contact Normal $\mathbf{n}_{ij}^F$

The initial face-face contact normal $\mathbf{n}_{ij}^{\texttt{init}}$ ($\alpha_{ij} = 0$) between P_i and P_j is:

$$\mathbf{n}_{ij}^{\texttt{init}} = \texttt{Normalized}(\mathbf{e}_{ij} \times (\mathbf{N}_i + \mathbf{N}_j)) \tag{B.8}$$

While `Normalized` is a function which normalizes the input vector and $\mathbf{e}_{ij}$ is the halfedge vector between P_i and P_j. Rotating initial contact normal by $x_{ij}\alpha_{ij}$ around the halfedge vector $\mathbf{e}_{ij}$ gives the expression of $\mathbf{n}_{ij}^F$:

$$E_{ij} = \begin{bmatrix} 0 & -e_{ij}^z & e_{ij}^y \\ e_{ij}^z & 0 & -e_{ij}^x \\ -e_{ij}^y & e_{ij}^x & 0 \end{bmatrix} \tag{B.9}$$

$$\mathbf{n}_{ij}^{F} = [\cos(x_{ij}\alpha_{ij})E_{ij} + \sin(x_{ij}\alpha_{ij})I_{3\times 3}]\mathbf{n}_{ij}^{\texttt{init}} \tag{B.10}$$

Thus, its derivative is:

$$\frac{\partial \mathbf{n}_{ij}^{F}}{\partial \alpha_{ij}} = [-x_{ij}\sin(x_{ij}\alpha_{ij})E_{ij} + x_{ij}\cos(x_{ij}\alpha_{ij})I_{3\times 3}]\mathbf{n}_{ij}^{\texttt{init}} \tag{B.11}$$

B.3.3 Edge-Edge Contact Normal $\mathbf{n}_{ij}^{E}$

The normal of edge-edge contact is involved four face normals associated to the contact point, the two normals $\mathbf{n}_0, \mathbf{n}_1$ from P_i and the two normals $\mathbf{n_2}, \mathbf{n_3}$ from P_j. Then the normal of its edge-edge contact is:

$$\mathbf{n}_{ij}^{E} = \frac{(\mathbf{n}_0 \times \mathbf{n}_1) \times (\mathbf{n}_2 \times \mathbf{n}_3)}{||(\mathbf{n}_0 \times \mathbf{n}_1) \times (\mathbf{n}_2 \times \mathbf{n}_3)||} \tag{B.12}$$

In case of the $\mathbf{n}_{ij}^{E}$ pointing toward P_i instead of P_j, we will reverse the direction of $\mathbf{n}_{ij}^{E}$. Its derivative can be computed by applying the chain rules.

B.3.4 Block Vertex $\mathbf{v}_i^h$

In section 5 of the paper, the geometry of P_i is constructed by intersecting several 3D half spaces (planes). A valid part vertex $\mathbf{v}_i^h$ are on at least three of these half spaces(planes):

$$(\mathbf{n}_j^{\mathbf{v}_i^h}, d_0^{\mathbf{v}_i^h}), j = 0, 1, 2 \tag{B.13}$$

The normal vector $\mathbf{n}_j^{\mathbf{v}_i^h}$, a.k.a $\mathbf{n}_j$, is one of the face-face contact normals associated to P_i vertex $\mathbf{v}_i^h$. The $d_j^{\mathbf{v}_i^h}$, a.k.a d_j, is the inner product of any point on jth plane in Eq.B.13 and the normal $\mathbf{n}_j$. Solving the following linear system gives the vertex $\mathbf{v}_i^h$:

$$\begin{aligned} \mathbf{v}_i^h &= \begin{bmatrix} \mathbf{n}_0^T \\ \mathbf{n}_1^T \\ \mathbf{n}_2^T \end{bmatrix}^{-1} \begin{bmatrix} d_0 \\ d_1 \\ d_2 \end{bmatrix} \\ &= (N_i^h)^{-1} D_i^h \end{aligned} \tag{B.14}$$

According to the derivative formulation of inverse matrix:

$$\frac{\partial N^{-1}}{\partial \alpha} = -N^{-1}\frac{\partial N}{\partial \alpha}N^{-1} \tag{B.15}$$

substitute Eq. B.15 into Eq. B.14 gives:

$$\frac{\partial \mathbf{v}_i^h}{\partial \alpha} = -(N_i^h)^{-1}\frac{\partial N_i^h}{\partial \alpha}(N_i^h)^{-1}D_i^h + (N_i^h)^{-1}\frac{\partial D_i^h}{\partial \alpha} \tag{B.16}$$

B.3.5 Block Volume V_i

Triangulating the faces of P_i gives a set of triangles $A_t(\mathbf{a}_t, \mathbf{b}_t, \mathbf{c}_t)$, $t = 0, \cdots, T-1$ which is ordered counter clockwise on its corresponding face of P_i. Suppose

$$\hat{\mathbf{n}}_t = (\mathbf{b}_t - \mathbf{a}_t) \times (\mathbf{c}_t - \mathbf{a}_t) \tag{B.17}$$

The formulation of block volume is:

$$V_i = \frac{1}{6}\sum_{t=0}^{T-1} \mathbf{a}_t \cdot \hat{\mathbf{n}}_t \tag{B.18}$$

Its derivative can be computed by applying the chain rules. Please refer to [Nurnberg, 2013] for more details.

