Mehrzad Esmaeili Charkhab, Yuxi Liu, Boris Belousov, Jan Peters, Oliver Tessmann

Designing for Robotic (Dis-)Assembly

Abstract: Geometrically interlocking building elements offer unique opportunities for the construction industry to minimize waste, maximize reuse and reduce its carbon footprint. Dry-joint elements allow for fast robotic assembly, disassembly, and reassembly of complex structures out of prefabricated modules. The article discusses the robotic assembly of SL blocks, which are modules that interlock with each other. The assembly process is challenging due to the intricate assembly sequences, design hierarchies, numerous potential grasping points, contact-rich assemblies, and instability until the assembly is completed. To implement the robotic assembly of self-interlocking structures, advances in several research scopes are necessary, including geometry, algorithms, and implementation. The study aimed to co-evolve SL block geometry and robotic grippers using a robot-oriented design approach to compensate for tolerances and add self-centering features. The article also presents an assembly environment that includes mechanical fixation cubes to secure the SL blocks during assembly and support cantilevered sections of the structure until the aggregation is stable. The article presents algorithms for robotic task and motion planning and the generation of assembly sequences inspired by recursive algorithms to design 3D interlocking puzzles and directional blocking graphs. The study highlights how the voxel-based representation of complex geometries can be used to prepare directional blocking graphs. The research provides insights into improving the element's geometry and robotic assembly, which could have significant applications in other studies.

Keywords: topologically interlocking, robotic assembly/disassembly, task planning, robotic-material oriented design, directional blocking graph, SL blocks

1 Introduction

Self-interlocking systems connect multiple components, similar in shape, into stable structures. The goals for architectural constructions made from self-interlocking systems are stability and resistance to compression and bending during assembly and in the final structure. The interlocking of elements is meant to replace any form of adhesives or additional mechanical fasteners. The presented system unfolds complex and multi-directional structural capacity by elements that interlock with multiple neighboring elements simultaneously. This creates a network of interlocking connections that transmits and distributes loads and stresses throughout the entire structure. This research explores not only the structural and tectonic aspects of self-interlocking structures but also their assembly, disassembly, and reassembly through robots. Two

https://doi.org/10.1515/9783111162683-021

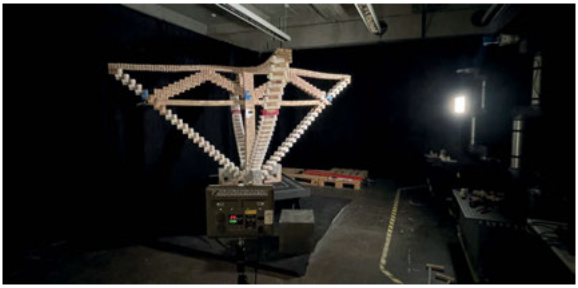

Fig. 1: SL block structure demonstrator

different approaches with opposing points of departure are explored: Pre-programmed, "blind" robots without sensor feedback assemble "smart" elements designed for robotic assembly with self-calibrating shapes and minimum friction during assembly. In contrast, the second approach uses "smart" robots equipped with tactile and force sensing for feedback during the assembly of "stupid" elements with little to no tolerance and high friction during assembly. The goal of this research is to take the best of both approaches: Elements designed for robotic assembly and responsive robots combined in one system in order to evolve automated assembly into the autonomous assembly. By reaching this goal, we seek to contribute to more circular material, and building element flows within architecture and construction. We expect self-interlocking structures to reduce the amount of composite materials systems that are hard to recycle as we reversibly assemble different mono-material elements. Furthermore, we are convinced that shifting computational resources in design from calculating parametrically unique parts to combinatorial explorations of repetitive elements might help to challenge current notions of permanence in architecture, which are focused on single buildings (Touw 2006). Instead, we advocate for a permanence of building elements that can be assembled, disassembled, and reassembled to be used for dynamic reconfiguration of space or in a series of consecutive buildings.

