CARNEGIE MELLON UNIVERSITY

School of Architecture

College of Fine Arts

Thesis

Submitted in Partial Fulfillment of the requirements for the degree of

Master of Science in Computational Design

TITLE:

Responsive Robotic Assembly with Heterogeneous Raw Wood

AUTHOR:

Jiaying Wei

ACCEPTED BY ADVISORY COMMITTEE:

Joshua Bard	Principal Advisor	May 10, 2023 DATE
Daniel Cardoso Llach	Co-Advisor	May 10, 2023 DATE

Abstract

This thesis reviews the state of art in architectural robotic fabrication, and develops a bi-system framework for responsive assembly with non-standardized raw branches, creating potentials for automation in mass customized fabrication with heterogenous natural materials. The framework synthesizes features of adaptive path planning, material analysis, and parametric design as an efficient and dynamic feedback system to adjust decision-making both on the trajectory planning side and design decision side. Through extensive testing and research, the thesis completed a series of prototypes to verify, improve and evaluate the performance of the system. Apart from prototypes, the results also include a branch-specific design strategy for global shape fitting, a material analysis strategy to quantify and make hierarchical categorization, and optimization for the pick and place procedure in a complicated context. It demonstrates the future potential of robotic automation to directly engage non-standardized natural materials in the field of architectural manufacture and construction.

Table of Content

1. Introduction

2. Background

2.1. Theory and context

2.2. Case studies

3. Hypothesis

4. Methods

Responsive Robotic System

4.1. System overview

4.2. Hardware tool setup

4.3. Design-operation communication module

4.4. Environmental dynamics and collision avoidance

4.5. Data segmentation module

4.6. Data lamination module

4.7. Real-Sim synchronization

4.8. Sim-Real synchronization

4.9. Localized rectification

4.10. Compliance strategy

4.11. Distributed computing

5. Results

5.1. Material Studies in Natural Wood

5.1.1. The node system

5.1.2. The database

5.2. Design Algorithms and Optimization

5.2.1. Modular testing: stacking

5.2.2. Modular testing: 3D positioning

5.2.3. Joinery

5.2.4. Rule-based design system

5.2.5. Work range expansion

5.3. Assembly Prototypes

6. Conclusion

6.1. Evaluation

6.2. Contribution

6.3. Future work

7. Appendix

Introduction

With the development of robotics and computation technologies, architecture studies are actively adopting robotics into complicated construction and fabrication besides design. However, most fabrication processes remain in offline programming and do not assign automatic design and adjustment ability during execution loops. While in the field of autonomous navigation and manipulators, sensor-based intelligence systems have shown their strength in flexibility to initiate human-robot collaboration in transient environments. We begin to question how autonomous fabrication and sensor-based planning inform interaction, design, and construction for architecture. If this adaptiveness is given to robotic fabrication workflows, How can these systems inform design and fabrication with non-standardized materials? In addition, How can automation enhance the potential for mass-customized parametric fabrication? With these questions, this thesis seeks to address the association between dynamic path planning practice and assembly workflow with non-standardized materials to investigate the design of such a framework.

My thesis presents a real-time responsive robotic assembly framework with a ROS-GH-based bi-directional system. The framework orchestrates sensor-based material inputs, geometrical analysis, and dynamic path planning with a parametric design pipeline as a synthesized infrastructure to enable robotic automation with customizable forms by using non-standardized materials. Through data segregation and lamination by using RGB-D SLAM, sampling-based path planning, Octotree, clustering algorithms, Poisson surface reconstruction, and laplacian skeleton contraction with the point cloud. The framework demonstrates distributed computing in collision avoidance, design generation, material analysis, and instance tracking by constructing a path planning scene in ROS and a design planning scene in Grasshopper. The thesis paper describes the incentive of decision-making and the detailed manual for replicating this system to any robot manipulator types and setups.

The particular material chosen to validate this framework is non-standardized raw wood from tree limbs. Each branch instance will be taken in as units to automatically generate a larger structure through means of a rule-based system or agent-based optimization. On a broader scale,

I would like to rethink how designers can transfer knowledge of craftsmanship and design intelligence to robots, to facilitate fabrications for mass-customized modular units to assemble into a greater system. Most of the current robotic fabrication pipelines for complex, parametric forms and designs are simulated without real-time adjustments and sensor feedback. The framework I developed will bridge the parametric design domain with sensor-based feedback path planning so that robots can adapt to a dynamic working environment and non-standardized input data. The framework is partially human-supported for making connections between members and has not yet developed details in the category of joinery. The successful implementation of the framework makes a case of how sensor-based, autonomous robotic systems provide the broad potential for natural raw material in architectural construction to bypass the detour of standardization. The investigation in both the material and the workflow aims to contribute to the blueprint of adaptive architectural fabrication in a great variety of context to support progress in research of architectural robotics.

The document is laid out with the following chapters: background, technical methods, results, and conclusions. In the background chapter, I reviewed the current state of the art about the application of robotics in architecture manufacturing and fabrication, as well as including several case studies about fabrication techniques and design logic with non-standardized wood materials. These precedents provide a foundation for interpreting the behavior of branch materials in this assembly system framework. The method chapter provides a comprehensive description of this adaptive bi-platform operation system, focusing on 1)hardware/software setups, 2) the main algorithm for material and environment, 3) cross-platform communication and synchronization, and 4)adjustments and error minimization. The results chapter contains three main categories of findings: a material processing strategy to quantify, and organize raw tree branches, a workflow development from simple planar composition to complex spatial configuration, and a bottom-up UV-guided design system based on sequential inputs with a series of prototype results. Finally, the thesis will conclude with an evaluation of this system's performance, contributions that support architectural robotics to engage with non-standardized material fabrication in an evolving environment, and possible future work to be integrated into the current system.

Background

Theory and Context

Robot manufacture originates from the need for production lines in handling, bin-picking, welding, and assembly industries, which are mostly repetitive and pre-programmed tasks, outlined by the typical robotic application chapter in *Industrial Robotics*[1]. Its application in fabrication research is the adaptation of the advantages of manufacturing techniques and the shared platform of software between architecture design and industrial design. The interfaces of commercial robots are gradually developing a more user-friendly environment for designers with less expertise in robotics and programming. The most widely industrial brands are ABB, KUKA, and YASKAWA[2]. These robot brands provide high-precision robots for mass production in the manufacturing industry. Customized robotic fabrication is the most popular growing construction methodology for architecture in computational design. Much of the related research work towards minimizing the gap of physical tolerance and geometric complexities like machine learning in timber assembly, and agent-based fabrication of casting units.

Compared to standardized robot operations in commercial settings, modern robotic educational fields, and research primarily focus on autonomous operations. Most of the framework builds on top of ROS(robotic operating system)[3]: an open-source framework for integrating unlimited algorithms and synthesizing them as multi-thread operations with a format restriction. This framework is widely adapted by navigational robots and manipulators. SLAM(simultaneous

---

[1] Martin Hägele et al., "Industrial Robotics," in *Springer Handbook of Robotics*, ed. Bruno Siciliano and Oussama Khatib, Springer Handbooks (Cham: Springer International Publishing, 2016), 1385–1422, https://doi.org/10.1007/978-3-319-32552-1_54.

[2] "World's Top 10 Industrial Robot Manufacturers | Market Research Blog," Text, Market Research Reports® Inc., May 8, 2019, https://www.marketresearchreports.com/blog/2019/05/08/world%E2%80%99s-top-10-industrial-robot-manufacturers.

[3] "ROS Noetic Ninjemys," May 23, 2020, http://wiki.ros.org/noetic.

localization and Mapping)[4] and Octomap[5] in particular, are the foundation for robots to build accurate and lightweight maps for collision avoidance path planning with sensors. Moveit[6], a wrapper software for manipulator operation was developed and first released in 2014, provides versatile APIs and plugins for building customized responsive robotic systems. Its complementary open-source motion planning library(OMPL)[7] carries numerous planners, which serves as the foundation for this research. These planning libraries enable sampling-based planning with both forwards kinematics and inverse kinematics.

Robotic fabrication for architecture design is still a relatively privileged research field, due to its highly-concentrated expertise in robots and requirements for space and funding. Therefore the range of exploration in the use of architectural robotics usually focuses on industrial-ready materials for structurally stable constructions. A variety of post-processed material studies are investigated with robotics to seek potential in prototype and structural designs, including standardized construction materials like timber, alloy, and concrete. Other additive materials include composite fiber, powder, wood filament[8], etc. These materials are generally processed with high coherence in quality as industrial products. In general, the state of the art can be categorized into four main streams:

1. Additive manufacturing: aggregate, soil, and ceramic are the most common materials used for additive manufacturing. It is the most prominent fabrication technique that engages with the commercial industry, with 3D concrete printed houses ready on the market.

[4] H. Durrant-Whyte and T. Bailey, “Simultaneous Localization and Mapping: Part I,” *IEEE Robotics & Automation Magazine* 13, no. 2 (June 2006): 99–110, https://doi.org/10.1109/MRA.2006.1638022.
[5] Armin Hornung et al., “OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees,” *Autonomous Robots* 34, no. 3 (April 2013): 189–206, https://doi.org/10.1007/s10514-012-9321-0.
[6] David T. Coleman et al., “Reducing the Barrier to Entry of Complex Robotic Software: A MoveIt! Case Study,” 2014, https://doi.org/10.6092/JOSER_2014_05_01_P3.
[7] Ioan A. Sucan, Mark Moll, and Lydia E. Kavraki, “The Open Motion Planning Library,” *IEEE Robotics & Automation Magazine* 19, no. 4 (December 2012): 72–82, https://doi.org/10.1109/MRA.2012.2205651.
[8] Philipp Eversmann et al., “Additive Timber Manufacturing: A Novel, Wood-Based Filament and Its Additive Robotic Fabrication Techniques for Large-Scale, Material-Efficient Construction,” *3D Printing and Additive Manufacturing* 9, no. 3 (June 1, 2022): 161–76, https://doi.org/10.1089/3dp.2020.0356.

2. formwork: It is a less popular fabrication process to deal with steel structures that bend and weld into a framework for infill. ETH employed this technique for the prototype of the project Mesh Mould[9].
3. Direct material processing: hard materials like timber usually adopt this technique to deform materials. Steam bending and cutting are applied to anchored materials for the robot tool path. Although large-scale robots provide great precision and strength with this fabrication process, the technology itself is long studied during manual craftsmanship for furniture making[10] .
4. Fiber winding: soft materials without a chemical curing process like composite fiber use this technique to wind repeatedly onto a solid framework as the second layer of the enclosure. It usually does not contain structural strength and its concept is based on the addition from line to surfaces by tension.

