Falling into Place: Drop Assembly of Interlocking Puzzles

Author Names Omitted for Anonymous Review. Paper-ID [add your ID here]

Abstract— **This paper explores a system for assembling structures by dropping block components into place. During and after assembly, the blocks are held together by geometric interlock, so that fasteners or mortar are only needed to bind the final block to one of its neighbors. Drop assembly is a promising strategy for assembly by swimming or flying robots, as it may allow structures to be built without requiring close contact with the existing structure. The current paper explores a mathematical model of interlock, and presents a particular block design that allows interlock to be achieved using only gravity. Proof-of-concept demonstrations of the system are presented using a low-cost and relatively low-precision robot arm. The paper finally analyses some of the potential limitations of the approach, particularly including flexing of the structure due to manufacturing tolerance limitations.**

I. Introduction

An assembly of rigid bodies is *interlocking* if there are no possible motions of any collection of the bodies relative to the other bodies; the entire assembly moves only as a single rigid body. Jigsaw puzzles are typically interlocked *in the plane*, but are assembled from above. Burr puzzles are *nearly interlocked* in space – there is a single assembly order, and if the last piece to be assembled is glued to a prior piece, the puzzle is completely interlocked.

This paper shows that some nearly-interlocked planar and 3D structures can be built by an extremely simple method: dropping each piece and allowing gravity to pull the piece into place.

This work is motivated by the eventual prospect of constructing structures with aerial drones, or underwater with submersible vehicles. Flying or swimming robots permit great ease of access to the structure being built, but lack of a fixed base places great demands on the control of the robot, so that neither the robot nor the component being placed crashes into the existing structure. The goal of the present work is to design components that can be dropped into place, so that the robot need not approach the existing structure too closely. *Drop assembly* also motivates a study of compliance – funnel-like joints in the existing structure can remove error and allow imperfectly positioned blocks to slide into place.

It is surprising to us that an interlocking structure can be built using only gravity; imagine an interlocking planar jigsaw puzzle that can be assembled without lifting any pieces off the table, and using only forces from one direction. There are two simple ideas to the approach: 1) prismatic joints allow motion in only one direction, and conflicts between those directions can jam or interlock an assembly, and 2) angled joints can serve as ramps that convert gravitational force in one direction to motion in another direction. Section III builds the mathematical model of *nearly interlocking* structures constructed using only prismatic joints.

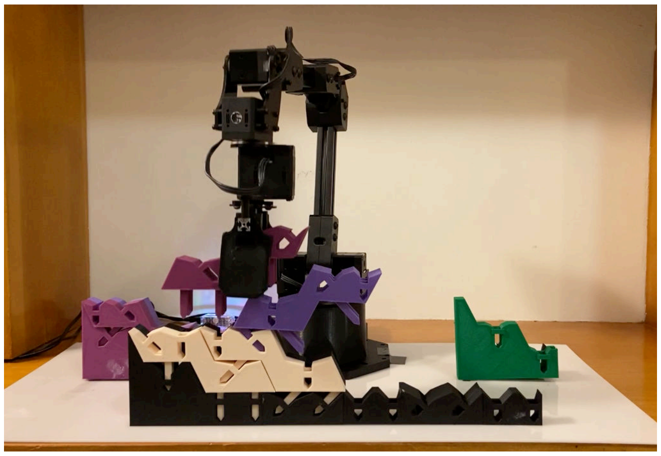


Fig. 1: Robot arm assembling interlocking blocks.

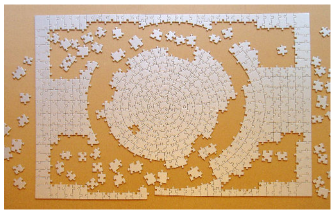
Section IV shows a particular design of two types of blocks that allow drop assembly of nearly interlocking structures, by adding angled ramps to the simplified blocks from the prior section. These blocks may be laid out in a standard brick pattern, allowing interesting shapes of structures to be built without the need for cement, glue, or other fasteners except on the last block placed – see Figure 1.

Interlocking blocks have some advantages relative to other construction techniques. Relative to 'harden in place' approaches used in large- and small-scale 3D printing, the component-based design may allow disassembly for re-use or repair. Components may be fabricated off-site and may be heterogeneous, containing reinforcing materials or embedded electronics. LEGO blocks are component-based and provide some of the same advantages of our design, but the friction locks that connect LEGOs are delicate and require high manufacturing precision as well as fairly precise assembly.

Interlocking blocks also have potential disadvantages. Mortarless assembly can leave small gaps between blocks, leading to undesired aggregate flexibility of the structure. Section VI explores some of these potential weaknesses, using a linearized representation of the configuration space to explore flexing and variation of structure shape based on this local motion of the components.

II. Related Work

The work presented in this paper is related to work done in interlocking, robotic assembly, self-assembly, and modular robotics.


(a) Source: Andreas Roever [23]

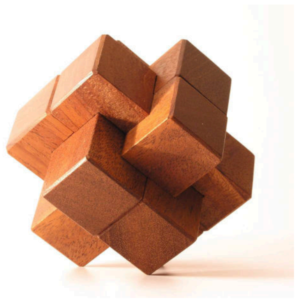

(b) Source: Muns [18]

Fig. 2: Jigsaw puzzles and burr puzzles are examples of interlocking structures.

A. *Interlock*

Interlocking pieces play a large part in puzzles and assembly – e.g., jigsaw and burr puzzles (Fig. 2). Snoeyink and Stolfi [29] give configurations that cannot be taken apart with two hands. Czyzowicz, Stojmenovic, and Urrutia [6] prove that polygons with no parallel edges can be immobilized with three points. More recently, Xin et al. [37] decompose 3D models into Burr Puzzles with one mobile part called a key; the puzzles can only be disassembled by removing the key. Similarly, Song, Fu, and Cohen-Or [31] generate designs for interlocking structures that are incrementally interlocked as pieces are added to the structure, which also ensures structures can only be disassembled in one direction. Generally, 3D printing large structures is challenging. Song et al. [32] addressed such challenges by decomposing structures into smaller pieces which are strongly connected, but can still be assembled or disassembled. Fu et al. [10] showed larger objects, such as furniture, can be decomposed into overlapping interlocking subsections that can be re-assembled as an interlocking union of the subsections. In [14], Lensgraf et al. provide a means of determining the free motions of a structure, which aids in proving a structure is interlocked.