B.3.6 Block Centroid $\mathbf{o}_i$

Denoting the standard basis in $\mathbb{R}^3$ by $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$, the centroid $\mathbf{o}_i$ of P_i is:

$$\begin{aligned}\mathbf{e}_u \cdot \mathbf{o}_i = \frac{1}{48V_i}\sum_{t=0}^{T-1} (\hat{\mathbf{n}}_t \cdot \mathbf{e}_u)([(\mathbf{a}_t + \mathbf{b}_t)\cdot \mathbf{e}_u]^2 \\ + [(\mathbf{c}_t + \mathbf{b}_t)\cdot \mathbf{e}_u]^2 \\ + [(\mathbf{c}_t + \mathbf{a}_t)\cdot \mathbf{e}_u]^2), u = 1,2,3\end{aligned} \tag{B.19}$$

Its derivative can be computed by applying the chain rules. Please refer to [Nurnberg, 2013] for more details.

B.3.7 Contact Vertex $\mathbf{c}_{ij}^g$

The paper lists two kinds of contact vertices, we only give the derivatives expression for the face-face contact case while the vertex of an edge-edge contact is a constant point of the surface tessellation.

The contact polygon C_l are the result of boolean intersection between two overlap faces of neighboring P_i and P_j, The vertices of this polygon are either 1) a vertex of the two neighboring part ($\mathbf{v}_i^h$ or $\mathbf{v}_j^h$) ; or 2) the intersection of two non-parallel edges of P_i and P_j, which called $\mathbf{c}_{ij}^g$ (g th point of the C_l). Suppose the two lines are:

$$(\mathbf{v}_0^{\mathbf{c}_{ij}^g}, \mathbf{v}_1^{\mathbf{c}_{ij}^g}), (\mathbf{v}_2^{\mathbf{c}_{ij}^g}, \mathbf{v}_3^{\mathbf{c}_{ij}^g}) \tag{B.20}$$

a.k.a as:

$$(\mathbf{v}_0, \mathbf{v}_1), (\mathbf{v}_2, \mathbf{v}_3) \tag{B.21}$$

The contact vertex $\mathbf{c}_{ij}^g$ satisfies the following equations

$$\mathbf{c}_{ij}^g = \mathbf{v}_0 + (\mathbf{v}_1 - \mathbf{v}_0)\, s = \mathbf{v}_2 + (\mathbf{v}_3 - \mathbf{v}_2)\, t \tag{B.22}$$

Denote the $\mathbf{v}_{a,b} = \mathbf{v}_a - \mathbf{v}_b$. The solution of linear equation s, t is:

$$\begin{bmatrix} s \\ t \end{bmatrix} = \begin{bmatrix} \mathbf{v}_{1,0} \cdot \mathbf{v}_{1,0} & -\mathbf{v}_{1,0} \cdot \mathbf{v}_{2,3} \\ \mathbf{v}_{1,0} \cdot \mathbf{v}_{2,3} & -\mathbf{v}_{2,3} \cdot \mathbf{v}_{2,3} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{v}_{1,0} \cdot \mathbf{v}_2 - \mathbf{v}_{1,0} \cdot \mathbf{v}_0 \\ \mathbf{v}_{3,2} \cdot \mathbf{v}_2 - \mathbf{v}_{3,2} \cdot \mathbf{v}_0 \end{bmatrix} \tag{B.23}$$

Substituting the solution into Equation B.22, its derivative can be computed by applying the chain rules.

B.3.8 Contact Area A_{ij}

The contact polygon C_l is a list of points $\mathbf{c}_{ij}^g (g = 0, ..., |C_l| - 1)$ which are in the same orientation of contact normal $\mathbf{n}_{ij}$. The area of the polygon is:

$$A_{ij} = \frac{1}{2} \sum_{g=1}^{|C_l|-2} [(\mathbf{c}_{ij}^g - \mathbf{c}_{ij}^0) \times (\mathbf{c}_{ij}^{g+1} - \mathbf{c}_{ij}^0)] \cdot \mathbf{n}_{ij} \tag{B.24}$$

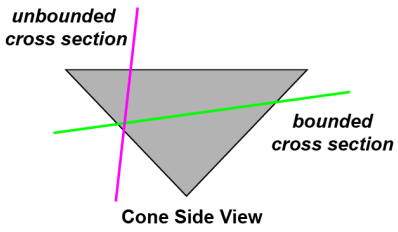
Its derivative can be computed by applying the chain rules.

C Supplementary Material for *Assemblies with Cone Joints*

The paper has mentioned supplementary material four times. We explain each of them following its order in the paper content.

C.1 Motion Cone Visualization

Choosing a proper cutting plane is needed to illustrate the 3D cone by its conic section. The inset shows a failure case where the conic section is an unbound region. Supposed that the plane has the form $\{\mathbf{x}|\mathbf{n}\cdot\mathbf{x}=1\}$, the normal $\mathbf{n}$ should be chosen to make the half-space $\{\mathbf{n}\cdot\mathbf{x}\geq 0\}$ contain the cones. More precisely, for cones in motion space, we choose the $\mathbf{n}$ to be the initial planar contact's normal appended with zeros. For cones in generalized normal space, we choose the $\mathbf{n}$ to be one of the part's translational disassembling directions appended with zeros.



C.2 Kinematic Based Infeasibility Measure

Computing the kinematic based infeasibility measure in Equation C.1 is a dual problem of computing the forced-based infeasibility measure in Equation C.2.