The emergence of organic materials in robotic fabrication is less explored due to their high uncertainty in structural properties. But existing projects that fabricate organic material with robots exhibit strong potential for spatial exploration in design as well as sustainability for material processing and transportation, which will be presented in the following case studies. These materials, especially hard fiber materials like lumber and bamboo, are closely integrated into the construction systems in many regions that contain rich resources. I intend to categorize non-uniform organic materials into solid materials and cluster materials. Solid materials like wood logs and bamboo contain strong structural properties in a single unit and are more used as a framework during post-processing. While cluster materials like grain straw, rice husk, and palm leaves require binding agents or stitching/weaving techniques to form structural integrity or be applied as the canopy.

In particular, I would focus on heterogeneous wood logs because 1. each piece of the material unit provides clear-identifiable geometrical identities and does not require excessive post-processing for connections. 2. Wood is a primary construction material, with many

[9] Norman Peter Hack, “Mesh Mould: A Robotically Fabricated Structural Stay-in-Place Formwork System,” application/pdf (ETH Zurich, 2018), https://doi.org/10.3929/ETHZ-B-000263345.
[10] M Gil Pérez et al., “Computational Co-Design Framework for Coreless Wound Fibre–Polymer Composite Structures,” *Journal of Computational Design and Engineering* 9, no. 2 (February 25, 2022): 310–29, https://doi.org/10.1093/jcde/qwab081.

standardized industrial products. However, its origin in organic form endured loss from standardized processing, and most components are wasted due to the nature of geometrical properties. It is not-desired in conventional construction but provides great opportunities in customized robotic fabrication, as listed in the following precedents.

Recently there has been an increasing amount of research geared toward manipulator path planning for architectural fabrication. One of the concentrations leans toward a learning-based approach for localization, some others explore sequential assembly arrangement with collision avoidance.

Precedents and Case Studies

1. The Ashen Cabin[11]

The Ashen Cabin is a project that profoundly influenced my interest in robotic fabrication and material studies. This is a full-scale experimental project that I participated in in 2019. It consisted of a 3D concrete printed base and a top timber framework that has both interior and exterior facades addressed by wood panels sliced directly from wood logs that are acquired locally, and unprocessed. These logs vary in size and shape, yet also create a great potential in reproducing the nature of these variations by extracting their properties but also maintaining a degree of control: the thickness. Logs are sliced, spanning from a coherent solid to oblique curved surfaces, transiting from centric nature to expandable typology. Yet arbitrary decisions in panels retain a gradual proceeding of individual units. I felt that there can be a lot more to investigate how production can be directly affected by the geometries and textures of the raw logs, and it became a spark that inspires my inquiry into the robotic fabrication of non-uniform organic materials.

[11] “Ashen,” HANNAH, accessed March 15, 2023, https://www.hannah-office.org/work/ashen.

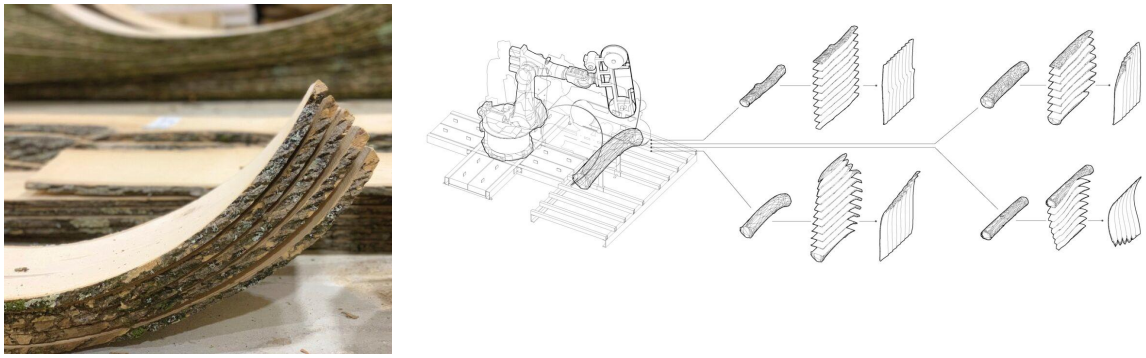


Fig. 1 robotically sliced slabs(left) slicing strategy(right)

2. The Tangential Timber[12] project from After Architecture presented a design system to preserve the individual characteristics of wood log slices while customizing them into modules to assemble into a larger structure. This approach is attractive in the way it rethinks about how heterogeneous materials can be adapted responsively, through a light data organization system, it is implemented and optimized smoothly. However, this entire process is manually driven by designers and does not incorporate any feedback systems from the operating units.



Fig. 2 Mapping trunk sections to a global geometry

3. Wood Chip Barn[13]: The wood chip barn by AA design and make program is an architectural scale construction that connects split trunks and interlinks them into an arch with 4 landing points, these splits utilize one side as an outer spine and the other side as a connection to its counterpart. This non-extendable but coherent structure is strong and interpolative in narrating the relationship between branches.

[12] "Before Building," accessed November 10, 2022, https://before-building.com.
[13] "Wood Chip Barn," *Hooke Park* (blog), accessed March 15, 2023, http://hookepark.aaschool.ac.uk/woodchip-barn/.

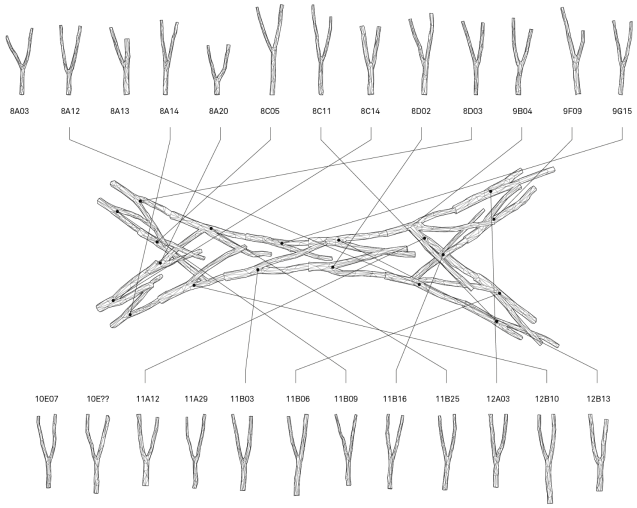


Fig. 3 Bifricated branch assembly strategy from Wood Chip Barn case study[14]

Hypothesis

Foundational for this research is the notion that robotic architectural fabrication can help transfer middle-level design intelligence and craftsmanship to a responsive automation system, which quantifies highly irregular geometry into the organized, hierarchical system for human designers/collaborators to inspect in the top-level blueprint. Our minds as designers can evaluate a general form or typology that exceeds the domain knowledge of robotics and therefore can associate different threads, however, we are also negligent at executing some high-resolution details at the micro-scale. A responsive system plays the role of staircases to bridge the gap between rigid execution to flexible and dynamic decision-making. The common presence of laser cutters, CNC machines, and 3d printers in architecture education fields are robotics that processes lower-level components with high precision. The dimension of materials, valid path, and forces are pre-defined and approved by human operators, and the remaining space for this equipment are left limited to shortest path adjustments and timing. Even in a good design workflow such as end-effector switching, it is supervised and controlled.

[14] “Wood Chip Barn.”

The problem remains: ‘Should we program robotics to adapt to design logic within the complex environment they created for themselves while they are working?’ I assume that the proper way is to build a framework for robots to communicate between their situated context and design interface in real-time with multiple strategies and wrappers to keep the operation running with proper decision-making.

There are three underlying assumptions for the system:

1. The advantage of a sensor-based robotic fabrication system presents its greatest potential while processing with irregular, natural materials.
2. The system is adaptable with different design strategies that work minimally at a modular scale to work with multiple instances (number>3)
3. The system leaves space for human intervention during important evaluation and visualizable phases.

Methods

Responsive Robotic System

1. **System Overview**

   Increasing demands for robotic automation in design fabrication are developed to accommodate the needs for constructing sophisticated structures and handling non-standardized materials. System design and algorithms for incorporating real-time feedback for generative fabrication and autonomous re-adjustments are also seeking engagements. To tackle the technical part of the thesis, a baseline framework for robotic automation is designed to extract material information from sensors and generate a parametric tool path corresponding to the geometric properties of materials inside grasshopper, while executable as online programming and incorporated real-time collision avoidance with SLAM(Simultaneous Localization And Mapping)[15] algorithms. The framework is based on the pipeline developed by COMPAS FAB[16] and took inspiration from the navigation stack to manage path planning similarly. This enclosed

[15] Durrant-Whyte and Bailey, “Simultaneous Localization and Mapping.”
[16] Gonzalo Casas et al., “Compas-Dev/Compas_fab: COMPAS FAB v0.27.0” (Zenodo, October 10, 2022), https://doi.org/10.5281/ZENODO.3469478.

automated pipeline provides opportunities for mass customization or modular assembly with non-standardized materials. It can facilitate the fabrication process by enclosing the loop so that a robot installed with sensors can look at unseen material geometries, make design decisions according to the rule systems built by designers, and execute the tool path while paying attention to the dynamic changes in the surrounding environment. Although the framework still requires optimization in certain performances, currently it can communicate data and execute without latency with a high degree of accuracy in data density.

1.1. Prerequisite

The physical hardware setup is comprised of a single robot manipulator (ABB robot family), an RGB-D camera with a holder, and a single end-effector to enable easy installation, calibration, and removal. The RGB-D camera is mainly used for visual SLAM algorithm for 3D environmental mapping and object pointcloud scanning. It is possible to replace the camera with a 3D laser scanner. The official ROS ABB driver package[17] by ROS industrial program is used to set up communication between the robot and ROS.

The software system is constructed based on bridging two significant platforms in robotics and architectural design: ROS 1(robotic operating system) and Grasshopper plugin for Rhino. Because ROS performs most stably in Linux Ubuntu systems, while Rhino and Grasshopper are only compatible with Mac and Windows systems, it is recommended to use two computers to run programs separately. The setup for this specific research is ROS noetic[18] with Ubuntu 20.04 and Anaconda/Grasshopper/Rhino 7 on Windows 11 with COMPAS FAB and roslibpy[19] for web-socket communication between two laptops.

[17] “Abb_driver - ROS Wiki,” accessed December 18, 2022, http://wiki.ros.org/abb_driver.
[18] “ROS Noetic Ninjemys.”
[19] “Roslibpy: ROS Bridge Library — Roslibpy 1.3.0 Documentation,” accessed December 15, 2022, https://roslibpy.readthedocs.io/en/latest/.