This work is closest in spirit to work done in [40] and [39]. Our work uses a similar definition of a key from [40], but we provide our own definition of interlock and provide a systematic way for proving whether shapes are interlocked.

Caging is a form of interlock; Rimon and Blake [19] formalized the notion of a caging set, where an object is completely surrounded by two fingers and has some freedom to move, but cannot escape the fingers. This work was extended to planar bodies [20] and to three fingers [7]. Rodríguez and Mason established a distinction between stretching caging and squeezing caging [22] and then later extended the idea to grasping [21].

B. *Robotic Assembly*

Drone Assembly is becoming more common. Two quadrocopters used foam blocks and adhesive to build a 6m tower in [2]. Magnets and block geometry can passively aid drone assembly [13, 9]. Using drones for assembly poses new constraints on weight and size of building blocks; Willmann et al. [35] use these constraints as the guiding force behind block fabrication and design. Beyond new block design restrictions, there are also constraints on the types of structures that can be built by drones alone [27]. Some of these restrictions can be overcome by adding features to the building materials such as magnets [15] or making the materials very lightweight [2, 4].

Nonprehensile Manipulation has been explored for some manipulation tasks. Moll and Erdmann [17] explored nonprehensile manipulation to infer the shape of an object via tactile sensors. Woodruff and Lynch [36] studied motion planning and feedback control of nonprehensile manipulation using sequences of motion primitives. Ryu, Ruggiero, and Lynch [26] stabilized an object in an upright position when either the base the object is balanced on or the object itself rotates to a specific orientation. Dogar and Srinivasa [8] determined feasible actions based on the mechanics of pushing. Mason [16] provided a survey of recent work in nonprehensile manipulation, including a study of using dynamics to manipulate an object. Geomans and Stappen [11] used V-shaped traps in vibratory tracks and passive mechanical compliance to sensorlessly orient parts.

C. *Self Assembly and Modular Robotics*

Self Assembly Intelligent building blocks can aid robotic assembly [33, 34]. Zhong Li, Balkcom, and Dollar [41] propose a method of discretizing a large, planar shape into connected triangles to cover shapes. Reconstructing shapes in the plane is possible using simple self-assemble-able robots [1], and more complex, pre-programmed shapes can be formed by a swarm of modular robots [25]. Drones are computationally designed in [9] given a collection of components.

Modular Robotics Some modular robotic systems have self-assembly properties [28, 24, 12] that enable small, programmable robots to form larger structures or to change shape for different tasks. Yim et al. [38] provide a survey of current modular robotics systems and challenges. Modular robots can themselves be fabricated using materials such as foldable sheets of polyester and laser cut parts [30] and be used for simple tasks. In [3], a single input signal is used to control a large swarm of modular robots. Our goal is somewhat related; we use a single direction of motion to attach our blocks to a structure.

III. Interlocking Model

Figure 3 shows some example blocks that we will use to illustrate interlock. We introduce our definition of interlock, and build a set of constraint equations that may be used to test if a structure is interlocked.

A. *Types of Blocks*

First, we show a system using square blocks with 60° and 30° constraints; these simpler blocks illustrate the process of interlock, but are not particularly reliable for drop assembly, which motivated a different design in the following sections.

We denote Type A blocks as having three joints at 60° and Type B blocks has having three joints at 45°, as

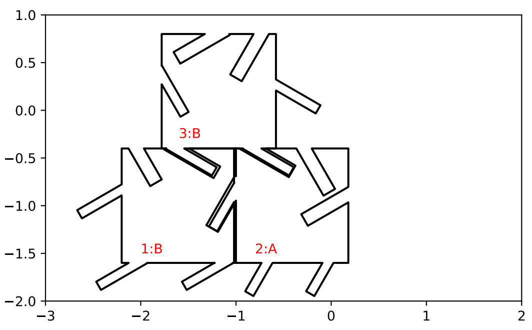


Fig. 3: Example configuration with 3 blocks. The blocks are labeled with the assembly number and the block type.

shown in Figure 3. Blocks connect via prismatic joints into complementary holes. Bottom joints spike two blocks in the previous layer, "stapling" the blocks together. Side joints connect blocks in the same layer.

We consider first an idealized model of the system in which the connections between each pair of blocks is a perfect prismatic joint, allowing linear translation but no rotation.

The last block inserted will only be spiked by one type of joint, so for our physical implementation we introduce an external constraint that glues together the last two blocks after insertion.

B. Constraints

For any pair of touching blocks connected by some joint, the general form of the constraint is

$$\mathbf{n_{i,j,k}} \cdot (x_i - x_j, y_i - y_j, z_i - z_j) = \mathbf{n_{i,j,k}} \cdot (\hat{x}_i - \hat{x}_j, \hat{y}_i - \hat{y}_j, \hat{z}_i, \hat{z}_j), \tag{1}$$

where constants $\hat{x}$, $\hat{y}$, and $\hat{z}$ denote the initial coordinates, variables x, y, and z denote the current coordinates, and $\mathbf{n_{i,j,k}}$ denotes the unit normal to the line of action of the joint.

We choose an arbitrary block as block 1, which will be the reference frame for all other blocks in the system. Although the entire system acts as a rigid body and can rotate, each block is connected through a chain of prismatic joints to block 1, and cannot rotate with respect to any other block. We therefore omit θ coordinates for the blocks from the following analysis.

We can define the constraints for systems like the one shown in Figure 3 and for any arbitrary system using this form.

To express the constraint on the relative motion of blocks, we take a time derivative of each constraint equation:

$$\mathbf{n_{i,j,k}} \cdot (\dot{x_i} - \dot{x_j}, \dot{y_i} - \dot{y_j}, \dot{z_i} - \dot{z_j}) = 0 \tag{2}$$

For any particular system, all the constraints can be written in matrix form as

$$J \cdot \dot{\mathbf{x}} = 0 \tag{3}$$

where J is the constraint Jacobian matrix of constraint equations and $\mathbf{x}$ is the vector of block locations. Note that since only pure translations are permitted by the prismatic joints, the elements of J are constants. For the first block, we add two additional rows to J: $\dot{x_1} = 0$, $\dot{y_1} = 0$, $\dot{z_1} = 0$.