2 Related Work

Robotic assembly, as part of various initiatives in construction automation, promises to increase efficiency in architecture and construction (Bock 2015). Robotic assembly of reversibly joined building elements is explored in various research projects. Fascetti et al. assemble self-calibrating building blocks called a "Drexel with robot arms" in a controlled laboratory environment (Fascetti et al. 2021). Zhang et al.'s present a design for 3D-printed interlocking blocks and an algorithm that allows these blocks to be assembled by a dual-robot system (Zhang et al. 2021). Jenett et al. presents a material-robot system consisting of mobile robots that can assemble discrete cellular structures. The researchers describe “relative” robots that can locomote on, transport, and place elements and navigate relative to and in coordination with a cellular structure with minimal feedback (Jenett et al. 2019). Petersen and colleagues propose the concept of collective embodied intelligence, which draws inspiration from natural collective behavior. This idea involves using individual agents that lack central control but can perform complex and robust behaviors to construct structures successfully. This approach has various benefits, such as avoiding the need for expensive sensors, global communication difficulties, and complicated mechanical movements. The field of automated construction is gaining popularity and features various examples of robot groups, brick-laying robotic arms, and 3D-printed houses. Although each method has its advantages and limitations, a combination of approaches is likely to pave the way for fully automated construction in the future (Petersen et al. 2017). Latteur and colleagues introduce novel research related to the construction of large-scale buildings using aerial vehicles, such as drones (Latteur et al. 2016). Previous work by the authors (Belousov et al. 2022) explores assembling building blocks with an autonomous robotic arm. A combination of reinforcement learning and planning is used to address complex contact dynamics. The first findings indicate that model-free deep reinforcement learning algorithms, which rely on trial-and-error, can be effectively used for this task, even with minimal prior knowledge of the problem.

3 SL block and hierarchical assemblies

The chosen self-interlocking system for this research is called SL block, developed by Shen-Guan Shih from the National Taiwan University of Science and Technology. An SL block consists of one S-shaped and one L-shaped octa cube. Two SL blocks can be joined into six different conjugate pairs called SL-engagements (Shih 2016) (Fig. 2). Aggregated SL-engagements can form a wide variety of forms called SL-Strands (Shih 2018). The use of design hierarchies that emerge from the complexity and number of aggregated SL blocks is used as a strategy for up-scaling aggregations into building elements (Wibranek et. al., 2021). Thus, large structures can be built from many smallelements, which allows for various future (swarm) robotic assembly systems to be developed (Augugliaro et al. 2014; Fascetti 2021; Goessens 2018).

Our approach consists of three steps: Firstly, take a voxelized form as input and divide it into two classes of voxels. These voxels can be selected and combined into a set of planar parts that promises a global interlocking for the desired structure. The joints-design of planar parts is based on three connection types that we defined in the expanded hierarchy: nesting connection, Borromean connection, and joinery connection (Fig. 3). Secondly, voxelize the planar parts into smaller units when designing with “d” and “y” SL-engagements. For instance, combining “d” SL-engagement to create an SL strand within the dark-grey planar part requires constructing a subpart from smaller

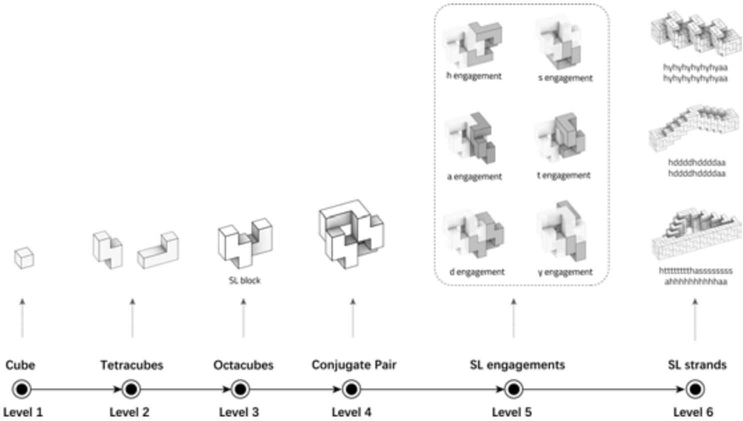

Fig. 2: Hierarchical structure of the SL Block system (based on the hierarchy proposed by Shih).

voxels (see Fig. 4). Thirdly, convert all voxel-based parts into SL block assembly. With the use of the hamiltonian path-finding algorithm, we can compute a closed polyline from non-ordered centroids of voxels. This closed polyline is a sequence of joined vertices, and each vector of vertices can correspond to an SL-engagement. Thus, we can decode the polyline into a string sequence for generating the corresponding SL-strand. Figures 1 and 5 illustrate the interlocking structure we designed and built with SL blocks. In this paper, our focus is solely on one-size voxels, and we are specifically exploring the construction of large-scale structures through hierarchical assemblies of these uniform voxels. Consequently, the design and assembly strategies discussed here are exclusively for this particular size of voxels and the corresponding SL blocks.

SL blocks are self-interlock through topological interlocking but also through friction between the elements (Shih 2020). As friction fit connections are challenging for robots to assemble, we present strategies for improving the SL block geometry for pre-programmed robotic assembly below. The assembly procedures make use of semi-interlocking moments during SL block assembly in which a structure is stable in most directions but still allows for collision-free robotic assembly from one trajectory. These properties are essential for the assembly algorithms described below.