Dual Problem:

$$\begin{aligned} \max_{\boldsymbol{v}} \quad & \boldsymbol{w}^T\boldsymbol{v} - \frac{1}{2}\boldsymbol{v}^T\boldsymbol{v} \\ \text{s.t.} \quad & B_{\text{in}}\boldsymbol{v} \geq 0 \end{aligned} \tag{C.1}$$

Primal Problem:

$$\begin{aligned} \min_{\boldsymbol{f},\boldsymbol{s}} \quad & \frac{1}{2}\boldsymbol{s}^T\boldsymbol{s} \\ \text{s.t.} \quad & A_{\text{eq}}\boldsymbol{f}+\boldsymbol{s}=-\boldsymbol{w}, \\ & \boldsymbol{f}\geq 0 \end{aligned} \tag{C.2}$$

Proof:

The Lagrange function of Equation C.2 is:

$$L(\boldsymbol{f},\boldsymbol{s},\boldsymbol{\mu},\boldsymbol{\lambda})=\frac{1}{2}\boldsymbol{s}^T\boldsymbol{s}+\boldsymbol{\mu}^T(A_{\text{eq}}\boldsymbol{f}+\boldsymbol{s}+\boldsymbol{w})-\boldsymbol{\lambda}^T\boldsymbol{f} \tag{C.3}$$

The dual Lagrange function $g(\boldsymbol{\mu},\boldsymbol{\lambda})$ is defined as the infimum of the Lagrange function $L(\boldsymbol{f},\boldsymbol{s},\boldsymbol{\mu},\boldsymbol{\lambda})$ in variables $\boldsymbol{f},\boldsymbol{s}$.

$$g(\boldsymbol{\mu},\boldsymbol{\lambda})=\inf_{\boldsymbol{f},\boldsymbol{s}} L(\boldsymbol{f},\boldsymbol{s},\boldsymbol{\mu},\boldsymbol{\lambda}) \tag{C.4}$$

The Lagrange function L is a quadratic function which reaches its minimum when both its derivatives with respect to $\boldsymbol{f},\boldsymbol{s}$ are zero.

$$\frac{\partial L}{\partial \boldsymbol{f}}=A_{\text{eq}}^T\boldsymbol{\mu}-\boldsymbol{\lambda}=0 \tag{C.5}$$

$$\frac{\partial L}{\partial \boldsymbol{s}}=\boldsymbol{s}+\boldsymbol{\mu}=0 \tag{C.6}$$

Substitute Equation C.5 and C.6 into the Lagrange function (Equation C.3).

$$g(\boldsymbol{\mu},\boldsymbol{\lambda})=-\frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\mu}+\boldsymbol{\mu}^T\boldsymbol{w} \tag{C.7}$$

where the lagrange multipliers $\boldsymbol{\lambda}$ must be non-negative. Due to the strong duality theorem, the dual of the optimization problem (Equation C.2) is:

$$\begin{aligned} \max_{\boldsymbol{\mu}} \quad & -\frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\mu}+\boldsymbol{\mu}^T\boldsymbol{w} \\ \text{s.t.} \quad & A_{\text{eq}}^T\boldsymbol{\mu}-\boldsymbol{\lambda}=0, \\ & \boldsymbol{\lambda}\geq 0 \end{aligned} \tag{C.8}$$

Since $B_{\text{in}}=A_{\text{eq}}^T$, by renaming $\boldsymbol{\mu}$ to $\boldsymbol{v}$ in Equation C.8, we prove that Equation C.1 is a dual problem of Equation C.2.

C.3 Infeasibility Derivatives $\frac{\partial E(\boldsymbol{w},\{\bar{\mathbf{V}}_{i,j}\})}{\partial \Psi_{i,j}}$

This section explains how to compute the first order derivatives of infeasibility energy $E(\boldsymbol{w},\{\bar{\mathbf{V}}_{i,j}\})$ with respect to the approximated motion cones' variables $\Psi_{i,j}$.

Our kinematic design uses an off-the-shelf interior point method to solve the optimization problem. Common solvers like BFGS require the first-order derivatives. The infeasibility energy $E(\mathbf{w}, \{\tilde{\mathbf{V}}_{i,j}\})$ is formulated as Equation C.1, where the matrix B_{in} is a function of $\Psi_{i,j}$ and the vector $\boldsymbol{w}$ is constant. Amos et al. [Amos and Kolter, 2019] proposed a sensitive analysis tool that computes the derivatives for series of quadratic programming problems including Equation C.1. Their core idea is that any optimal solution of Equation C.1 must satisfy the following KKT conditions.

$$\begin{aligned} \boldsymbol{\mu} &\geq 0 \quad \text{positive multipliers} \\ -\boldsymbol{\mu}^T(B_{\text{int}}\boldsymbol{v}) &= 0 \quad \text{complementary} \\ \boldsymbol{w} - \boldsymbol{v} - \boldsymbol{\mu}^T B_{\text{int}} &= 0 \quad \text{stationarity} \end{aligned} \tag{C.9}$$

where the $\boldsymbol{\mu}$ is the Lagrange multipliers of the constraints $B_{\text{int}}\boldsymbol{v} \geq 0$. Let's ignore the inequality constraints $\boldsymbol{\mu} \geq 0$ and take partial derivatives on both sides of the equalities in Equation C.9:

$$\begin{aligned} \left(\frac{\partial \boldsymbol{\mu}}{\partial \Psi_{i,j}}\right)^T B_{\text{int}}\boldsymbol{v}^* + \boldsymbol{\mu}^{*T} B_{\text{int}} \frac{\partial \boldsymbol{v}}{\partial \Psi_{i,j}} &= -\boldsymbol{\mu}^{*T} \frac{\partial B_{\text{in}}}{\partial \Psi_{i,j}} \boldsymbol{v}^* \\ \frac{\partial \boldsymbol{v}}{\partial \Psi_{i,j}} + \left(\frac{\partial \boldsymbol{\mu}}{\partial \Psi_{i,j}}\right)^T B_{\text{int}} &= -\boldsymbol{\mu}^{*T} \frac{\partial B_{\text{in}}}{\partial \Psi_{i,j}} \end{aligned} \tag{C.10}$$

where $\boldsymbol{v}^*$ and $\boldsymbol{\mu}^*$ are the optimal solutions for the optimization problem. We can solve this large sparse linear system (Equation C.10) to obtain the derivatives $\frac{\partial \boldsymbol{\mu}}{\partial \Psi_{i,j}}$ and $\frac{\partial \boldsymbol{v}}{\partial \Psi_{i,j}}$. In practice, this system is pre-factorized to reduce solving time. The derivatives of infeasibility energy then are:

$$\frac{\partial E(\boldsymbol{w}, \{\tilde{\mathbf{V}}_{i,j}\})}{\partial \Psi_{i,j}} = (\boldsymbol{w} - \boldsymbol{v}^*)^T \frac{\partial \boldsymbol{v}}{\partial \Psi_{i,j}} \tag{C.11}$$

Moreover, we can adpot the same strategy to solve the optimization problem in geometric realization whose objective is also a quadratic function. Its derivatives can be computed by the same sensitive analysis approach.

C.4 New Interlocking Test

Supposed that all cone joints used in an assembly are not degenerated, we use a special external force $\boldsymbol{w}_{\text{int}} = -\sum_{i<j}(\boldsymbol{v}_j - \boldsymbol{v}_i)\cdot \boldsymbol{d}_{i,j}$ to verify the assembly's globally interlocking property. Here, $\boldsymbol{d}_{i,j}$ is chosen such that $(\boldsymbol{v}_j - \boldsymbol{v}_i)\cdot \boldsymbol{d}_{i,j} \geq 0$ and $(\boldsymbol{v}_j - \boldsymbol{v}_i)\cdot \boldsymbol{d}_{i,j} = 0$ if only if $\boldsymbol{v}_j = \boldsymbol{v}_i$. The assembly is globally interlocking if only if the infeasibility energy (Equation C.1) under the external force $\boldsymbol{w}_{\text{int}}$ is zero.

Proof:

To avoid all parts in the assembly moving together, let's assume the key part and one of the

remaining parts are fixed. The assembly is globally interlocking if no part can move in this circumstance.

For each cone joint whose motion space is $\mathbf{V}_{i,j}$, the associated two parts' velocities satisfy $\boldsymbol{v}_j - \boldsymbol{v}_i \in \mathbf{V}_{i,j}$. There always exists a constant vector $\boldsymbol{d}_{i,j}$, often chosen to be the initial planar contact normal appended with extra zeros, such that $(\boldsymbol{v}_j - \boldsymbol{v}_i) \cdot \boldsymbol{d}_{i,j} \geq 0$.

Moreover, if the cone $\mathbf{V}_{i,j}$ is not degenerated, the condition $(\boldsymbol{v}_j - \boldsymbol{v}_i) \cdot \boldsymbol{d}_{i,j} = 0$ holds if only if $\boldsymbol{v}_j = \boldsymbol{v}_i$. This constant vector $-\boldsymbol{d}_{i,j}$ acts like a repulsive force, which pushes the two adjacent parts away from each other. Since the external force $\boldsymbol{w}_{\text{int}}$ is set to be the summation of all repulsive forces $-\sum_{i,j}(\boldsymbol{v}_j - \boldsymbol{v}_i) \cdot \boldsymbol{d}_{i,j}$. The infeasibility energy is zero if only if each term $(\boldsymbol{v}_j - \boldsymbol{v}_i) \cdot \boldsymbol{d}_{i,j} = 0$. All parts must have the same velocity (i.e., zero) and therefore the assembly is globally interlocking.

Implementing this special external force $\boldsymbol{w}_{\text{int}}$ into our kinematic-geometric optimization framework allows designing new globally interlocking assemblies.

Bibliography

[Agrawala et al., 2003] Agrawala, M., Phan, D., Heiser, J., Haymaker, J., Klingner, J., Hanrahan, P., and Tversky, B. (2003). Designing effective step-by-step assembly instructions. *ACM Trans. on Graph. (SIGGRAPH)*, 22(3):828–837.

[Akleman et al., 2020] Akleman, E., Krishnamurthy, V. R., Fu, C.-A., Subramanian, S. G., Ebert, M., Eng, M., Starrett, C., and Panchal, H. (2020). Generalized Abeille Tiles: Topologically interlocked space-filling shapes generated based on fabric symmetries. *Comp. & Graph. (SMI)*, 89:156–166.

[Alexandrov, 2005] Alexandrov, A. D. (2005). *Convex Polyhedra*. Springer.

[Amos and Kolter, 2019] Amos, B. and Kolter, J. Z. (2019). Optnet: Differentiable optimization as a layer in neural networks.

[ApS, 2019] ApS, M. (2019). Mosek software package. https://www.mosek.com/.

[Araújo et al., 2019] Araújo, C., Cabiddu, D., Attene, M., Livesu, M., Vining, N., and Sheffer, A. (2019). Surface2Volume: Surface segmentation conforming assemblable volumetric partition. *ACM Trans. on Graph. (SIGGRAPH)*, 38(4):1:1–1:16.