The null space of J thus gives the free motions $\dot{\mathbf{x}}$. This suggests a way to check if a system is interlocked: a system is interlocked if and only if the null space of the constraint Jacobian is empty.

As an example, consider the three blocks $b \in \{b_1, b_2, b_3\}$ shown in Figure 3. The normal for the line of action of b_3 is $[\frac{\sqrt{3}}{2}, \frac{-1}{2}]$ and the normal for the line of action of b_2 is $[\frac{\sqrt{3}}{2}, \frac{1}{2}]$. Applying these to Equation 1 and Equation 2, we find a J for the system. The null space of J has a single column $\left(0, 0, 0, 0, 0.5, 0.9\right)^T$. The free motion in this system is produced by b_3, which can detach from b_1 and b_2, as indicated by the entries in the null space. The structure in Figure 3 is not interlocked.

C. Interlocking patterns

Computing the null space of the constraint Jacobian is a *test* to check if a system is interlocked, but we would like to construct large arbitrary structures that are interlocked by design. We use the following approach. First, we construct a small pattern of blocks and use the null-space approach to prove that it is interlocked. If these interlocked subsets overlap with some other interlocked subset and this is done inductively over the structure, we can prove that the entire structure is interlocked.

Before we present our argument for interlocking through overlap, we introduce the notion of a key; keys are not unique to this work and are also presented in [40] and [31]. Because the structure is assembled by translation, the last block can always escape; the structure is not interlocked until the key is bound to the the structure using a fastener of some type.

We will show each subset has a key, and the keys of each subset become incrementally interlocked as the structure is assembled. When we say a structure is *nearly-interlocked*, we are referring to structures that are locked if the key or keys are bound to the structure.

Consider the system in Figure 4, where the set of all blocks is B. Blocks are labeled using a number representing their place in the global assembly order.

For the system to be interlocked, each block $b \in B$ must also be in a smaller set s such that each set $s_1 \cup s_2, ..., s_{n-1} \cup s_n \equiv B$. We define each set s as the smallest subset of blocks that can be nearly-interlocked. In Figure 4, the nearly-interlocked subsets are the two quartets highlighted in red and blue, and a third quartet of blocks {2, 3, 6, 7}.

Consider the blue subset as a standalone structure where block 6 is the key. Similarly, consider the middle subset as a standalone structure where block 7 is the key. We can say both these structures are nearly-interlocked.

Now consider a structure that is the union of the blue subset and the middle subset. We see that block 6 is interlocked as a part of the middle subset, and block 7 becomes the structure's single key.

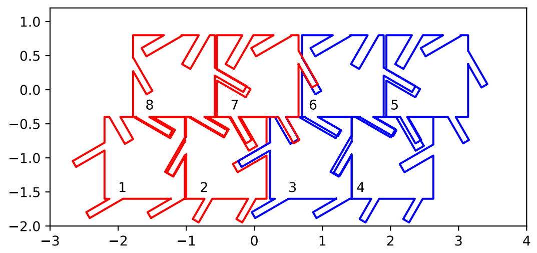
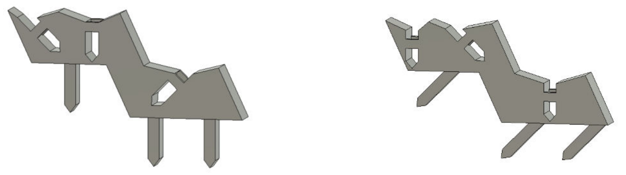


Fig. 4: This structure is *nearly-interlocked*, with block 8 acting as the key.



(a) Type A blocks (b) Type B blocks

Fig. 5: Rectilinear-z blocks for drop assembly.

If we now consider the union of the red subset with the new structure we just defined, we see that block 7 will be interlocked as part of the red subset, and block 8 will become the sole key. If block 8 is glued in, we have a fully interlocked structure composed of three smaller overlapping interlocked structures.

IV. System Implementation

We design physical blocks that have the same interlocking properties as the polygonal blocks discussed in Section III. The experiments in this section are meant to both exemplify the key-interlocking definition we present above, and to show the applicability of this principle to building arbitrary shapes using a collection of only two types of blocks.

A. Block Design

We design rectilinear-z blocks shown in Figure 5. These blocks have the same interlocking properties as their simpler counterparts described in the prior section, but are shaped so as to permit drop assembly.

We also simplified drop assembly by using a vertical spike and reducing the $30°$ constraint to $45°$. The blocks are placed in an ABA pattern to form layers. Each successive layer changes direction so the blocks are built in a snaking pattern, and we design the blocks to be symmetric so a block can be flipped over and used for the next layer. This reduces the number of different blocks needed to build a structure.

B. Experimental Setup

We built small structures with a 5 DoF Trossen Robotics WX-200 robot arm, and we also built larger-scale structures by hand, since the robot has a limited work space. We 3D printed the blocks shown in Figure 5 using an Ultimaker S3.

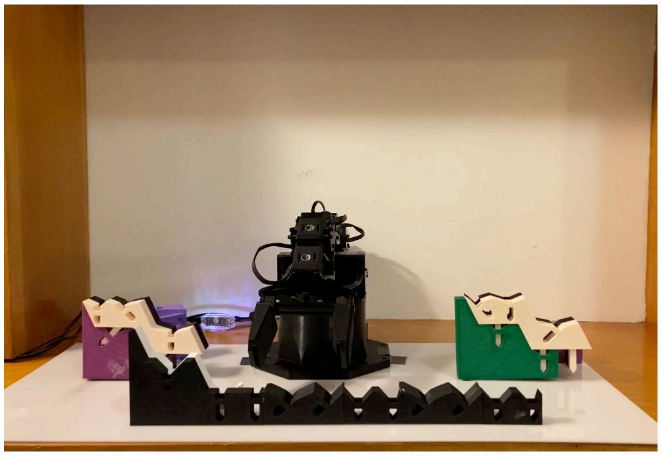


Fig. 6: Robotic assembly setup.