4 SL block geometry

Geometry and robotic grippers were co-adapted with a robot-oriented-design approach (Bock 2015). The pure SL block octa cube geometry was gradually refined. Edges were rounded off to compensate for tolerances of robotic movements as well as imperfections of the aggregated structure. An SL block with fillet edges furthermore self-calibrates when being placed. Additionally, self-centering features and notches in the SL blocks

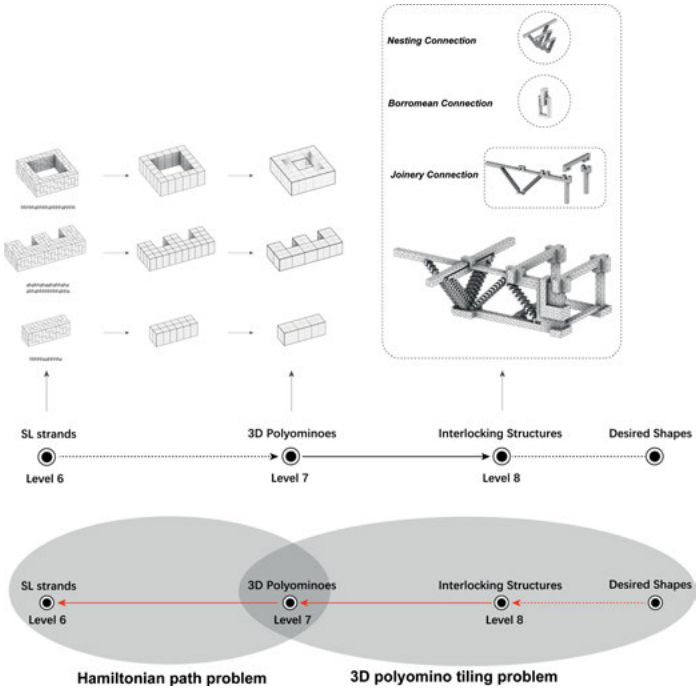

Fig. 3: An expanded hierarchy when using SL blocks to design interlocking structures. Based on a bottom-up analysis of the geometrical hierarchy, we propose two critical Problems that should be solved in the design process according to Top-Down Thinking.

as counterparts of a fork-like robotic gripper were tested (Fig. 6). The conical shape of the notch cross-section calibrates the SL block position during grabbing.

5 SL block fabrication

For the production of the more complex shaped SL block geometry, we tested FDM 3d-printing, laser cutting, and CNC milling (Fig. 7). The production of SL blocks from cross-laminated veneer plywood panels required a decomposition of the SL block into elements that can be CNC milled from sheet material. The geometry of the self-centering notches was adjusted to the tools of the CNC router. To avoid gluing pieces together, we created parts that could be friction-fit to form an SL block using tongue and groove joinery (Fig. 8). These parts can be fabricated from sheet materials with varying thicknesses. Using thinner wooden sheets can create a lighter structure creating porosities and carrying the potential to respond to varying loads and stresses in a structure.

Other SL blocks were cast from concrete in two different ways: collapsible molds and permanent formwork. Doyle et al. (2019) suggest using custom dissolvable form-

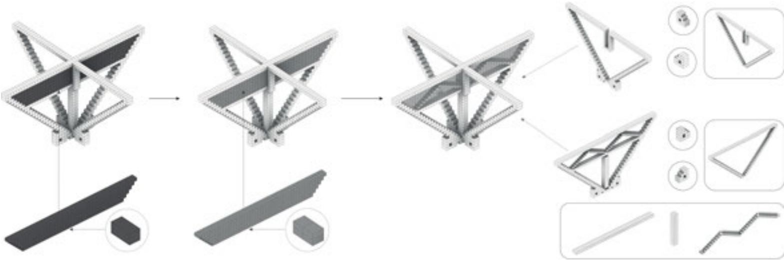

Fig. 4: Redesign of the planar part from smaller voxels. To automate the reconfiguration of the voxel-based part using d SL-engagement, we should further subdivide the dark-grey voxel into four units and then design a subpart from these smaller voxels.

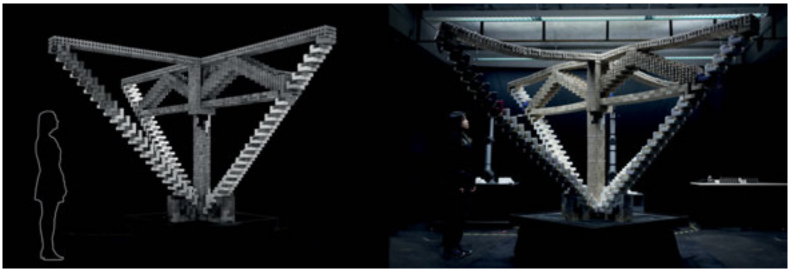

Fig. 5: Left: Rendering of the cantilever structure made from 1030 SL blocks. Right: The physical prototype assembled from concrete, veneer plywood, and 3d-printed SL blocks.