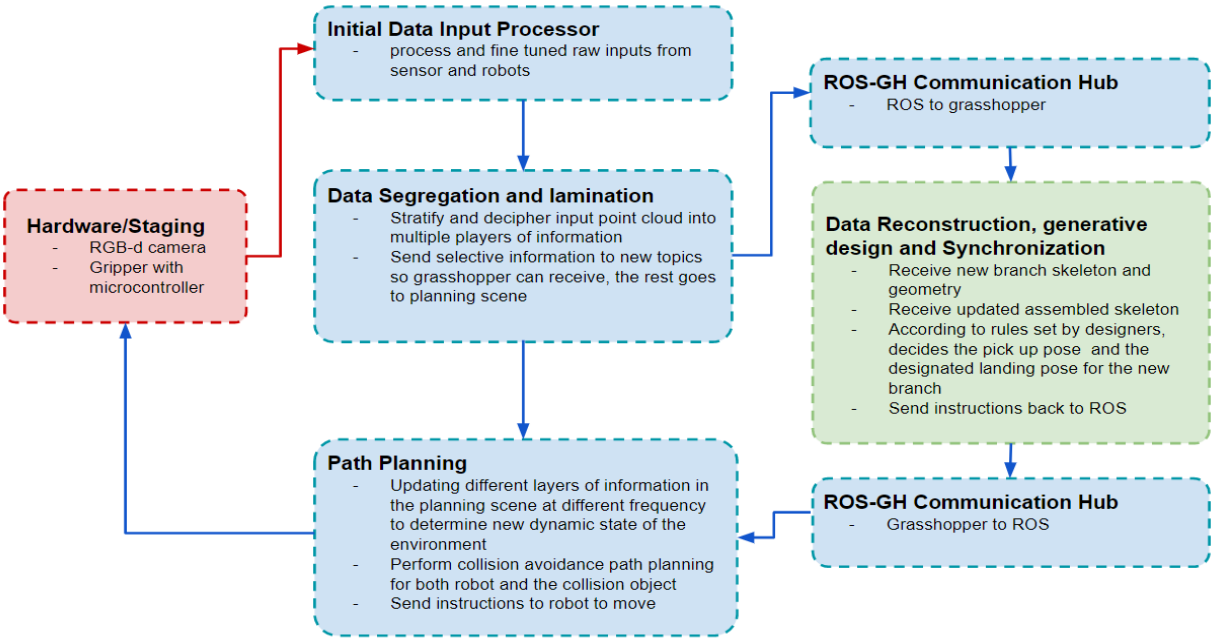


Fig. 4 Function description of the seven system modules in the framework

The system can be organized into the following modules, each containing designated functions and package/API:

1. Hardware/staging module**(Physical)**: the design and integration of camera and end-effector to maximally provide assistance to environment mapping, dynamic surveillance, and flexible operation, the replaceable design allows fast setup and removal without calibration. The robot requires two empty spaces with flat surfaces within its reach: a working cell to scan and work with new coming instances and an assembly cell for the assembled structure.
2. Initial data input processor**(ROS)**: The ABB ROS driver receives real-time joint status changes in a physical robot, the realsense2_camera[20] package receives/fine-tuned input pointcloud from the camera, and serial port reading from the Arduino that controls the gripper.
3. ROS-GH communication hub**(ROS)**: Grasshopper and ROS communicate through WebSockets, as an included element in COMPAS FAB built by the Gramazio Kohler research team from ETH. It is able to subscribe and publish real-time ROS messages from both ends, however, the message types currently

[20] "Realsense2_camera - ROS Wiki," accessed December 18, 2022, http://wiki.ros.org/realsense2_camera.

limit to ROS basic message types, and self-built message types are not yet recognizable.

4. Data segregation and lamination**(ROS)**: In order to extract effective information from the scanned pointcloud, this module filters and segregate data and associate them with different labels and updating frequency, the primary Package that runs the data processing are RTABmap[21] , Ocotomap[22] , box-filtering, euclidean clustering[23] and pointcloud skeleton with laplacian contraction[24] .
5. Data reconstruction, generative design, and Synchronization**(Grasshopper)**: This is the primary interaction module for designers, where designers can monitor visualized materials, build design rules and modify the robot's planning if necessary. The ultimate goal for this module is that once a global design is formed (a given geometry to map on or a design principle is formulated) the entire loop from physical environment initial scanning → material scanning → processing → design → placement → physical environment status update will be done automatically.

[21] "Rtabmap_ros - ROS Wiki," accessed November 10, 2022, http://wiki.ros.org/rtabmap_ros.
[22] Hornung et al., "OctoMap."
[23] D. N. Sparks, "Algorithm AS 58: Euclidean Cluster Analysis," *Applied Statistics* 22, no. 1 (1973): 126, https://doi.org/10.2307/2346321.
[24] Junjie Cao et al., "Point Cloud Skeletons via Laplacian Based Contraction," in *2010 Shape Modeling International Conference* (2010 Shape Modeling International (SMI), Aix-en-Provence, France: IEEE, 2010), 187–97, https://doi.org/10.1109/SMI.2010.25.

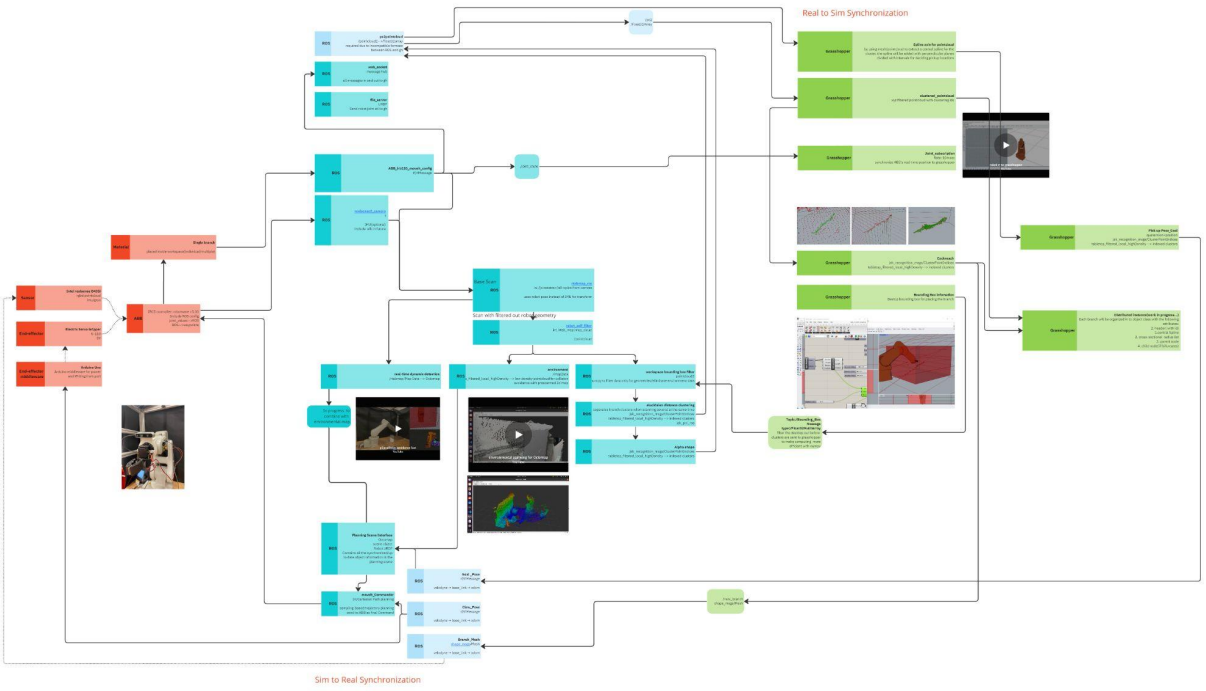


Fig. 5 a detailed atlas of algorithms/packages in the system, access viable through https://miro.com/app/board/uXjVPqtOGRk=/?share_link_id=944853260476

2. Hardware tool setup

In the field of robotics, two common sensor configurations can be categorized as ‘eye to hand(figure 6. a)’ and ‘eye in hand(figure. 6. b)’, respectively corresponding to stationary cameras that are calibrated and posed in the scene versus cameras anchored on a robot joint that moves with the robot.

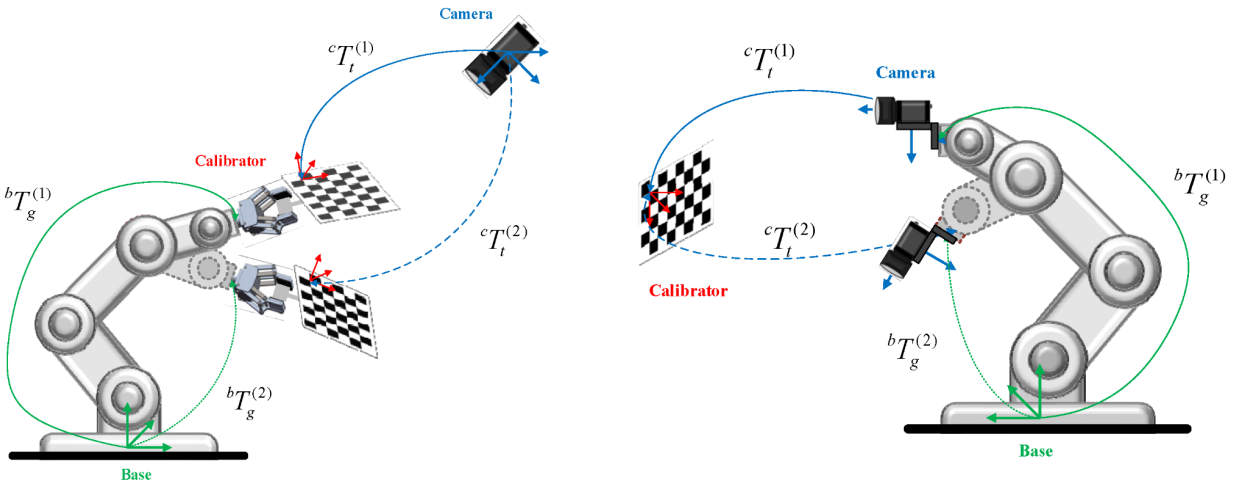


Fig. 6 (a) eye to hand (b) eye in hand[25]

Generally, stationary cameras provide stable and greater fields of view, while cameras on a robot enable mobility and panoramic scanning of the environment. Some related studies in architectural fabrication and construction used different setups to navigate the workflow, for example, '*Designing Natural Wood Log Structures with Stochastic Assembly and Deep learning*'[26] used a setup of a stationary kinetic camera to monitor the scene and provide geometry reconstruction for the log material(fig. 8). The team removed from the steps of picking in place by feeding logs directly to the robot's gripper, the static camera is able to retrieve log mesh according to the rotation of the robot and removal of robot volume. However, apart from a limited field of view, this setup requires a staged space, otherwise, it would require calibration every time during setup. It would be preferable for large robot manipulators to set up surveillance cameras to watch dynamics in dead corners from their carried sensors at a larger scale, yet the scale of irb120 requires fewer data augmentation to complete this task. Considering the adaptive nature of the robot to observe an evolving structure and unprecedented materials, eye-in-hand appears to be a better strategy in my scenario for limited resources and a dynamic operating environment.