The experimental setup used for assembly with the robot arm is shown in Figure 6. The insertion strategy for blocks is to pick up a block, position over the next spot in the layer, and drop the block. The pickup for blocks of the same type does not change; only the position of the drop-off changes between blocks. The pick up locations are on either side of the robot, and the base where the first layer is placed is directly in front of the robot. Building with the robot arm is meant to be a proof-of-concept of drop assembly to show the possibility of automation. We did not optimize the building setup or the block design for precise pick-ups and drop-offs. There are slight deviations in the the pitch of the block when picked up, but typically error is removed as the dropped block slides into the joint.

V. Drop Assembly

In this section, we show the structures built with the robot arm and by hand. The structures built in this section are planned out manually, and we discuss our plans for an automated layout algorithm in Section VII.

A. Assembly with a Robot Arm

Figure 7 shows a planar wall built with the robot arm. This example shows the ability of the blocks to tolerate overhangs, which we use extensively in the other structures we built by hand. We choose this structure for the robot to build because this five block configuration occurs in the other structures, and so we can see the robot is capable of handling the basic subsets that the structures are composed of.

B. Building General Structures

Planar A The planar A demonstrates the ability to bring two separate structures into one, and it also creates bridges over empty spaces. Additionally, we show that we can build taller structures that don't necessarily need very long bases, and we can remove blocks that are not needed for the interlocking subsets.

The assembled planar A is shown in Figure 8. The key for this structure is labeled, as is the assembly order (numbers).

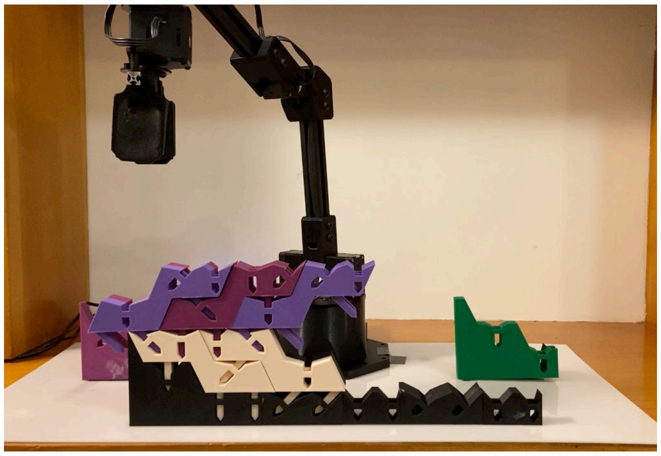


Fig. 7: Planar wall assembled by a robot arm.

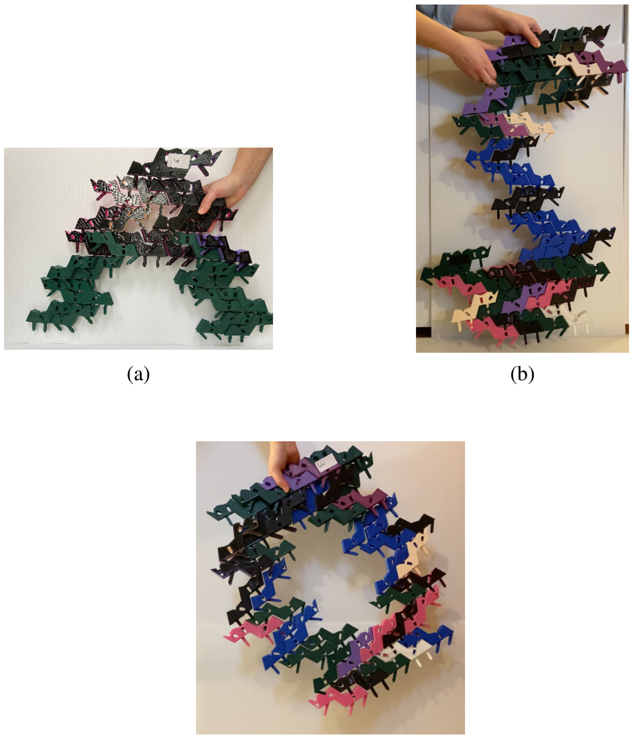


Fig. 9: Structures held without support to demonstrate interlock.

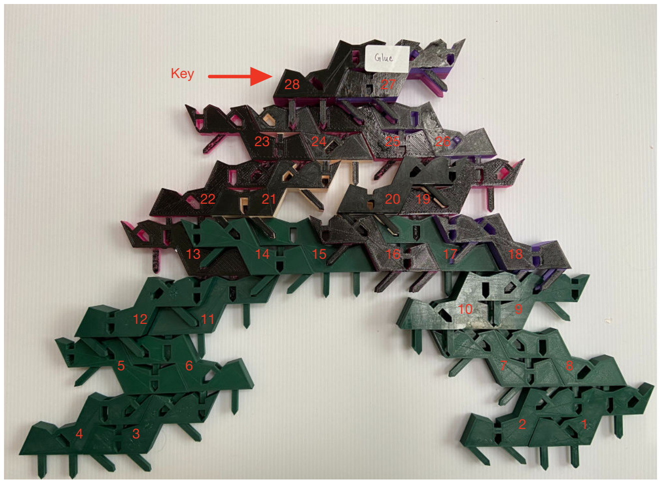


Fig. 8: Planar A with key and global assembly numbers.

This structure is composed of 28 blocks and we build the structure from smaller interlocked subsets. As an example, we see that blocks on the left leg $\{4, 3, 5, 6\}$ are a nearly-interlocked subset with block 6 acting as the key, as is $\{5, 6, 11, 12\}$ with block 12 acting as the key. The other leg of the structure and the main body have similar subsets that inductively interlock to reduce to a single key. We can also verify the structure is locked by picking it up and holding it without any support, as shown in Figure 9a. We can assemble structures with holes by ensuring the blocks around the empty spaces are locally locked without relying on the blocks that could be in those spaces. We are able to achieve two gaps close together by ensuring that block 15 and block 16, which are both adjacent to the gaps, are interlocked with subsets $\{13, 14, 15, 22, 21\}$ and $\{16, 17, 18, 20, 19\}$, respectively.