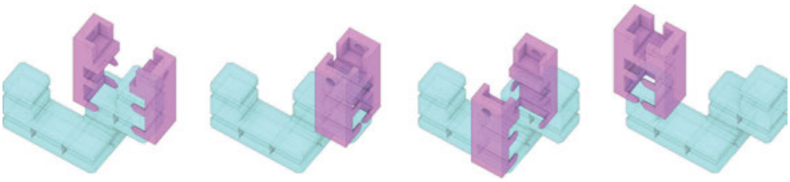

Fig. 6: SL block geometry with self-centering notches and counterpart fork-like gripper.

work and flexible steel reinforcement with traditional concrete construction methods. They propose an experimental approach to integrating these materials and techniques in concrete casting. We tested 3D printing molds using water-soluble filaments for collapsible molds and achieved precise SL blocks in a reliable process (Fig. 9); however, every 3d-printed mold can only be used once, which contradicts the serial character of the SL blocks. Using the 3d-printed hull as permanent formwork creates a thin layer of plastic around the concrete core and between the discrete elements. This makes the assembly of elements easier and protects the brittle concrete during assembly and disassembly. Such a layer might improve durability but leads to undesirable material composites.

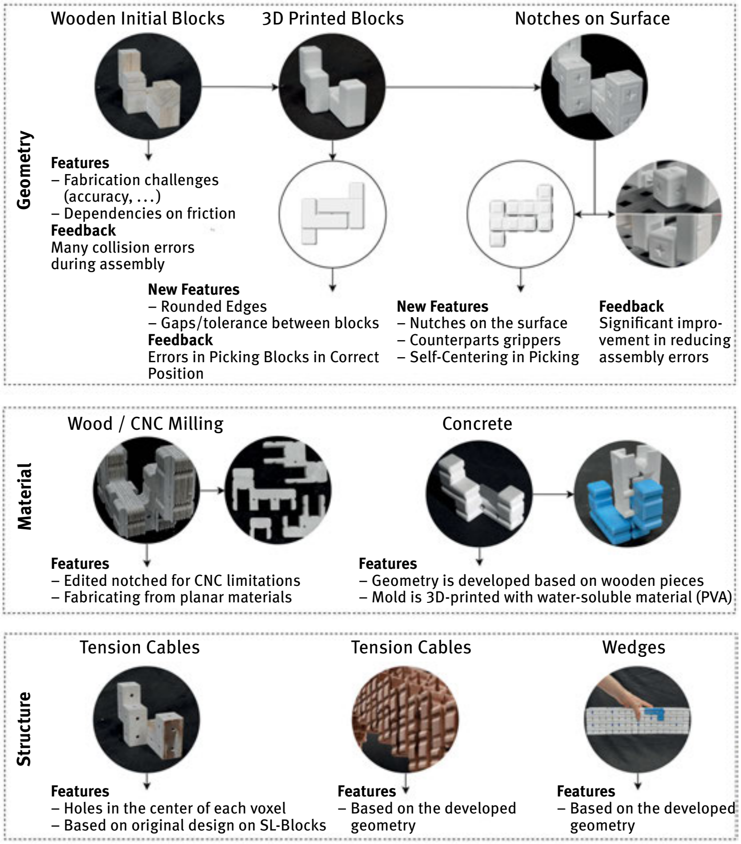

Fig. 7: The diagram depicts how the SL block undergoes a geometric evolution process. The first step involves testing 3D-printed SL blocks to overcome any fabrication difficulties. After achieving the desired shape for robotic assembly, alternative materials were investigated for production purposes. Finally, multiple structural enhancements were experimented with.

6 Robotic Assembly

The assembly process of complex self-interlocking small-scale elements into larger structures is time-consuming, and SL block aggregations are complicated due to the intricate nature of the various assembly sequences, design hierarchies, and the sheer number of elements. SL block assemblies limit movement between blocks in certain directions but allow for it in others (Fig. 10). The degree of freedom depends on the complexity of the aggregation and the interlocking mechanism that could range from mere surface contact to rotational and recursive procedures of connecting elements. For robotic assembly, stability through interlocking and ease of assembly must be balanced during the process. In certain SL engagements, *x* and *y* directions are interlocked while

Fig. 8: SL block CNC milled from cross-laminated veneer plywood elements and a demonstrator element made from planar material of varying thicknesses.