[Artelys, 2019] Artelys (2019). Knitro software package. https://www.artelys.com/solvers/knitro/.

[Artelys, 2020] Artelys (2020). Knitro software package. https://www.artelys.com/solvers/knitro/.

[Block and Ochsendorf, 2007] Block, P. and Ochsendorf, J. (2007). Thrust Network Analysis: A new methodology for three-dimensional equilibrium. *Journal of the International Association for Shell and Spatial Structures*, 48(3):167 – 173.

[Bouaziz et al., 2012] Bouaziz, S., Deuss, M., Schwartzburg, Y., Weise, T., and Pauly, M. (2012). Shape-up: Shaping discrete geometry with projections. *Comp. Graph. Forum (SGP)*, 31(5):1657–1667.

[Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

[Brunete et al., 2017] Brunete, A., Ranganath, A., Segovia, S., de Frutos, J. P., Hernando, M., and Gambao, E. (2017). Current trends in reconfigurable modular robots design. *International Journal of Advanced Robotic Systems*, pages 1–21.

[Chen et al., 2014] Chen, H., Jin, S., Li, Z., and Lai, X. (2014). A comprehensive study of three dimensional tolerance analysis methods. *Computer-Aided Design*, 53:1–13.

[Cutler, 1978] Cutler, W. H. (1978). The six-piece burr. *Journal of Recreational Mathematics*, 10(4):241–250.

[Davidson and Hunt, 2004] Davidson, J. K. and Hunt, K. H. (2004). *Robots and Screw Theory: Applications of Kinematics and Statics to Robotics*. Oxford University Press.

[de Mello and Sanderson, 1990] de Mello, L. S. H. and Sanderson, A. C. (1990). AND/OR graph representation of assembly plans. *IEEE Transactions on Robotics and Automation*, 6(2):188–199.

[Deepak, 2012] Deepak, B. (2012). Sustainable dry interlocking block masonry construction. In *15th International Brick and Block Masonry Conference*.

[Deuss et al., 2015] Deuss, M., Deleuran, A. H., Bouaziz, S., Deng, B., Piker, D., and Pauly, M. (2015). Shapeop - a robust and extensible geometric modelling paradigm. In *Design Modelling Symposium*, pages 505–515. https://www.shapeop.org/.

[Deuss et al., 2014] Deuss, M., Panozzo, D., Whiting, E., Liu, Y., Block, P., Sorkine-Hornung, O., and Pauly, M. (2014). Assembling self-supporting structures. *ACM Trans. on Graph. (SIGGRAPH Asia)*, 33(6):214:1–214:10.

[Duenser et al., 2020] Duenser, S., Poranne, R., Thomaszewski, B., and Coros, S. (2020). RoboCut: Hot-wire cutting with robot-controlled flexible rods. *ACM Trans. on Graph. (SIGGRAPH)*, 39(4):98:1–98:15.

[Dyskin et al., 2013] Dyskin, A., Pasternak, E., and Estrin, Y. (2013). Topological interlocking as a design principle for hybrid materials. In *Proceedings of the 8th Pacific Rim International Congress on Advanced Materials and Processing*, pages 1525–1534.

[Dyskin et al., 2003a] Dyskin, A. V., Estrin, Y., Kanel-Belov, A. J., and Pasternak, E. (2003a). Topological interlocking of platonic solids: A way to new materials and structures. *Philosophical Magazine Letters*, 83(3):197–203.

[Dyskin et al., 2019] Dyskin, A. V., Estrin, Y., and Pasternak, E. (2019). Topological interlocking materials. In Estrin, Y., Bréchet, Y., Dunlop, J., and Fratzl, P., editors, *Architectured Materials in Nature and Engineering*, chapter 2, pages 23—49. Springer International Publishing.

[Dyskin et al., 2003b] Dyskin, A. V., Estrin, Y., Pasternak, E., Khor, H. C., and Kanel-Belov, A. J. (2003b). Fracture resistant structures based on topological interlocking with non-planar contacts. *Advanced Engineering Materials*, 5(3):116–119.

[Eversmann et al., 2017] Eversmann, P., Gramazio, F., and Kohler, M. (2017). Robotic prefabrication of timber structures: Towards automated large-scale spatial assembly. *Construction Robotics*, 1:49–60.

[Fairham, 2013] Fairham, W. (2013). *Woodwork Joints: How to Make and Where to Use Them*. Skyhorse.

[Fallacara et al., 2019] Fallacara, G., Barberio, M., and Colella, M. (2019). Topological interlocking blocks for architecture: From flat to curved morphologies. In Estrin, Y., Bréchet, Y., Dunlop, J., and Fratzl, P., editors, *Architectured Materials in Nature and Engineering*, chapter 14, pages 423–445. Springer International Publishing.

[Farkas, 1902] Farkas, J. (1902). Theorie der einfachen ungleichungen. *Journal für die Reine und Angewandte Mathematik*, (124):1 – 27.

[Frick et al., 2015] Frick, U., Mele, T. V., and Block, P. (2015). Decomposing three-dimensional shapes into self-supporting, discrete-element assemblies. In *Proc. the Design Modelling Symposium*, pages 187–201.

[Fu et al., 2015] Fu, C.-W., Song, P., Yan, X., Yang, L. W., Jayaraman, P. K., and Cohen-Or, D. (2015). Computational interlocking furniture assembly. *ACM Trans. on Graph. (SIGGRAPH)*, 34(4):91:1–91:11.