Planar S The S is shown in Figure 10, with 44 blocks and 2 keys. The first key is the left block on the top layer, and the second key is the left block at the tail of the S. We show this structure is interlocked in Figure 9b by holding the structure away from the back plane and off the ground, but we also start to see flexing of the structure, particularly in the middle of the S. We discuss ways to reduce this flexing by adjusting

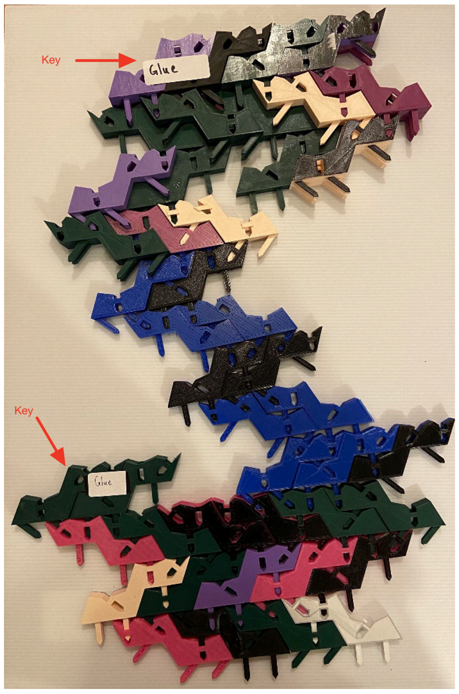


Fig. 10: Planar S with labeled keys.

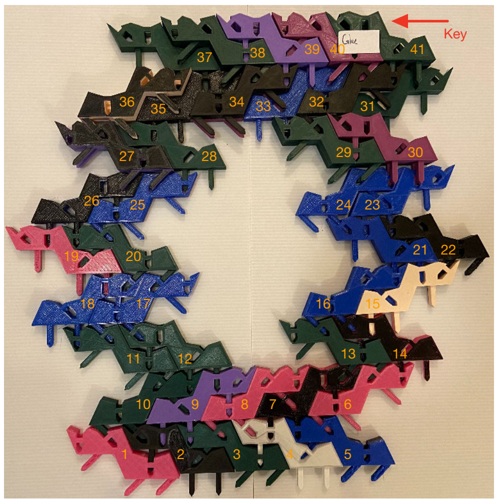
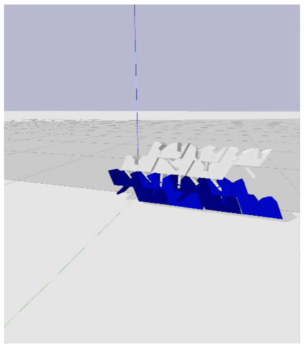
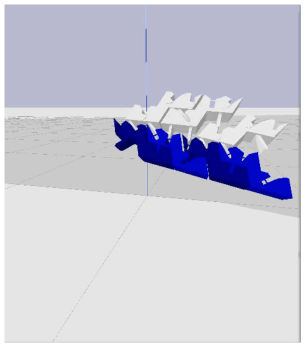


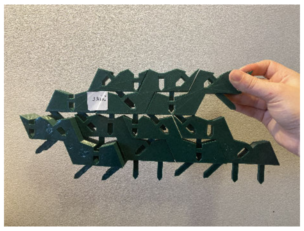
Fig. 11: Wreath with marked keys and assembly order.



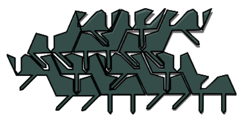
(a) Simulation of seven blocks.



(b) Top layer pulls away from bottom layer.



(c) Physical structure remains interlocked when picked up.



(d) Simulated in PuzzleFlex.

Fig. 12: The simulation in PyBullet produces a failure ((a) and (b)) for a structure that is interlocked in the physical version (c). PuzzleFlex instead correctly determines interlocking (d).

the tolerances of the blocks in Section VI.

Wreath The wreath in Figure 11 has a large hole in the center, but is nonetheless one of the more stable structures we built. We also see, as we saw in the S, that we can start at a common point and build two different threads away from each other. We imagine repeating the wreath to build a chain or a lattice.

VI. Analysis of Limitations and Challenges

One of the primary limitations of the approach is that small gaps between blocks can allow build-up of error throughout the structure, potentially limiting the size and rigidity of assemblies. Initially, we attempted to explore this limitation using the Bullet Physics Engine [5] and PyBullet, but the simulation tool appears to be unsuitable for the task.

An example of the simulated environment is shown in Figure 12a, with an example structure of seven blocks. When the structure is picked up, the top layer begins to pull away from the previous layer (Figure 12b). We show the same example (Figure 12c) using the physical versions of the blocks and find that we do not have the same failures in the physical system as is apparent in the simulated system; the multiple components in close proximity appear to cause the physics engine to permit motion that violates geometric constraints.

A. *Flexing motion in a linearized constraint space*

The PuzzleFlex [14] turns interlocked rigid bodies motions computing to a linear programming problem by modeling joints as linearized constraints between vertices and another block's edge segments. We believe that the framework provided in PuzzleFlex is efficient and suitable for our case, so we will use the PuzzleFlex framework to simulate large scale structures.

PuzzleFlex framework is designed for 2D blocks flexibility analysis, so we focus on analyzing these blocks on the assembly plane. We first project block design in Figure 5 to a polygon on the assembly plane and removed the bridges in holes. The removal of them turns blocks into continuous polygon without holes, which meets with PuzzleFlex input requirement. There is one more thing we need to worry about, our blocks are staggered for assemblability. As discussed in PuzzleFlex [14], these blocks' motion are limited in small convex regions in this case. We make the following modifications to this framework.

B. *Modify optimization framework*

Assume we have a point $p \in \mathbb{R}^2$ on the plane and a polygon, which consist of n vertices

$$V = \{v_i | i \in \mathbb{R}, i \in [0, n], v_i \in \mathbb{R}^2\}$$

is defined as a set collection of line segments

$$G = \{s_i = [v_i, v_j] | v_i, v_j \in V, (j \bmod n) \equiv (i \bmod n + 1)\}$$

Assume function $d(p, s), (\mathbb{R}^2, \mathbb{R}^4) \to \mathbb{R}^+$ always returns the shortest Euclidean distance between point p and line segment s, and let function wind(p, G) return the winding number for point p regarding polygon G. With these, we then can define signed distance between point p and polygon G as

$$g(p, G) = \begin{cases} \min\{d(p, s) | s \in G\}, \text{wind}(p, G) = 0 \\ -\min\{d(p, s) | s \in G\}, \text{wind}(p, G) \neq 0 \end{cases} \quad (4)$$

This function describes the signed distance between a point and a polygon and zero-level set of this function is our polygon boundary.