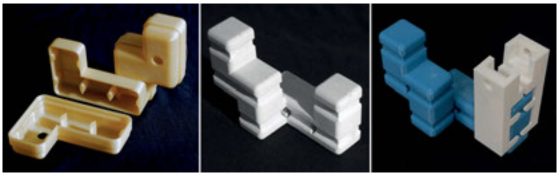

Fig. 9: The concrete SL block is produced using a 3D printed mold made of water-soluble (PVA) material, and the function of the designed fork-like gripper.

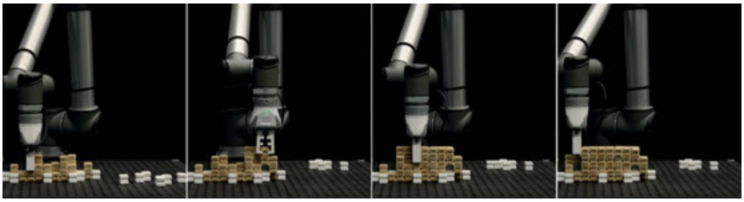

Fig. 10: UR10 robotic arm in the assembly process.

constraining the z-axis exclusively relies on friction. Such contact-rich connections pose challenges for assembly. In response, the above described geometric modifications were made to the SL blocks.

7 Assembly Environment Setup

An additional challenge posed by the strong contact between the elements during assembly is the risk of movement of already aggregated SL blocks during the placement of an additional block. Such an unintended and unregistered movement can hinder the placement of additional blocks. To address this issue, mechanical fixation cubes for the block placement process were added (Fig. 11). They are similar in size to the cubes SL blocks are made of and can be attached to an assembly plate via tongue and groove connections. The blocks aid in the self-calibration of SL block placement, secure the SL blocks during the assembly process, and support cantilevered sections of the structure until the aggregation is stable through placing key-stone-like SL blocks.

Fig. 11: Assembly environment setup parts. From left to right: Assembly Environment Setup, 2SL-Engagement: hddhddh, Support Blocks, Fixation Blocks.

The robot frequently rearranges those blocks during assembly. The below-described algorithm facilitates this process.

8 Robotic Task and Motion Planning

Robotic task and motion planning determine a sequence of actions and movements a robot should perform to efficiently complete a given task. For robotic assembly, task planning involves determining which parts to assemble, in what order, and how to perform each assembly operation. Task planning in robotic assembly can be divided into two main stages: high-level and low-level planning (Fig. 12).

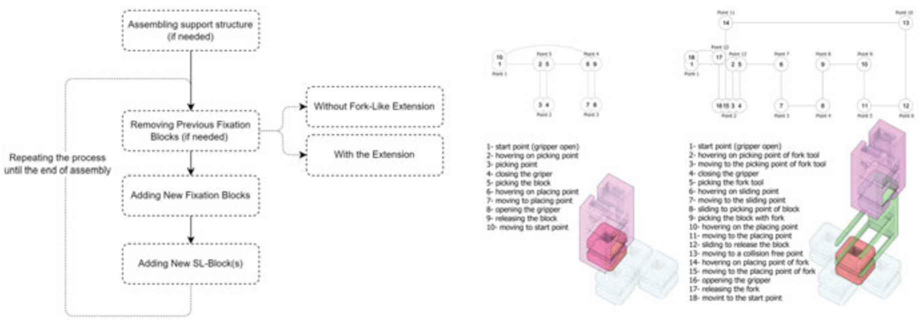

Fig. 12: Left: Robotic task planning in each assembly sequence, Right: General motion planning of the robotic arm with fork-like finger and fork tool extension.

A large number of parts need to be assembled in a specific order and in a tight space without the robots or its end effector colliding with any obstacles or already aggregated parts. Thus, the robot's motion planning algorithm must consider the geometry of the parts and the environment and the constraints of the robot's motion, such as joint limits.

The high-level task planning considers two main scenarios: picking/removing SL blocks, transferring them, and placing them in their intended location. The low-level task planning addresses the proper assembly and disassembly sequences for SL blocks, determining the direction in which they should be assembled or disassembled, finding the gripping points for the SL blocks, and generating arrangements for fixation blocks. Additionally, the algorithm must identify the correct assembly, disassembly, or rearranging sequences for fixation blocks and their optimal gripping points and directions.