The design of sensor installation and end-effector in this experiment meets the specific requirements of mobile scanning and visual SLAM map building. Visual SLAM requires steady movement to approximate the camera position from tf_tree, and the rotation of the 6-axis is undesirable due to unneeded spinning. The Sensor is attached to the end of the fifth axis of IRB120 to stay close to the gripper while remaining independent from the rotation of the 6th axis. The URDF robot configuration file in ROS provides the simulated status of the robot, sensor, and end-effector tool 0 while planning, to keep the movement of the camera integrated into the motion of the joints(fig. 7).

---

[25] Wenwei Lin et al., "Research of Online Hand–Eye Calibration Method Based on ChArUco Board," *Sensors* 22, no. 10 (January 2022): 3805, https://doi.org/10.3390/s22103805.

[26] Kaicong Wu and Axel Kilian, "Designing Natural Wood Log Structures with Stochastic Assembly and Deep Learning," in *Robotic Fabrication in Architecture, Art and Design 2018*, ed. Jan Willmann et al. (Cham: Springer International Publishing, 2019), 16–30, https://doi.org/10.1007/978-3-319-92294-2_2.

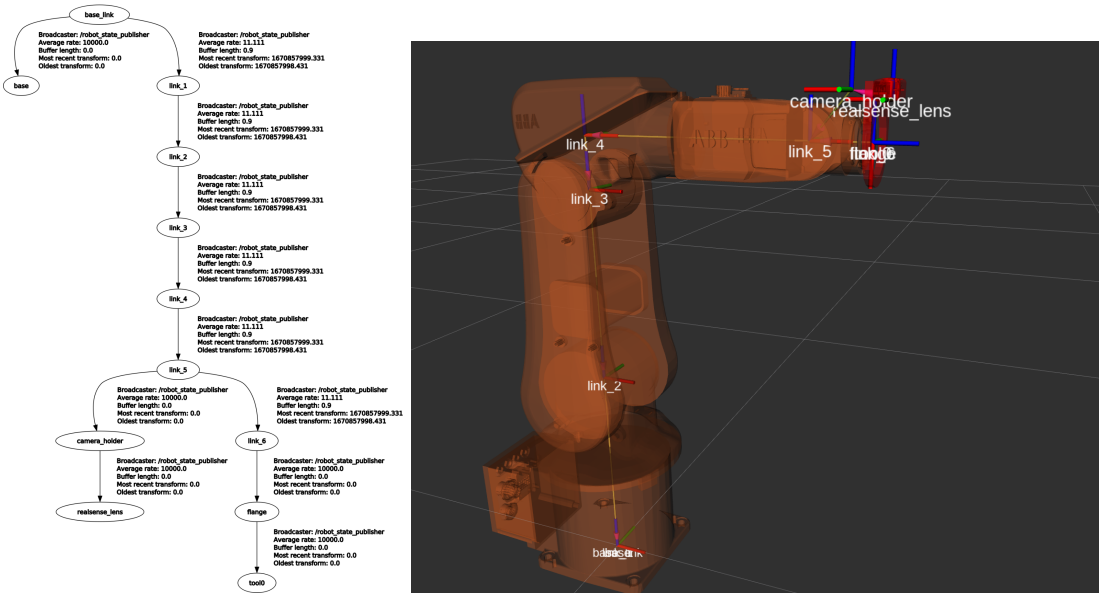


Fig. 7 URDF tf tree (diagram and RVIZ visual)

In order to create an end-effector to successfully pick up an irregular member and hold it firmly before it is detached, the gripper needs to have a coarse/sticky contact and wrapping surface, enough force to carry branches >= 500 g, and an accurate opening degree to adapt to branches with different radius. Two options were proposed in the early stage of end-effector design, one is a pneumatic gripper powered by the IRB120 I/O switch with gripper extension. However, the Pneumatic gripper is a single on/off opening system whose claws degree cannot be fine-tuned, although its control is integrated smoothly into the ABB RRC module, this option was aborted due to some technical challenges to modifying IRB120's connection board. Therefore I adopted an electric aluminum gripper, powered and controlled by an Arduino UNO microcontroller that connects to the ROS computer directly through a USB Cable(fig. 8).

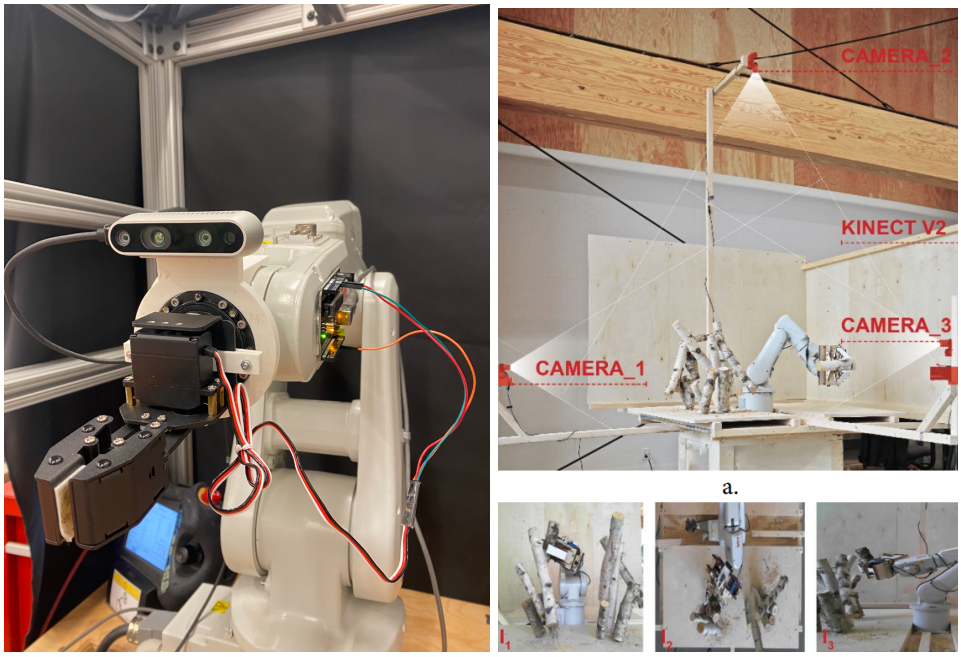


Fig. 8(left) installation of camera and gripper. (right) installation and setup of natural log assembly precedents[27]

3. Design-Operation communication module

In order for Grasshopper to receive real-time data to make considered design decisions, a selection of ROS messages is delivered to the grasshopper through WebSocket, while the meshes of the robot joints are delivered through the file_server package when WebSocket is launched. It is important to have all launch files up, especially the abb_experimental package that links the control with the physical robots and runs in ROS prior to enabling connections in Grasshopper. To connect the fundamentals with the robot configuration, COMPAS Fab provides a 'ROS connect' module which takes in the IP address of the Linux computer. Once connected, the system would begin to stream joint robot meshes, end-effector, and sensor geometries so that those movements can also be visualized inside the grasshopper. In addition to the robot itself, it is also necessary to take in some critical low-weight material information for the grasshopper to process design rules and generate corresponding results.

The following topics are subscribed:

1. Robot joint_states[sensor_msgs/JointState]: a list of 6 values

[27] Wu and Kilian.

2. Material cluster pointcloud[std_msgs/Float32MultiArray]: flattened into 1d array with dimensions and offsets
3. Branch skeleton information package[std_msgs/Float32MultiArray]: the synthesized message that connects the skeleton information of each branch is processed and published by a self-built ROS package which runs pointcloud laplacian contraction and assigns the branch with ids, which the detailed description would be presented in the material module. The Branch skeleton information
4. Assembly scene geometry skeleton[std_msgs/Float32MultiArray]:

The following topics are published:

1. Branch mesh: a Shape_msgs/mesh message that contains the points of a mesh and a list of triangular vertices indexes.
2. work cell/assembly cell filter box: 2 points with XYZ coordinates that specify the range and location of the cell to filter all residual environmental data besides materials. (fig. x)
3. Synthesized branch information: a sequential list of goal poses and their corresponding gripper status
    a. Quternion coordinates: 7 floats
    b. Attached object operation: 0(Attach)/1(Detach)/2(None)
    c. Gripper opening degree: 0-110
4. Operative confirm signal: toggle to proceed, this serves as an extra layer of safety precaution to acknowledge the operator that in digital synchronization the robot has attached/detached the correct branch geometry for collision avoidance path planning and has acknowledged the next accurate goal pose.

4. Environmental Dynamics and Collision Avoidance

One of the key components of this system is its adaptability to the changes in its working environment which accounts for the changes in collision avoidance path planning. In this system, RTABmap[28] , a real-time appearance-based mapping is used to construct the scanned scene with loop closure.

[28] Mathieu Labbé and François Michaud, "RTAB-Map as an Open-Source Lidar and Visual Simultaneous Localization and Mapping Library for Large-Scale and Long-Term Online Operation: LABBÉ AND MICHAUD," *Journal of Field Robotics* 36, no. 2 (March 2019): 416–46, https://doi.org/10.1002/rob.21831.

To build real-time 3d map with RGB-d camera and convert the pointcloud map to Octomap.

One needs to modify the configuration file and run the following package in sequence:

1. In abb_irb120_moveit_config: find ‘sensors_kinect_pointcloud.yaml’ and change the point_cloud topic to RTABmap topic with message type: [sensor_msgs/Pointcloud2]
2. Launch ABB robot control
3. Launch camera
4. Launch RTABmap and set RTABmap to use odometry instead of the camera’s internal imu, to change the resolution of the mapping quality, set rtabmap/Cellsize to customized density, the optimal resolution found is around 0.005m to 0.002m.