In this way, we defined a new type of constraint between a point and a polygon. For two polygons G_A and G_B which have configuration $q_A = (x_A, y_A, \theta_A)$ and $q_B = (x_B, y_B, \theta_B)$, assume their transformation matrix are $T(q_A)$ and $T(q_B)$. we can build a constraint between any vertex in A with polygon B and vice versa. For a constraint between a point p_A, which is presented in polygon A's local coordinate, and a polygon B, we denote $p = T_B^{-1} T_A p_A$. Then constraint's Jacobian matrix can be represented as

$$\begin{aligned} J([q_A, q_B]) &= \left[\frac{\partial g(p, G_B)}{\partial q_A}, \frac{\partial g(p, G_B)}{q_B}\right] \\ &= \left[\frac{\partial g(p, G_B)}{\partial p}\frac{\partial p}{\partial q_A}, \frac{\partial g(p, G_B)}{\partial p}\frac{\partial p}{\partial q_B}\right] \\ &= \left[\frac{\partial g}{\partial p} T_B^{-1} \frac{\partial T_A}{\partial q_A} p_A, \frac{\partial g}{\partial p}\frac{\partial T_B^{-1}}{\partial q_B} T_A p_A\right] \end{aligned} \tag{5}$$

Let's denote the objective weight vector by c^T and constraints distance before optimization by d_0. Then we can solve on the same linear programming problem as in PuzzleFlex [14].

$$\begin{aligned} \max_{\Delta \mathbf{q}} \quad & \mathbf{c}^T \Delta \mathbf{q} \\ \text{subject to} \quad & J(q)\Delta \mathbf{q} + \mathbf{d}_0 \geq 0 \end{aligned} \tag{6}$$

After getting the configuration changes $\Delta \mathbf{q}$, we will update blocks configuration at $m+1$ step to $\mathbf{q}^{(m+1)} = \mathbf{q}^{(m)} + t \cdot \Delta \mathbf{q}$, where t is the largest scale value that will not cause any collision in the new configuration.

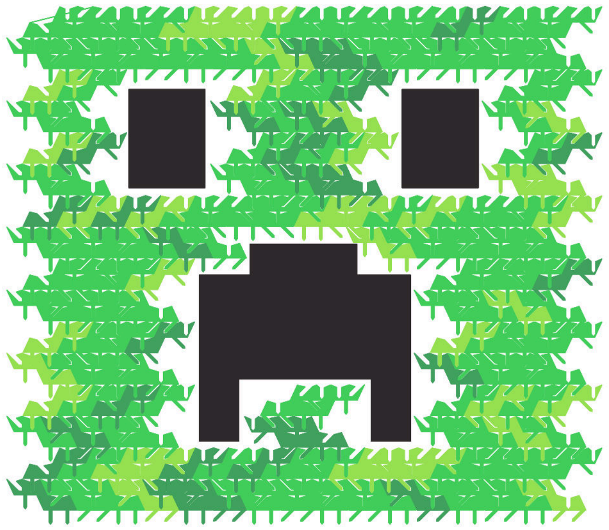
With this framework, we can simplify the block interlocking analysis to the 2D space and analyze it on a larger scale. If the optimization on all blocks leads to no separation between neighboring blocks, we can say the whole structure is interlocked.

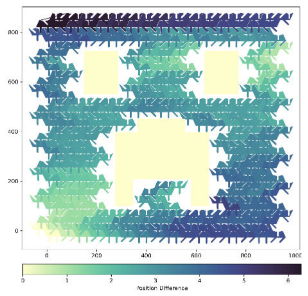
C. Verification with PuzzleFlex

There are several aspects we need to consider to apply this interlocking design to a structure in the real world. We first want to confirm this pattern is repeatable and can still keep its interlocking property in a large scale structure. Secondly, we need to consider the gap between virtual and real world. For blocks in physical world, their dimensions are also affected by tolerance which is controlled by fabrication methods and material. Tolerance might greatly affect our final interlocking status. More details will be discussed in Sec. VI-D We expect to get some estimations of these problems out of the simulation.

We first examined this modified PuzzleFlex method on the 7 interlocked blocks showed in Fig. 12c. The Result is displayed in Fig. 12d. It turns out that PuzzleFlex stopped after few iterations and determined this structure as interlocked. This follows our observation in the physical world.

A large scale structure in Fig. 13 is tested by using this discussed framework. We assume all blocks are placed in


(a) Stacked Minecraft Creeper Head.



(b) Blocks position difference between optimized and original position.

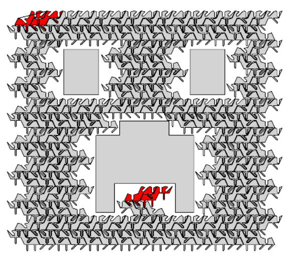
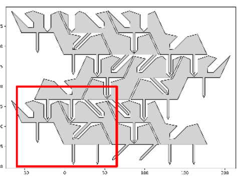
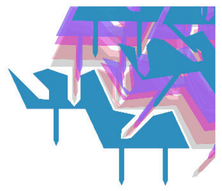

(c) Keys need to be glued on Creeper are showed in red.

Fig. 13: Stacked interlocking blocks create a Minecraft Creeper Head. Bottom left blocks, its eyes and mouth are fixed in position.

position already, and we are examining their interlocking status. In this 16 layers structure, we hollowed three areas intentionally: its eyes and mouth. We placed three fixed black blocks in these areas. The bottom left-most block is also fixed to the ground. Additionally, we set an extra constraint to glue the last two pieces of this structure (top left-most 2 pieces) together. The same constraint is also added to the last two pieces beneath the mouth. The whole structure remains interlocked with no block break up with its neighbors when optimization reached its local optimal solution. This shows that if all blocks are placed in position for a large structure like these, then the structure is interlocked.