Our assembly sequences are inspired by Song's recursive algorithm to design 3D interlocking puzzles (Song 2012) and Wang et al.'s research on identifying disassembly sequences and reversing them in order to find assembly sequences (Wang, 2021). Directional blocking graphs were utilized in several phases, initially to identify the sequence and direction of assembly and subsequently to rearrange fixation blocks and find suitable grasping points that would prevent collisions and enable the blocks to be held in a stable manner, whether it was one SL block, pair of SL blocks or fixation block. We consider an assembly plan as valid if there are no collisions while assembling each part. Assemblies and parts are represented as graphs. We identify removable parts by analysing the directional blocking graph. Each SL block is surrounded by several others limiting its translational degrees of freedom. To numerically represent this situation, every SL block is subdivided into the eight cubes it consists of. These cubes are subsequently converted into lists of x, y, and z-coordinates. These voxel-based geometries interact in a voxelized environment. All data about aggregation geometries and the assembly environment is efficiently stored as point lists. Due to the voxel-based algorithm and the utilization of coordinate comparisons in the x, y, and z coordinates, the computational intensity of the calculations is low, resulting in a relatively fast calculation time.

Every SL block aggregation has two options: either a single SL block can be disassembled, or a pair of adjacent SL blocks can be disassembled together. The algorithm checks these options recursively and prioritizes disassembling single SL blocks.

We name SL blocks that limit the movement of its neighbours "blockees". In order to determine the blockees around each SL block, a list of six possible directions ($\pm z$, $\pm x$, $\pm y$) for assembling the SL block is analysed. The algorithm iterates through each direction and moves the SL block's voxel points by one voxel. If one SL block voxel has the same x, y, and z-coordinates as the voxel of neighbouring obstacles, such as another SL block, fixation, or support blocks, it is considered to not be removable (see Fig. 13). The obstacle coordinates are added to a Blockers list including the index numbers of blockees in each direction. To decrease calculation time, the initial data is calculated and saved initially. Alternatively, voxel points of each SL block can be moved in different directions to create directional checking point lists. Then, each point of each list can be iterated through to check for obstacles in that direction and save the final list. Figure 14 provides visual examples of the Blockees List.

We can determine the sequences and directions for disassembly using a recursive algorithm. In each recursive level, there is a main loop that iterates over each SL block object and another loop in each object that checks directional points. The algorithm compares the X, Y, and Z coordinates of these points to find matches in other blocks or assembly environments. If the coordinates match, it means the block is an obstacle [not movable]; otherwise, it is free to move in that direction. If there is no match, we can

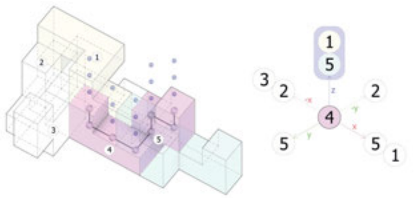

Fig. 13: Creation of part-graphs using voxel representations of SL blocks.

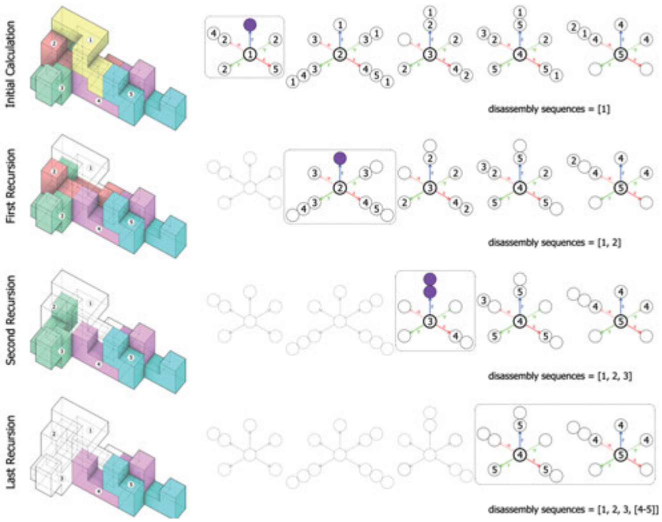

Fig. 14: The recursive disassembly sequence search algorithm relies on directional blocking graphs.

disassemble the block in that direction and add the free block's object and moving direction to the disassembly list while removing it from the main aggregation list. Finally, the list of SL block objects needs updating to remove the index number of the free SL block from all blockees lists, opening up the aggregation space for the next recursion.

When no blocks can be moved individually, it may be necessary to move multiple blocks together. Two pairs of Sl blocks with a specific configuration can be moved together. To identify these pairs, we compare the Sl block index and Blockee indexes in different directions. A pair of movable SL blocks are detected if both SL blocks have a Blockee in the same direction and the index numbers match. These SL blocks are then added to the disassembly list, and the list of SL block objects is updated. This process is repeated recursively until no SL blocks are left in the aggregation list. Finally, by reversing the disassembly list, we obtain the assembly list. This can be seen in the bottom image of Fig. 14, where two blocks can move together in the z-direction.