[Gao et al., 2019] Gao, Y., Wu, L., Yan, D.-M., and Nan, L. (2019). Near support-free multi-directional 3D printing via global-optimal decomposition. *Graphical Models (CVM)*, 104. Article No. 101034.

[Ghandi and Masehian, 2015] Ghandi, S. and Masehian, E. (2015). Review and taxonomies of assembly and disassembly path planning problems and approaches. *Computer-Aided Design*, 67-68:58–86.

[Halperin et al., 2000] Halperin, D., Latombe, J.-C., and Wilson, R. H. (2000). A general framework for assembly planning: The motion space approach. *Algorithmica*, 26(3–4):577–601.

[Heiser et al., 2004] Heiser, J., Phan, D., Agrawala, M., Tversky, B., and Hanrahan, P. (2004). Identification and validation of cognitive design principles for automated generation of assembly instructions. In *Proc. the Working Conference on Advanced Visual Interfaces*, pages 311–319.

[Heyman, 1966] Heyman, J. (1966). The stone skeleton. *International Journal of Solids and Structures*, 2(2):249–279.

[Hu et al., 2018] Hu, R., Savva, M., and van Kaick, O. (2018). Functionality representations and applications for shape analysis. *Comp. Graph. Forum (Eurographics STAR – State of The Art Report)*, 37(2):603–624.

[Jacobson et al., 2018] Jacobson, A., Panozzo, D., et al. (2018). libigl: A simple C++ geometry processing library. https://libigl.github.io/.

[Javan et al., 2016] Javan, A. R., Seifi, H., Xu, S., and Xie, Y. M. (2016). Design of a new type of interlocking brick and evaluation of its dynamic performance. In *Proceedings of the International Association for Shell and Spatial Structures Annual Symposium*, pages 1–8.

[Jiménez, 2013] Jiménez, P. (2013). Survey on assembly sequencing: A combinatorial and geometrical perspective. *Journal of Intelligent Manufacturing*, 24(2):235–250.

[Jones and Wilson, 1996] Jones, R. E. and Wilson, R. H. (1996). A survey of constraints in automated assembly planning. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 1525–1532.

[Kanel-Belov et al., 2010] Kanel-Belov, A. J., Dyskin, A. V., Estrin, Y., Pasternak, E., and Ivanov-Pogodaev, I. A. (2010). Interlocking of convex polyhedra: Towards a geometric theory of fragmented solids. *Moscow Mathematical Journal*, 10(2):337–342.

[Kerbl et al., 2015] Kerbl, B., Kalkofen, D., Steinberger, M., and Schmalstieg, D. (2015). Interactive disassembly planning for complex objects. *Comp. Graph. Forum (Eurographics)*, 34(2):287–297.

[Krishnamurthy et al., 2021] Krishnamurthy, V. R., Akleman, E., Subramanian, S. G., Ebert, M., Cui, J., an Fu, C., and Starrett, C. (2021). Geometrically interlocking space-filling tiling based on fabric weaves. *IEEE Trans. Vis. & Comp. Graphics*. DOI: 10.1109/TVCG.2021.3065457.

[Larsson et al., 2020] Larsson, M., Yoshida, H., Umetani, N., and Igarashi, T. (2020). Tsugite: Interactive design and fabrication of wood joints. In *Proc. ACM UIST*, pages 317–327.

[Leung et al., 2021] Leung, P. Y., Apolinarska, A. A., Tanadini, D., Gramazio, F., and Kohler, M. (2021). Automatic assembly of jointed timber structure using distributed robotic clamps. In Globa, A., van Ameijde, J., Fingrut, A., Kim, N., and Sky Lo, T. T., editors, *'PROJECTIONS' – Proceedings of the 26th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA 2021)*, volume 1, pages 583 – 592. 26th International Conference of the Association for Computer-Aided Architectural Design Research in Asia: Projections (CAADRIA 2021) (virtual); Conference Location: Hong Kong, China; Conference Date: March 29 – April 1, 2021; Conference lecture held on March 30, 2021. Due to the Coronavirus (COVID-19) the conference was conducted virtually.

[Ma et al., 2019] Ma, L., He, Y., Sun, Q., Zhou, Y., Zhang, C., and Wang, W. (2019). Constructing 3D self-supporting surfaces with isotropic stress using 4D minimal hypersurfaces of revolution. *ACM Trans. on Graph.*, 38(5):144:1–144:13.

[Magrisso et al., 2018] Magrisso, S., Mizrahi, M., and Zoran, A. (2018). Digital joinery for hybrid carpentry. In *Proc. ACM CHI*, pages 167:1–167:11.

[Masehian and Ghandi, 2020] Masehian, E. and Ghandi, S. (2020). ASPPR: A new assembly sequence and path planner/replanner for monotone and nonmonotone assembly planning. *Computer-Aided Design*, 123:102828:1–102828:22.

[Mellado et al., 2014] Mellado, N., Song, P., Yan, X., Fu, C.-W., and Mitra, N. J. (2014). Computational design and construction of notch-free reciprocal frame structures. In *Proc. Advances in Architectural Geometry*, pages 181–197.

[Mirkhalaf et al., 2018] Mirkhalaf, M., Zhou, T., and Barthelat, F. (2018). Simultaneous improvements of strength and toughness in topologically interlocked ceramics. *Proceedings of the National Academy of Sciences of the United States of America*, 115(37):9128–9133.

[Mitra et al., 2019] Mitra, N. J., Kokkinos, I., Guerrero, P., Thuerey, N., Kim, V., and Guibas, L. (2019). CreativeAI: Deep learning for graphics. In *SIGGRAPH Courses*.