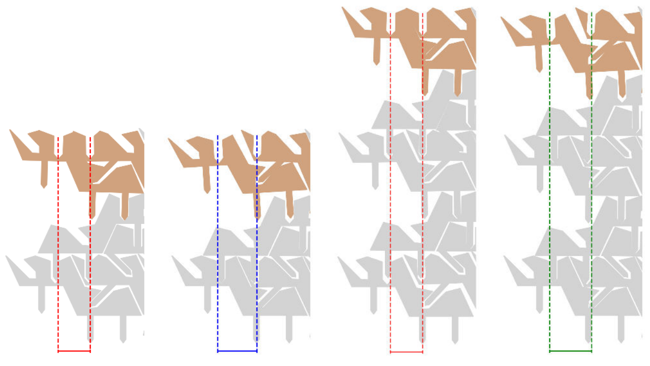
One important property for this interlocking structure is that, for each layer, the last glued two pieces of blocks is the key to connected blocks in the same layer and layers whose keys are covered by these connected blocks. If there is a hole in the structure, connected blocks on one layer will break into more than one group. This will generate more than one key for the whole structure. If we want to keep only one key for the whole structure, we need to guarantee another layer covering them all, like the layer above the creeper's eyes. If there is no such cover layer, like the small spike underneath the creeper's mouth, it will create an extra key

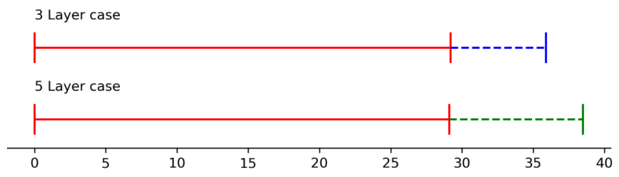

(a) Original nine blocks set up. We mainly focus on disassembly which happens in red rectangle.


(b) Blocks escape process. Key frames with blocks orientation are showed in Fig. 14c-Fig. 14f.

(c) (d) (e) (f)

Fig. 14: Blocks with large gap between peg and hole can escape from the fixed block which is supposed to be fully interlocked with others.


(a) Minimal hole distance in 3 layers (b) Maximal hole distance in 3 layers (c) Minimal hole distance in 5 layers (d) Maximal hole distance in 5 layers



(e) Comparison of minimal and maximal hole distance between different layers structure. Minimal hole distances are showed in solid lines and maximal hole distances are showed in dashed lines. Distance are measured in millimeters.

Fig. 15: Flexibility between blocks changes hole distance between two neighboring blocks. In this case, structure flexibility will increase with layers. And hole distance tolerance will increase with structure flexibility.

for the whole structure. This property constrains the structure to have only closed convex holes or open cave above layer whose keys are not on the cave boundary.

D. *Tolerance Stack-up and future work*

Tolerance impacts greatly on assembly in the physical world. If blocks are designed to be assembled tightly, the dropping assembly process might get jammed. If we have a loose tolerance, unwanted flexibility might cause interlocking failure to be brought into the structure. We showed a failure case in Fig. 14. In this one, large gap between peg and hole grants blocks flexibility. If all this tolerance stack-up, a path is opened in configuration space for other blocks to disassemble from the fixed block.

Tolerance stack-up will also make our assembly process pretty challenging. With the increase of block layers, tolerance will stack-up and increase structure flexibility. We analyzed this flexibility on three layers blocks and five layers blocks. One example is showed in Fig. 15. We use PuzzleFlex to compute the minimal and maximal hole distance between two neighboring blocks. We can see that minimal hole distances are almost the same between these two. However, with tolerance stacking up, the maximal hole distance increased. If a new block is going to be dropped on the top layer, its pegs may not be able to drop in between these holes, or it may get jammed in assembly.

In future work, we need to consider the tolerance impact in assembly. Break tolerance stack-up is important to get a smooth assembly. Glue blocks that pass tolerance to the next layer might be a good feature we can try in future work. More simulation work also needs to be done in identifying locations that accumulate tolerance.

VII. Future Work

This work proposes an interlocking model and a possible block design that implements this model. We first plan to design a layout algorithm that will automatically determine the assembly number of the blocks. In addition to speeding up the assembly process, this will also allow us to optimize the number of blocks needed for a structure, and only use the number required for interlocking.

We also plan to formalize the relationship between peg angle, tolerance, and interlocking. The Type B blocks have a $45°$ joint, which could possibly be increased to reduce the horizontal motion needed to insert the pegs; however, this could reduce the effect of the interlock. We also see from our analysis in Section VI that the tolerances of the blocks can have an effect on interlocking, so we plan to more rigorously study how the design choices of the blocks affect our interlocking model.

Additionally, we show building with the robot arm as a proof of concept, but would like to expand the capabilities of building with the robot arm. This will be aided by the development of the layout algorithm, and by more thoroughly measuring the errors that occur during dropping.

VIII. Acknowledgements

Omitted for double-blind review.

References

[1] DJ Arbuckle and Aristides AG Requicha. “Self-assembly and self-repair of arbitrary shapes by a swarm of reactive robots: algorithms and simulations.” In: *Autonomous Robots* 28.2 (2010), pp. 197–211.

[2] Federico Augugliaro et al. “Cooperative Construction with Flying Machines.” In: *IEEE Control Syst. Mag.* 34.4 (Aug. 2014), pp. 46–64.

[3] Aaron Becker et al. “Massive uniform manipulation: Controlling large populations of simple robots with a common input signal.” In: *Proc. IROS*. IEEE. 2013, pp. 520–527.

[4] Adam Braithwaite et al. “Tensile web construction and perching with nano aerial vehicles.” In: *Robotics Research*. Springer, 2018, pp. 71–88.

[5] Erwin Coumans et al. *Bullet physics library*. URL: `https://bulletphysics.org`.

[6] Jurek Czyzowicz, Ivan Stojmenovic, and Jorge Urrutia. “Immobilizing a shape.” In: *International Journal of Computational Geometry & Applications* 9.02 (1999), pp. 181–206.

[7] C. Davidson and A. Blake. “Caging planar objects with a three-finger one-parameter gripper.” In: *Proc. ICRA*. Vol. 3. 1998, 2722–2727 vol.3.

[8] Mehmet Remzi Dogar and Siddhartha S. Srinivasa. “A Planning Framework for Non-Prehensile Manipulation under Clutter and Uncertainty.” In: *Autonomous Robots* 33 (2012), pp. 217–236.