A procedure has been created to detect the location of fixation blocks for each SL block in assembly sequences to avoid undesired movements during their placement

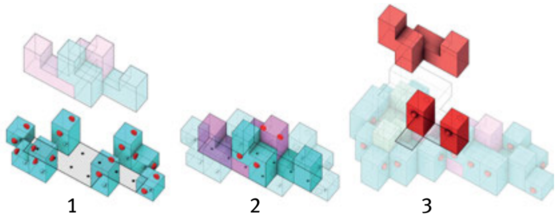

Fig. 15: The diagram illustrates the steps involved in positioning fixation blocks and grasping points, identifying a collision-free grasping point for the SL block and eliminating previous fixation blocks if necessary for the upcoming assembly of the SL block. 1) Finding fixation blocks for each assembly step, Finding gripping points of fixation blocks, 2) Finding collision-free gripping points of SL-Blocks, 3) Finding fixation blocks that need to be removed before the next assembly step (if needed).

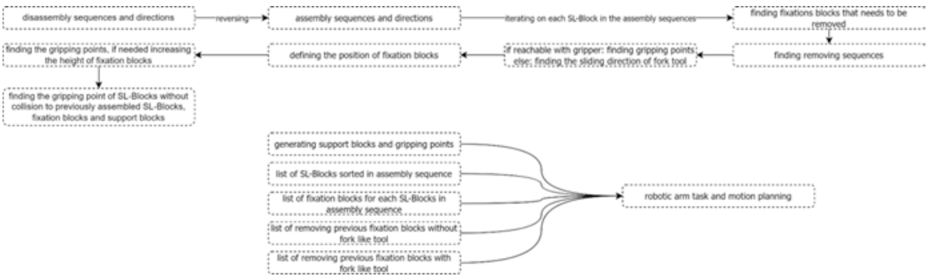

Fig. 16: Algorithmic process following the discovery of assembly sequences.

(Fig. 15, 16). In addition, it's necessary to identify the gripping points of these fixation blocks. The algorithm operates in two stages. Firstly, it recognizes potential locations where support or fixation is needed around the present SL block placement. Then, it checks each location to make sure that there are no collisions between the gripper, the SL block being positioned, and other blocks in the assembly environment. If there is no collision, the gripping point is recorded. However, if there is a collision, the fixation block is extended by including another block on top, and the process repeats. This strategy guarantees that the configuration of fixation blocks with collision-free grasping points can be removable in later steps. Afterward, the algorithm calculates the potential gripping points for the current placement of the SL block, taking into account the position of the fixation blocks. Since there may be various feasible grasping points, it selects the highest one. Finally, a segment of the algorithm confirms if any blocks obstruct the current placement of SL blocks and if so, those blocks are added to the removal list. Next, the algorithm verifies the location of the gripping point for removing the blocks and employs the fork tool extension if necessary (Fig. 12).

The presented research is complemented by the exploration of a system consisting of advanced robotic technology, sensor feedback, and machine learning to control the assembly process (Belousov et al. 2022). To implement a reliable stacking policy, we employ an impedance-based controller (Hogan 1984) in combination with imitation learning (Osa et al. 2018). A human first demonstrates how the SL blocks need to be inserted, and their movement is tracked by an RGB camera Logitech BRIO 4K Pro based on AprilTag markers. Subsequently, the programmatic compliance of the Franka Panda robot arm is leveraged to follow the demonstrated part trajectory. Thanks to the material properties of the SL blocks and the smoothened corners, a reliable insertion policy can thus be learned for each type of connection. Importantly, the approaching movement demonstrated by the human needs to be robust with respect to small perturbations to level out the errors in perception and execution by the robot. However, as long as there is constant contact between the parts, they slide automatically into the desired position. Employing such a robotic compliance controller allows for insertion of SL blocks and SL pairs from top to bottom. Our research aims to merge both approaches and take the most out of robot-oriented elements design and AI-driven robots to contribute to more circular material and building element flows within architecture and construction.

9 Conclusion and Future Work

In our research, we explored design strategies for self-interlocking SL blocks and assembly/disassembly strategies of such aggregated structures. We focussed on the use of pre-programmed robots without sensor feedback that assembled elements designed and optimised for robotic assembly. We regard the combination of dry-joint, reversible structures with automated robotic assembly as necessary for the circular use of building elements. The permanence and durability of elements reused in a series of buildings can contribute to significantly saving material resources and energy. However, economic feasibility might only be achieved through automation. We suggest approaching this challenge with small-scale elements so that large structures can be built from many small elements that will co-evolve with robotic systems to come. Similar to brick dimensions historically evolving in relation to the human agent and a socio-economic context.