Base maps are scanned when the initial blank working scene is established by launching the camera and the roadmap package. Users then may call the save_map service to store the Octomap under a unique name for it to load when needed. During extensive assembling sessions, Octomap updates immediately after each new branch is placed and can be saved as another temporary working scene to be retrieved when assembly resumes. The typical scan time under jogging motion is around 20 minutes for irb120 in this setting, where the primary physical restrain comes from the scaffolding bar behind the robot. Therefore from a time-efficient perspective, pre-scanned Octomap is favorable for a persistent environment.

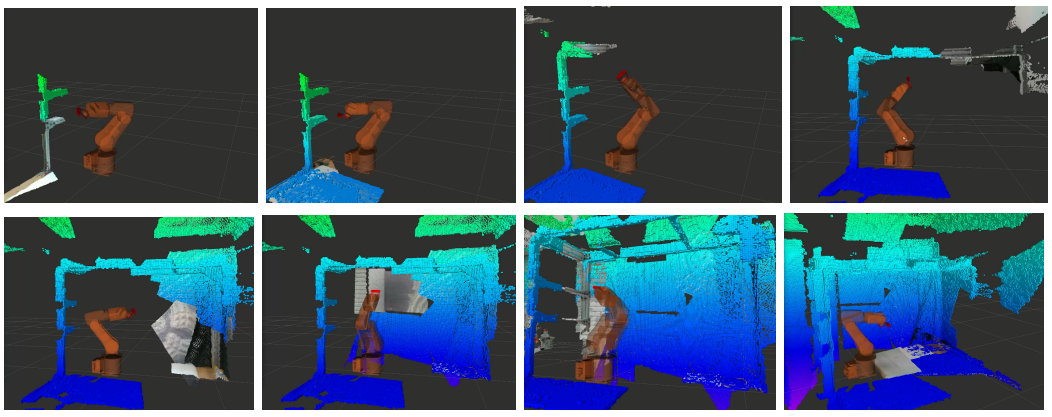


Fig. 9 Base map scanning process with RTABmap

Blank scene scanning with different resolutions was tested to understand its compatibility with the resolution of scanned branches. The two figures below demonstrate resolution at 0.0025 and 0.02m

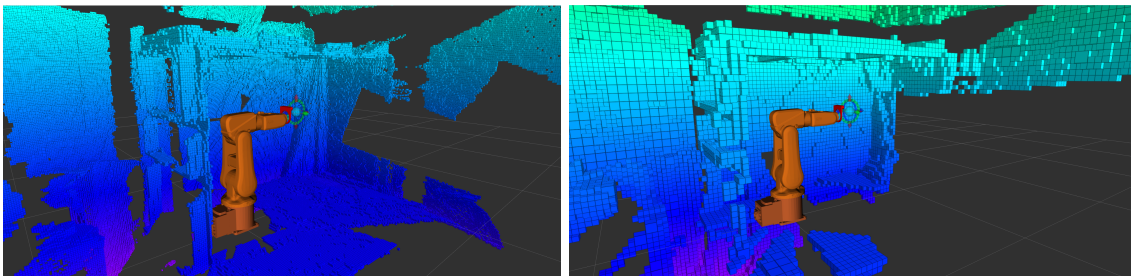


Fig. 10 a comparison of base maps with different resolutions(left: 0.0025)（right:0.02)

5. Data segmentation module

The fundamental goal of performing data segmentation is to help the system understand how to decipher the inputs into 3 layers: working environments, objects for immediate engagement, and indexed materials.

1. Working Environment: working environment refers to a general surrounding of an initial/blank working scene. It is the previous Octomap constantly updated by The pointcloud occupancy map updater in Moveit, accepting input from visual SLAM to monitor changes in the environment and engaged users. The working environment seeks to be completely avoided by the robot and its attached objects.

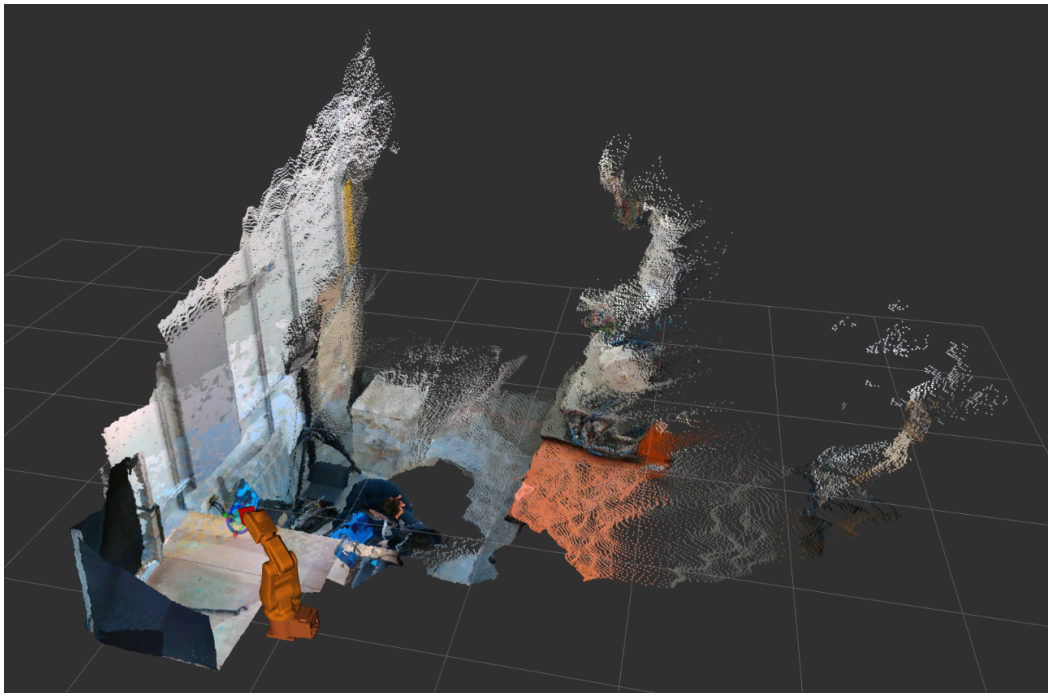


Fig. 11 From cloud_map we obtain the raw input from the RGB-d camera at the cell size of 0.002m, it contains all information

2. Objects for immediate engagement: these items in the experimental setting generally refer to the tabletop where new materials and assembly structures are placed, potential scaffolds, and the assembly where the new incoming objects will be placed and anchored as a part of the component. This layer of information requires high resolution because it needs to be intimately in touch with the planning object while not getting hit by it. In addition, during the updates in info exchanges, this layer may transit back and forth from pure pointcloud data to a mix of mesh, primitive geometries, and Octotree voxels inside the Moveit Planning Scene.

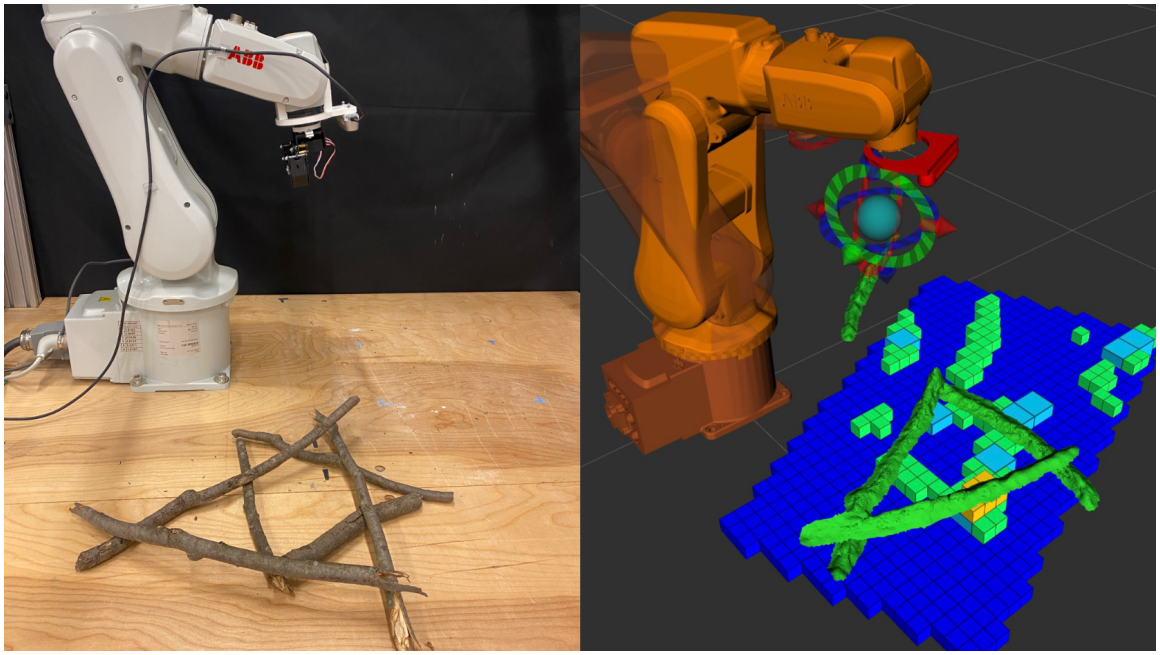


Fig. 12 object of immediate engagement which represents a mix of voxels and meshes.