[Nurnberg, 2013] Nurnberg, R. (2013). http://wwwf.imperial.ac.uk/ rn/centroid.pdf.

[Ochsendorf, 2002] Ochsendorf, J. A. (2002). *Collapse of Masonry Structures*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA.

[Panozzo et al., 2013] Panozzo, D., Block, P., and Sorkine-Hornung, O. (2013). Designing unreinforced masonry models. *ACM Trans. on Graph. (SIGGRAPH)*, 32(4):91:1–91:11.

[Shin et al., 2016] Shin, H. V., Porst, C. F., Vouga, E., Ochsendorf, J., and Durand, F. (2016). Reconciling elastic and equilibrium methods for static analysis. *ACM Trans. on Graph.*, 35(2):13:1–13:16.

[Siegmund et al., 2016] Siegmund, T., Barthelat, F., Cipra, R., Habtour, E., and Riddick, J. (2016). Manufacture and mechanics of topologically interlocked material assemblies. *Applied Mechanics Reviews*, 68(4). Article No. 040803.

[Skouras et al., 2015] Skouras, M., Coros, S., Grinspun, E., and Thomaszewski, B. (2015). Interactive surface design with interlocking elements. *ACM Trans. on Graph. (SIGGRAPH Asia)*, 34(6). Article No. 224.

[Song et al., 2016] Song, P., Deng, B., Wang, Z., Dong, Z., Li, W., Fu, C.-W., and Liu, L. (2016). CofiFab: Coarse-to-fine fabrication of large 3D objects. *ACM Trans. on Graph. (SIGGRAPH)*, 35(4):45:1–45:11.

[Song et al., 2012] Song, P., Fu, C.-W., and Cohen-Or, D. (2012). Recursive interlocking puzzles. *ACM Trans. on Graph. (SIGGRAPH Asia)*, 31(6):128:1–128:10.

[Song et al., 2017] Song, P., Fu, C.-W., Jin, Y., Xu, H., Liu, L., Heng, P.-A., and Cohen-Or, D. (2017). Reconfigurable interlocking furniture. *ACM Trans. on Graph. (SIGGRAPH Asia)*, 36(6):174:1–174:14.

[Song et al., 2015] Song, P., Fu, Z., Liu, L., and Fu, C.-W. (2015). Printing 3D objects with interlocking parts. *Comp. Aided Geom. Des. (GMP)*, 35-36:137–148.

[Stegmann, 2018] Stegmann, R. (2018). Rob's puzzle page - interlocking puzzles. http://www.robspuzzlepage.com/interlocking.htm.

[Sun et al., 2021] Sun, Y., Ouyang, W., Liu, Z., Ni, N., Savoye, Y., Song, P., and Liu, L. (2021). Computational design of self-actuated deformable solids via shape memory material. *IEEE Trans. Vis. & Comp. Graphics*. DOI: 10.1109/TVCG.2020.3039613.

[Tang et al., 2019] Tang, K., Song, P., Wang, X., Deng, B., Fu, C.-W., and Liu, L. (2019). Computational design of steady 3D dissection puzzles. *Comp. Graph. Forum (Eurographics)*, 38(2):291–303.

[Tarjan, 1972] Tarjan, R. (1972). Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160.

[Wang et al., 2019] Wang, Z., Song, P., Isvoranu, F., and Pauly, M. (2019). Design and structural optimization of topological interlocking assemblies. *ACM Trans. on Graph. (SIGGRAPH Asia)*, 38(6):193:1–193:13.

[Weizmann et al., 2017] Weizmann, M., Amir, O., and Grobman, Y. J. (2017). Topological interlocking in architecture: A new design method and computational tool for designing building floors. *International Journal of Architectural Computing*, 15(2):107–118.

[Whiting et al., 2009] Whiting, E., Ochsendorf, J., and Durand, F. (2009). Procedural modeling of structurally-sound masonry buildings. *ACM Trans. on Graph. (SIGGRAPH Asia)*, 28(5):112:1–112:9.

[Whiting et al., 2012] Whiting, E., Shin, H., Wang, R., Ochsendorf, J., and Durand, F. (2012). Structural optimization of 3d masonry buildings. *ACM Trans. on Graph. (SIGGRAPH Asia)*, 31(6):159:1–159:11.

[Wilson, 1992] Wilson, R. H. (1992). *On Geometric Assembly Planning*. PhD thesis, Stanford University.

[Wilson and Latombe, 1994] Wilson, R. H. and Latombe, J.-C. (1994). Geometric reasoning about mechanical assembly. *Artificial Intelligence*, 71(2):371–396.

[Wilson and Matsui, 1992] Wilson, R. H. and Matsui, T. (1992). Partitioning an assembly for infinitesimal motions in translation and rotation. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pages 1311–1318.

[Xin et al., 2011] Xin, S.-Q., Lai, C.-F., Fu, C.-W., Wong, T.-T., He, Y., and Cohen-Or, D. (2011). Making burr puzzles from 3D models. *ACM Trans. on Graph. (SIGGRAPH)*, 30(4):97:1–97:8.

[Yao et al., 2017a] Yao, J., Kaufman, D. M., Gingold, Y., and Agrawala, M. (2017a). Interactive design and stability analysis of decorative joinery for furniture. *ACM Trans. on Graph.*, 36(2):20:1–20:16.

[Yao et al., 2017b] Yao, M., Chen, Z., Xu, W., and Wang, H. (2017b). Modeling, evaluation and optimization of interlocking shell pieces. *Comp. Graph. Forum (Pacific Graphics)*, 36(7):1–13.