We expect self-interlocking structures to reduce the amount of composite materials systems that are hard to recycle as we reversibly assemble different mono-material elements. Furthermore, we are convinced that shifting computational resources in design from calculating parametrically unique parts to combinatorial explorations of repetitive elements might help to challenge current notions of permanence in architecture, which are focused on single buildings (Touw 2006). Instead, we advocate for a permanence of building elements that can be assembled, disassembled, and reassembled to be used for dynamic reconfiguration of space or in a series of consecutive buildings.

Acknowledgments

This research was supported and executed by architecture students at the Digital Design Unit, TU Darmstadt. Among the students involved, Bingqun Liu assisted in the robotic assembly experiments and documentation. We would also like to acknowledge Timm Schneider from IAS TU Darmstadt, who supported our team with the robotic assembly.

Funding for this project was generously provided by Zukunft Bau with project number 10.08.18.7-21.34. The FiF at TU Darmstadt financially supported the initiation of the research.

References

Augugliaro, F., S. Lupashin, M. Hamer, C. Male, M. Hehn, M., M. W. Mueller, J. S. Will, F. Gramazio, M. Kohler, R. D'Andrea. (2014). The flight assembled architecture installation: Cooperative construction with flying machines. *IEEE Control Syst.* 34(4), 46–64.

Bock, T. (2015). The future of construction automation: Technological disruption and the upcoming ubiquity of robotics. *Autom. Constr.* 59, 113–21. DOI: 10.1016/j.autcon.2015.07.022

Fascetti, A., P. Latteur, and S. H. Lim. (2021). Ground-based automated construction of droxel structures: An experimental approach. *Autom. Constr.* 131, 103899.

Goessens, S., C. Mueller, and P. Latteur. (2018). Feasibility study for drone-based masonry construction of real-scale structures. *Autom. Constr.,* 94, 458–80.

Jenett, B., A. Abdel-Rahman, K. Cheung, and N. Gershenfeld. (2019). Material-robot system for assembly of discrete cellular structures. *IEEE Robot. Autom. Lett.* 4(4), 4019–26.

Liu, Y., B. Belousov, N. Funk, G. Chalvatzaki, J. Peters, and O. Tessmann. (2022). Auto (mated) nomous Assembly. In *Trends on Construction in the Digital Era: Proceedings of ISIC 2022* (167–81).

Peng, S., C.-W. Fu, D. Cohen-Or. (2012). Recursive interlocking puzzles. *ACM Trans. Graph.* (31)6, 1-10.

Shih, S. G. (2016). On the Hierarchical Construction of SL blocks – A Generative System That Builds Self-interlocking Structures. In *Advances in Architectural Geometry 2016,* 124–36.

Shih, S. G. (2018). The art and mathematics of self-interlocking SL blocks. *Proceedings of Bridges: Mathematics, Art, Music, Architecture, Education, Culture. 2018.*

Shih, S. G. (2020). Grammars of Interlocking SL blocks. In *Advances in Architectural Geometry 2020.*

Touw, K. (2006). Firmitas re-visited: Permanence in Contemporary Architecture (Master's thesis, University of Waterloo).

Wang, Z., S. Peng, and M. Pauly. (2021). State of the Art on Computational Design of Assemblies with Rigid Parts, *Computer Graphics Forum* 40(2).

Zhang, Y., Y. Koga, and D. Balkcom. (2021). Interlocking Block Assembly With Robots. *IEEE Transactions on Automation Science and Engineering,* 18(3), 902–16.

Hogan, N. (1984). Impedance Control: An Approach to Manipulation. *American Control Conference,* San Diego, CA, USA, 1984, 304–13, DOI: 10.23919/ACC.1984.4788393

Osa, T., J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters. (2018). An algorithmic perspective on imitation learning. *Foundations and Trends in Robotics,* 7(1-2), 1–179.

Doyle, S. E. and E. L. Hunt. (2019). Dissolvable 3D Printed Formwork, 10.52842/conf.acadia.2019.178

Petersen, K. and R. Nagpal. Complex Design by Simple Robots: A Collective Embodied Intelligence Approach to Construction, 05 July 2017, DOI: 10.1002/ad.2194

Latteur, P., S. Goessens, and C. T. Mueller. Masonry construction with drones, Conference: International Association for Shell and Spatial Structures (IASS) Symposium 2016 At: Tokyo

Wibranek, B. and O. Tessmann O. (2021). Interfacing architecture and artificial intelligence, The Routledge Companion to Artificial Intelligence in Architecture, London, p. 486.