[9] Tao Du et al. “Computational multicopter design.” In: *ACM Transactions on Graphics (TOG)* 35 (2016), pp. 1–10.

[10] Chi-Wing$*$ Fu et al. “Computational Interlocking Furniture Assembly.” In: *ACM Transactions on Graphics (SIGGRAPH 2015)* 34.4 (2015). $*$ joint first author, 91:1–91:11.

[11] Onno C. Geomans and A. Frank van der Stappen. “On the design of traps for feeding 3D parts on vibratory tracks.” In: *Proc. ICRA* (2008), pp. 385–392.

[12] K. Gilpin, K. Kotay, and D. Rus. “Miche: Modular Shape Formation by Self-Dissasembly.” In: *Proc. ICRA*. 2007, pp. 2241–2247.

[13] Pierre Latteur et al. “Masonry Construction with Drones.” In: *IASS Annual Symposium* (Sept. 2016).

[14] Sam Lensgraf et al. “PuzzleFlex: kinematic motion of chains with loose joints.” In: *Proc. ICRA*. May 2020, pp. 6730–6737.

[15] Quentin Lindsey, Daniel Mellinger, and Vijay Kumar. “Construction with quadrotor teams.” In: *Auton. Robot.* 33.3 (2012), pp. 323–336.

[16] Matthew T. Mason. “Progress in Nonprehensile Manipulation.” In: *The International Journal of Robotics Research* 18.11 (1999), pp. 1129–1141.

[17] Mark Moll and Michael A. Erdmann. “Manipulation of Pose Distributions.” In: *The International Journal of Robotics Research* 21.3 (2002), pp. 277–292.

[18] Muns. *Puzzle Krypt*. 2005. URL: `https://commons.wikimedia.org/wiki/File:Puzzle_Krypt.jpg`.

[19] E. Rimon and A. Blake. “Caging 2D bodies by 1-parameter two-fingered gripping systems.” In: *Proc. ICRA*. Vol. 2. 1996, 1458–1464 vol.2.

[20] Elon Rimon and Andrew Blake. “Caging Planar Bodies by One-Parameter Two-Fingered Gripping Systems.” In: *The International Journal of Robotics Research* 18.3 (1999), pp. 299–318.

[21] Alberto Rodriguez, Matthew T Mason, and Steve Ferry. “From caging to grasping.” In: *The International Journal of Robotics Research* 31.7 (2012), pp. 886–900.

[22] A. Rodríguez and M. T. Mason. “Two Finger Caging: Squeezing and Stretching.” In: *Proc. WAFR*. 2008.

[23] A Roever. *Six Part Wood Knot*. Oct. 2004. URL: `https://commons.wikimedia.org/wiki/File:SixPartWoodKnot.jpg`.

[24] J. W. Romanishin, K. Gilpin, and D. Rus. “M-blocks: Momentum-driven, magnetic modular robots.” In: *Proc. IROS*. 2013, pp. 4288–4295.

[25] Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. “Programmable self-assembly in a thousand-robot swarm.” In: *Science* 345.6198 (2014), pp. 795–799.

[26] J. Ryu, F. Ruggiero, and K. M. Lynch. “Control of Nonprehensile Rolling Manipulation: Balancing a Disk on a Disk.” In: *IEEE Transactions on Robotics* 29.5 (2013), pp. 1152–1161.

[27] Sérgio R Barros dos Santos et al. “Iterative Decentralized Planning for Collective Construction Tasks with Quadrotors.” In: *Journal of Intelligent & Robotic Systems* 90.1-2 (2018), pp. 217–234.

[28] Seung-kook Yun and D. Rus. “Self assembly of modular manipulators with active and passive modules.” In: *Proc. ICRA*. 2008, pp. 1477–1482.

[29] Jack Snoeyink and Jorge Stolfi. “Objects that cannot be taken apart with two hands.” In: *Proceedings of the ninth annual symposium on Computational geometry*. ACM. 1993, pp. 247–256.

[30] Daniel Soltero et al. “A Lightweight Modular 12-DOF Print-and-Fold Hexapod.” In: *Proc. IROS*. Nov. 2013.

[31] P. Song, Chi-Wing Fu, and D. Cohen-Or. “Recursive interlocking puzzles.” In: *ACM Transactions on Graphics (TOG)* 31 (2012), pp. 1–10.

[32] Peng Song et al. “Printing 3D Objects with Interlocking Parts.” In: *Computer Aided Geometric design (Proc. of GMP 2015)* 35-36 (2015), pp. 137–148.

[33] Ken Sugawara and Yohei Doi. “Collective construction of dynamic structure initiated by semi-active blocks.” In: *Proc. IROS*. IEEE. 2015, pp. 428–433.

[34] Justin Werfel et al. “Distributed construction by mobile robots with enhanced building blocks.” In: *Proc. ICRA*. IEEE. 2006, pp. 2787–2794.

[35] Jan Willmann et al. “Aerial robotic construction towards a new field of architectural research.” In: *International Journal of Architectural Computing* 10.3 (2012), pp. 439–459.

[36] J. Z. Woodruff and K. M. Lynch. “Planning and control for dynamic, nonprehensile, and hybrid manipulation tasks.” In: *Proc. ICRA*. 2017, pp. 4066–4073.

[37] Shiqing Xin et al. “Making burr puzzles from 3D models.” In: *ACM Transactions on Graphics (SIGGRAPH 2011 issue)* 30.4 (Aug. 2011), 97:1–97:8.

[38] M. Yim et al. “Modular Self-Reconfigurable Robot Systems [Grand Challenges of Robotics].” In: *IEEE Robotics Automation Magazine* 14.1 (2007), pp. 43–52.

[39] Y. Zhang and D. Balkcom. “Interlocking structure assembly with voxels.” In: *Proc. IROS*. 2016, pp. 2173–2180.

[40] Yinan Zhang and Devin Balkcom. “Interlocking Block Assembly.” In: *Proc. WAFR*. May 2020, pp. 709–726.

[41] Zhong Li, D. J. Balkcom, and A. M. Dollar. “Rigid 2D space-filling folds of unbroken linear chains.” In: *Proc. ICRA*. 2013, pp. 551–557.