3. Working objects: working objects are single instances of branches that the robot would choose to pick up and place. It is a mesh reconstructed by a grasshopper to represent a detailed digital replica. (fig. 14) There are 3 filters to extract the information for each branch. Inside batch tests, multiple branches are placed on top of the working cell surface, and the raw scanning input contains both the point cloud from the branches and the surface. By setting 2 points with XYZ coordinates, the bounding box will filter out the surface data according to height. To identify which point belongs to a specific branch, the Euclidean clustering algorithm is then performed to assign an index to each individual

point in the pointcloud, segregating the pointcloud into a number of point groups equivalent to the number of branches in the scene. Finally, to understand what kind of information is contained within each branch (number of splits, radius, orientations, etc), a curve skeleton that carries the central axis and interconnects relationship is extracted using laplacian-based contraction, reference from the paper *Point Cloud Skeletons via Laplacian-Based Contraction*[29]

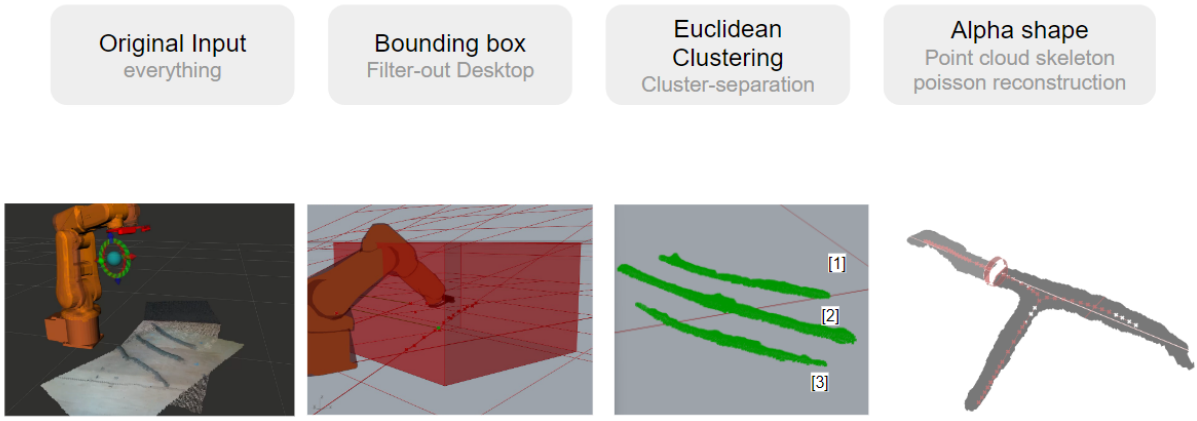


Fig. 13 branch analysis through 3 layers of filter

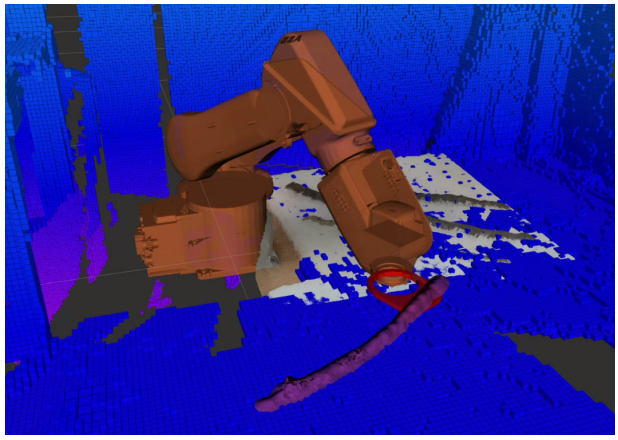


Fig. 14 working object in a purple mesh which represents the attached status to the robot's effector

6. Data lamination module

After sensor input data are organized into layers, the system would laminate them back for both planning and visual purposes. This represents an active selection process of combining and updating layers at different frequencies. The layering of such information

[29] Cao et al., "Point Cloud Skeletons via Laplacian Based Contraction."

is inspired by the design of a navigational stack in *Layered costmaps for context-sensitive navigation*.[30] Although mobile navigation is drastically different from the manipulator planning scene, I assume the same logic would be applicable to planning scenarios that seek to maximize their performance without losing the resolution in its target work object. Inside the system arrangement, Octomap will play the role equivalent to the bottom canvas layer in Photoshop, while additions of sensor inputs will aggregate on top of the base map. Note that the advantage of the base map is not only for time-saving purposes, it performs as an additional safety layer which the Octomap would preserve the voxels underneath the material, reminding the planning scene that a surface exists underneath the branch. In a single-layer scan where branches are scanned together with the placing surface, the rebuilt mesh will nullify its neighboring voxels, and the planning scene will assume that spaces are empty underneath the material. This is a dangerous dead corner for sampling-based path planning because the end-effector may try to push the material down to avoid neighboring voxels.

The working environment, the immediate engagement objects, and the working object together compose the final planning scene for OMPL to perform inverse kinematic path planning inside Moveit ROS.

7. Real-Sim Synchronization

The purpose of synchronizing motions and environmental data is less related to quantifiable data gathering but supports more for the visual of the designer. To carry forward communication from the physical world to its simulated environment in ROS and Grasshopper, the ROS client in Grasshopper is subscribed to the joint_states topic to know the current pose of the robot, and the data subclass of joint_state messages offers a list of 6 numbers that specifies the value for the data, they are streamed in order into the configuration component in grasshopper where the COMPAS FAB's robot component module updates the real-time status of robot visuals in rhino. The advantage of synchronizing ROS and grasshopper is enabling the designer to observe, measure, bake, and perform extra actions in Rhino, such as user input as a

[30] "Layered Costmaps for Context-Sensitive Navigation | IEEE Conference Publication | IEEE Xplore," accessed March 12, 2023, https://ieeexplore.ieee.org/document/6942636.

testing element to understand how to calibrate a good work cell space and then send the filter restriction back to ROS. Understand that the printouts in terminals and user interface in ROS are not suitable for any design operation, nor would they provide any immediate grasp of space understanding from a human perspective. Therefore it is important to associate designers with the robot and the material inside the design interface where configurations can be tested to work out a legitimate design strategy.

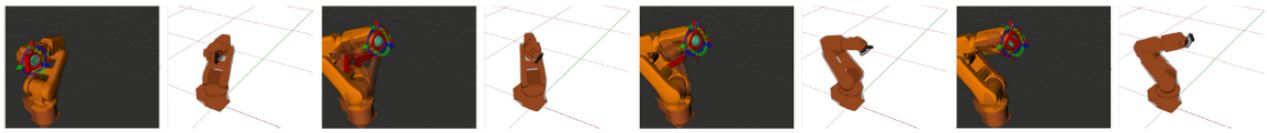


Fig.15 Synchronizing robot motion with ROS(left) and grasshopper(right)

8. Sim-Real Synchronization

Sim-Real Synchronization guides the operation of the robots so physical branches would be moved according to the geometries rebuilt inside the simulation environment. In order to update the robot and end effector with the right branch, in each iteration when multiple branches are placed inside the working cell, pick-up and placement goal poses are assigned to a certain branch through the grasshopper. Designers will be able to approve the visualized design prior to robot execution. The branch mesh reconstructed in Grasshopper will be published on the **branch_mesh** and subscribed by the **move_group** node to add the mesh to the planning theme. It will appear green inside the working cell(fig. 16) when the gripper has confirmed the attachment to the branch, mesh color will transit to purple to indicate its attachment to the end-effector(fig. 17) The branch mesh will resume green immediately after attachment and remove from the planning scene for local rectification.

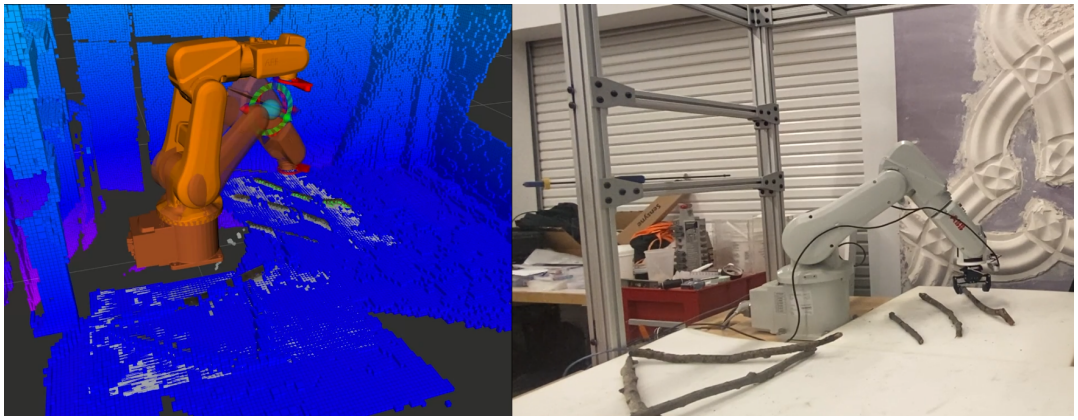


Fig. 16 one of the branches initiates the process by appearing as green mesh on the left

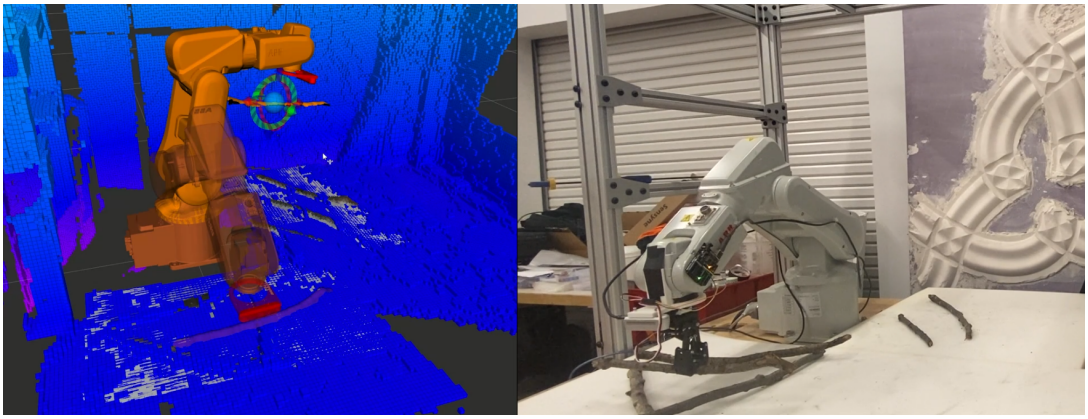


Fig. 17 the branch mesh appears purple while it is being moved by the gripper

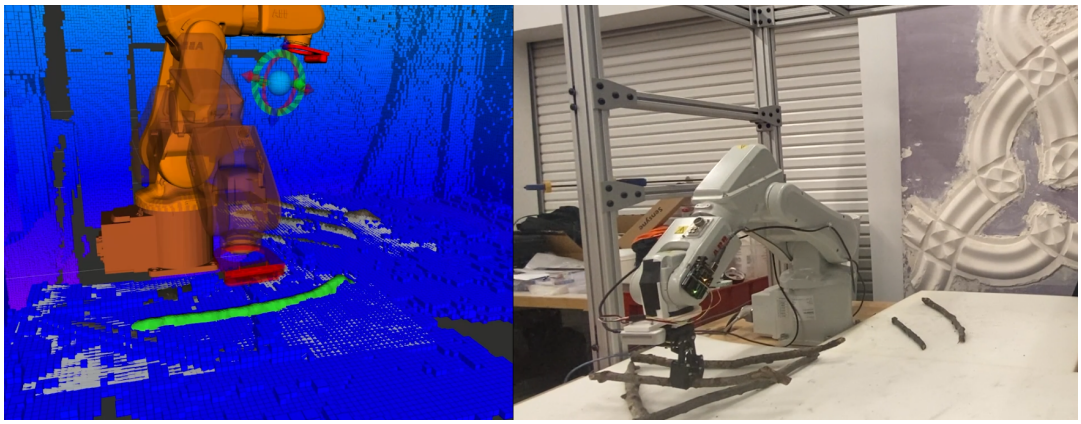


Fig. 18 The branch is detached and returns green when releasing decision is confirmed

9. Localized rectification

To acknowledge the discrepancy between the sensor-constructed planning scene and the physical result that the operation produced. Instead of using mesh to represent the branch that is being placed in the assembly cell, the robot would rescan the updated assembly structure per input of a new branch, registering pointcloud to Octomap. Recalculate the network skeleton of the assembly and update available edges of the branch for new incoming material instances to attach to.