[Zessin, 2012] Zessin, J. F. (2012). *Collapse Analysis of Unreinforced Masonry Domes and Curving Walls*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA.

[Zhang et al., 2020] Zhang, X., Belfer, R., Kry, P. G., and Vouga, E. (2020). C-space tunnel discovery for puzzle path planning. *ACM Trans. on Graph. (SIGGRAPH)*, 39(4):104:1–104:14.

[Zhang and Balkcom, 2016] Zhang, Y. and Balkcom, D. (2016). Interlocking structure assembly with voxels. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pages 2173–2180.

[Zhang et al., 2018] Zhang, Y., Whiting, E., and Balkcom, D. (2018). Assembling and disassembling planar structures with divisible and atomic components. *IEEE Transactions on Automation Science and Engineering*, 15(3):945–954.

[Zheng et al., 2017] Zheng, C., Do, E. Y.-L., and Budd, J. (2017). Joinery: Parametric joint generation for laser cut assemblies. In *Proc. ACM SIGCHI Conference on Creativity and Cognition*, pages 63–74.

Ziqi Wang | Curriculum Vitae

BC346, EPFL – 1015 Lausanne – Switzerland

+41-21-693-7533 • ziqi.wang@epfl.ch • kiki007.github.io

Education

EPFL **Lausanne**
PhD Candidate, Switzerland *2017 - 2021.12(Expected)*
Geometric Computing Laboratory
School of Computer and Communication Sciences
Advisor: Prof.Dr.Mark Pauly (EPFL, Switzerland)
Co-Adviosr: Prof.Dr.Peng Song (SUTD, Singapore)

University of Science and Technology of China **Hefei**
Bachelor, China *2013 - 2017*
Information & Computational Science
Department of Mathematics
Rank: 9/145 (Top 6%)

Publications

[1] **Ziqi Wang**, Peng Song, and Mark Pauly. Mocca: Modeling and optimizing cone-joints for complex assemblies. *ACM Transactions on Graphics (SIGGRAPH 2021)*, 2021.

[2] **Ziqi Wang**, Peng Song, and Mark Pauly. State of the art on computational design of assemblies with rigid parts. *Computer Graphics Forum (Proc. of Eurographics)*, 2021.

[3] Yang Xu, **Ziqi Wang**, Siyu Gong, and Yong Chen. Reusable support for additive manufacturing. *Additive Manufacturing*, 39:101840, 2021.

[4] **Ziqi Wang**, Peng Song, Florin Isvoranu, and Mark Pauly. Design and structural optimization of topological interlocking assemblies. *ACM Transactions on Graphics (SIGGRAPH Asia 2019)*, 38(6), 2019.

[5] **Ziqi Wang**, Peng Song, and Mark Pauly. DESIA: A general framework for designing interlocking assemblies. *ACM Transactions on Graphics (SIGGRAPH Asia 2018)*, 37(6), 2018. Article No. 191.

[6] **Ziqi Wang**, Jack Szu-Shen Chen, Jimin Joy, and Hsi-Yung Feng. Machined sharp edge restoration for triangle mesh workpiece models derived from grid-based machining simulation. *Computer-Aided Design and Applications*, 15(6):905–915, 2018.

[7] Peng Song, Bailin Deng, **Ziqi Wang**, Zhichao Dong, Wei Li, Chi-Wing Fu, and Ligang Liu. CofiFab: Coarse-to-fine fabrication of large 3d objects. *ACM Transactions on Graphics (SIGGRAPH 2016)*, 35(4), 2016. Article 45.

Professional experience

EPFL **Lausanne**
Teaching Assistant, Switzerland *Sep 2017 - present*
MATH-111(E) Linear Algebra (Fall 2020)
CS-341 Introduction to Computer Graphics (Spring 2019, 2020)
CS-446 Digital 3D Geometry Processing (Fall 2018, 2019)
CS-457 Geometric Computing (Fall 2021)

ETH Zurich **Zurich**
Academic Visiting, Switzerland *2021 Summer*
Gramazio Kohler Research

University of Southern California **Los Angeles**
Academic Visiting, USA *2017 Spring*
Host: Prof.Dr.Yong Chen
Project for designing supports-free 3D FDM printer.

The University of British Columbia **Vancouver**
Research Assistant, Canada *2016 Summer*
Host: Prof.Dr.Hsi-Yung Feng
Worked on topics in CNC machining simulation.

Talks

DESIA: A General Framework for Designing Interlocking Assemblies (with Peng Song) *2018.12*
ACM SIGGRAPH Asia

Design and Structural Optimization of Topological Interlocking Assemblies *2019.12*
ACM SIGGRAPH Asia

Computational Assembly for Fabrication: Shape Optimization *2021.3*
Computational Fabrication Seminar
Invited by Peng Song

State of the Art on Computational Design of Assemblies with Rigid Parts *2021.5*
Eurographics State of The Art Report

MOCCA: Modeling and Optimizing Cone-joints for Complex Assemblies *2021.8*
ACM SIGGRAPH

Professional service

- Reviewer, TVCG, TOG
- Reviewer, Computer Aided Geometric Design

Professional skills

Programming: C/C++, Python, C#

Software: Rhino/Grasshopper

Language: Chinese(native), English(fluent), Japanese(beginner)

Honors

2016: The Baogang Scholarship, top 5%

2015: USTC Outstanding Student Scholarship(Grade 1), top 10%