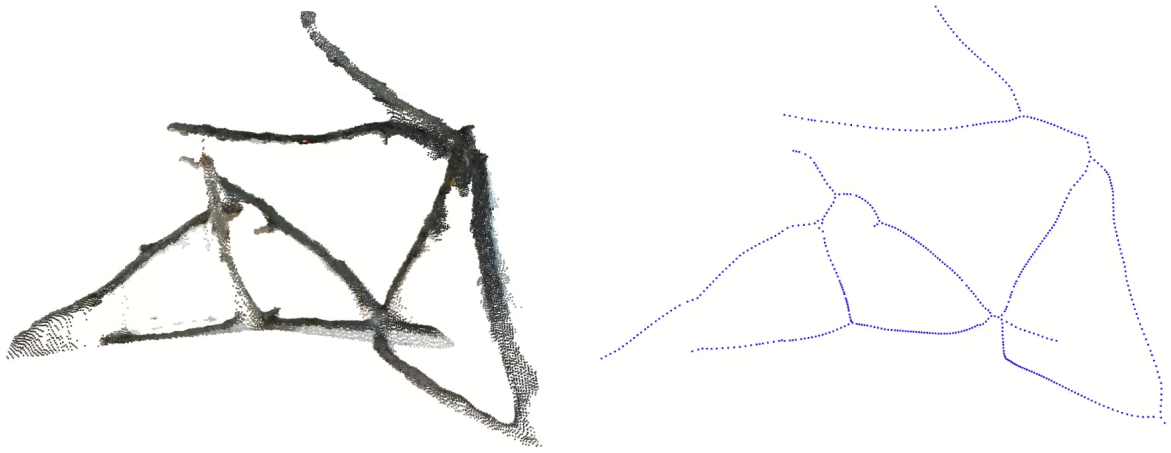


Fig. 19 Extracting network skeleton of a composition



Fig. 20 Evolvement of assembly structure through multiple cycles

10. Compliance Strategy

Physical compliance of spring for the gripper, local replanning, and shifting goal within a range. The performance of the gripper does not ensure that the branch is taken exactly in place and will result in a small scale of off-shift while pick-up, therefore a physical compliance strategy is designed to negotiate the difference when the gripper distorts the position of the branch. As the grippers close, it simultaneously creates a downward pushing force, which can be counteracted by a spring on the z-axis. Therefore I have designed a two-piece holder to connect

the 6th axis of irb120 and the gripper through a pivot center to allow small compliance in distortion(fig. 21)

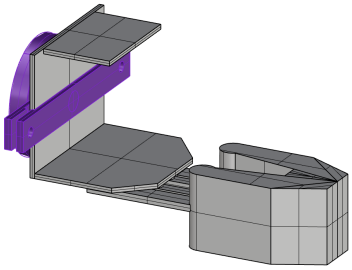


Fig. 21 Two-piece gripper holder

11. Distributed computing

It is observed that high-resolution 3D mapping and laplacian contraction constitute most of the time for planning and analysis. In order to compensate for the fast branch geometry rebuild, the scan cannot cover the part of the branch facing underneath that attaches to the tabletop. It cannot generate a closed mesh to perform skeleton extraction with the cockroach plugin in the grasshopper. The laplacian contraction of a skeleton with an incomplete pointcloud solved this problem and therefore part of the fundamental geometry decomposition analysis is allocated back to ROS, it sends lightly weighted information of nodes and connection lines through a flattened message to Grasshopper and waits to be unpacked. Sectional radii are defined by trimming the meshes in planes perpendicular to a line that connects two neighboring nodes, then identifying the largest distance between the mesh and the node.

To argue that with the increase of the resolution and volume of data, insufficient computing power and a slow network may impair the performance of the system. Therefore I designed the system to update periodically with certain inputs, allocate functions in different ways and selectively transmit data between computers to avoid latency. The COMPAS framework provides grasshopper computation and simulation for path planning, which enables in-scene collision objects and attached collision objects. However, the Ocotomap and OMPL path planning library in Moveit already provided a fast and accurate workflow for sensor input updates and path planning, therefore it is redundant and heavily loaded to compute collision avoidance path planning in Grasshopper. The only path planning relatable message generated by the grasshopper is the pick and place quaternion coordinates for the end-effector, to provide

sufficient space for the grasshopper to perform the iterative calculation for designing instance placement, mesh reconstruction and provide visuals for designers to look at the placing decision. Due to the fact that the system remains in the experimental condition, many gateways are placed prior to proceeding to the next operation, the double check includes 1), the sending of a single goal message. 2), the sending of the mesh geometry of the branch. 3), the opening and closing of the gripper 4), the attachment and detachment of the object in RVIZ. This is to prevent falsified design decisions from the grasshopper, and the manual mode of the controller provides additional safety precautions.

Results

Material Studies in Natural Wood

1. The Node System

   By using laplacian extraction, I am able to get points and lines that refer to point indexes as a network for each individual branch, by inspecting the node relationship as well as its sectional radius, a hierarchical grading of the branch and its sub-branch can be stored and be used towards the assembly design. The following are the major principal node for connections

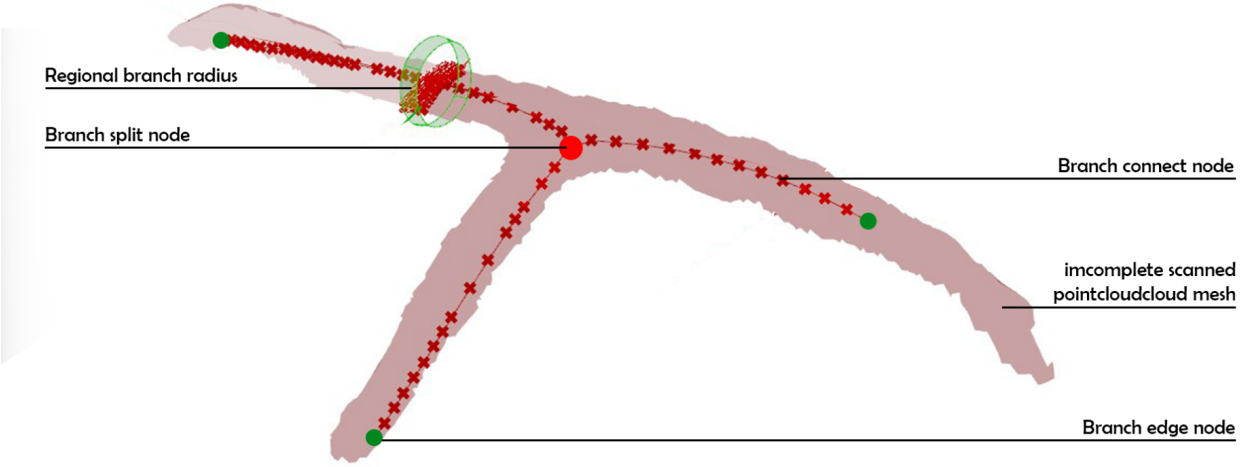


Fig. 22 a sample bifurcated branch with its skeleton and mesh reconstruction in Grasshopper

   a. The edge node: nodes that reach the end of the branch, connecting to a single line.
   b. The connecting node: nodes that connect two skeleton lines.
   c. The split node: nodes that connects more than two skeleton lines.

2. The Database





Fig. 23 around 50 samples of branches were collected for initial design analysis, ranging from 40cm to 90 cm

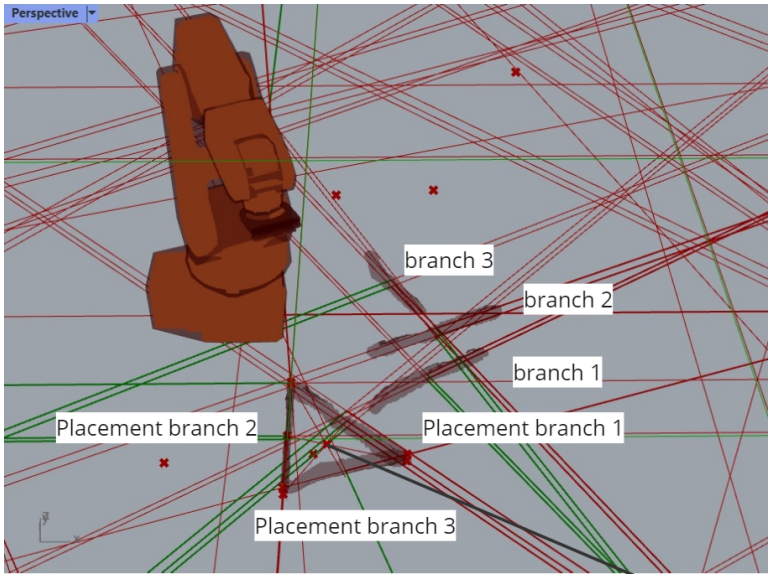


Fig. 24 The display of the robot, 3 physical branches with their relative location on the right, and placement simulation on the left

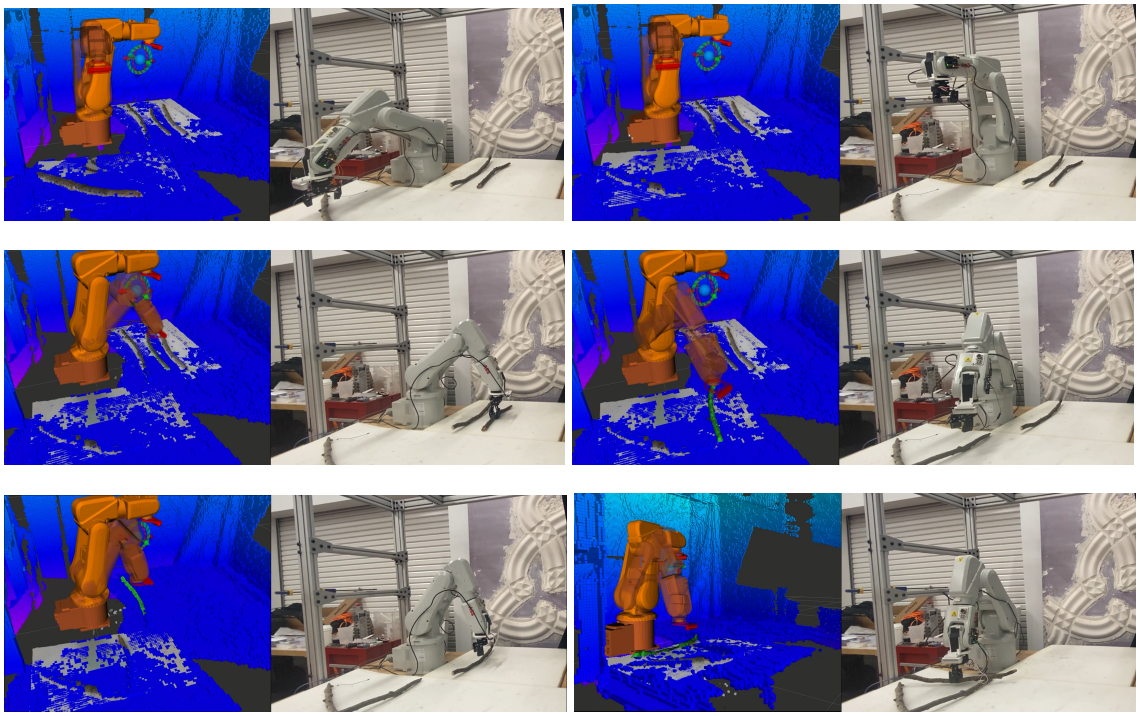


Fig. 25 composing a planar triangle based on a given centroid with a batch of 3 branches

Design Algorithms and Optimizations

1. Modular testings: stacking

   This test explores the possible challenges of overlaying a second module onto the first assembly. By creating a specific inference according to the previous task. The modular

orients around the same centroid but with a different rotational degree, forming 2 supporting points for each of the subsequent branches.

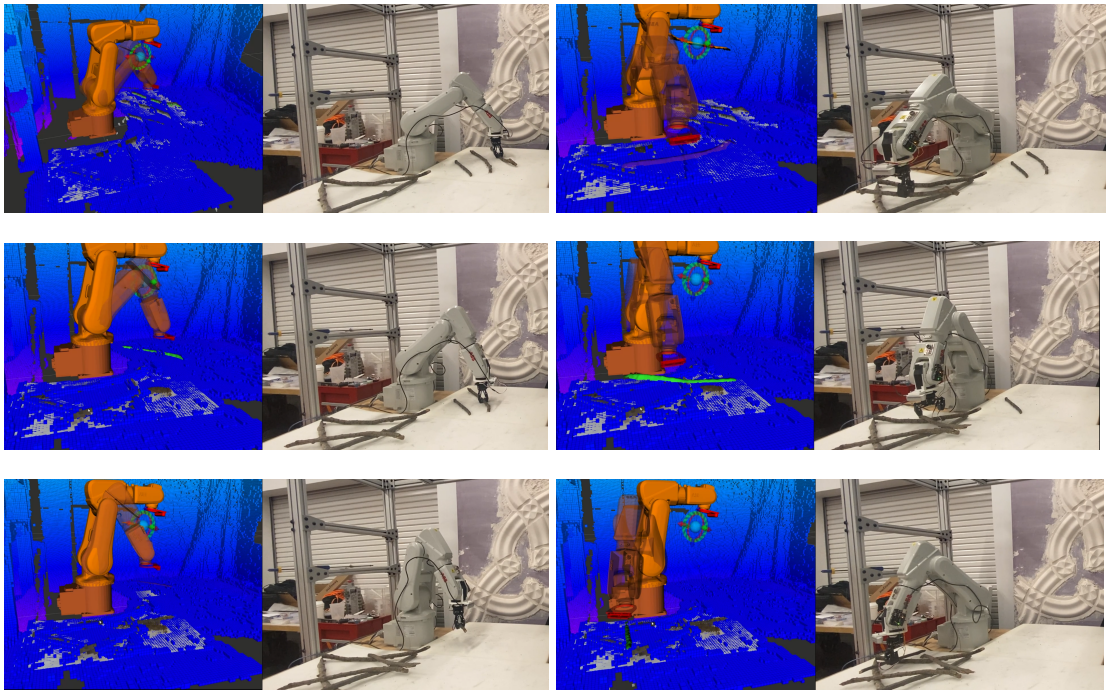


Fig. 26 stacking additional module onto the previous module

2. Modular testing: 3D positioning

   In order to understand how the modular should be arranged according to an existing non-planar condition in space, which in the next step can potentially contribute to a space frame or other double curved surfaces. A test to place a 3D modular assembly that leans against a tilted wall is performed.

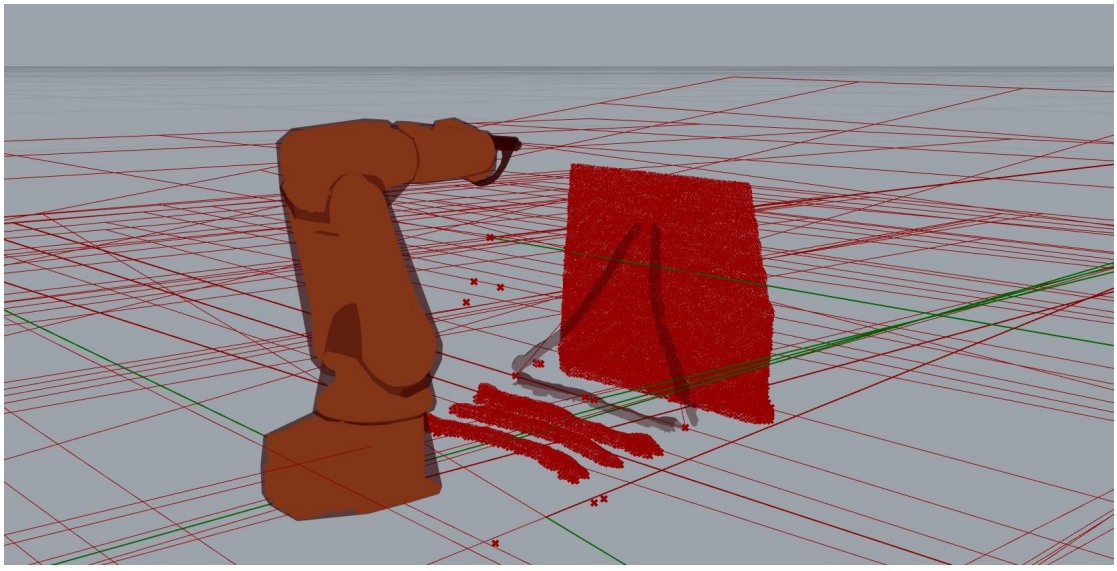


Fig. 27 grasshopper design decision with visuals

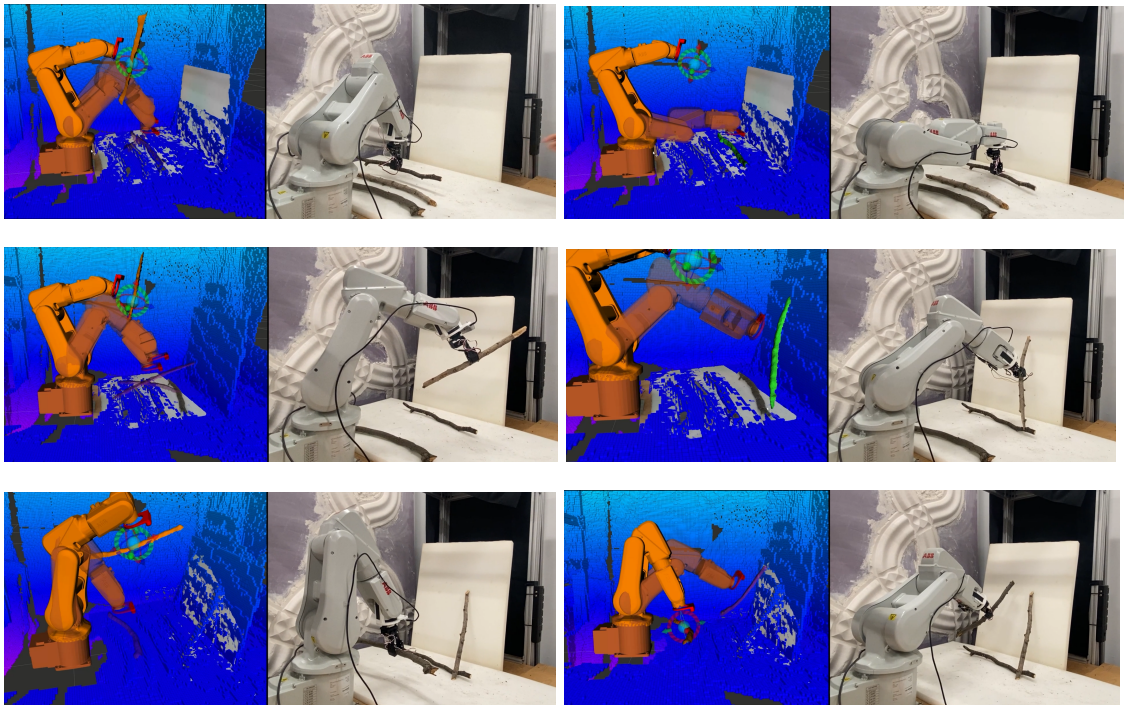


Fig. 28 The test demonstrating how branches' positions are updated in real-time Octomap while they are assembled

3. **Joinery**

Hard joinery is the first choice to work for the branch assembly process, by using a nail stapler with nails around 1-1/4in length to anchor one branch to another. It is strong and stable for anchoring the first element of the assembly. However, it is observed that the force of backlash is often two strong while two branches are only holding together in the air, without back support to cancel the force. In addition, due to the safety toggle to prevent misfires, the angles allowed for stapling are limited and sometimes result in loose fixings.

In two previously discussed research, the stochastic log assembly and unified sequenced path planning, both use glue and drills to keep pieces in space. Therefore elastic joinery tying is introduced to help with the situation, by using elastic ties/temporary tapes to connect branches in the first place and fill the connection with fluid hot glue giving some

degrees of tolerance and flexibility to the assembled structure. While on the other hand, the offset of the location of the branches can be locally rectified by performing structure rescanning and network skeleton extraction after every single iteration. This is equivalent to the adjusting skills of craftsmanship where measurements and decisions are moving dynamically to morph deviated physical conditions back into a general blueprint. The accumulative rectification would ensure that the final production adheres closely to the intended general geometry. It is observed that resin glue is strong and stable for self-standing structures assembled for about 2-3 meters in height, but this connection can fall apart due to the following reasons:

1. The peel off of the branch bark
2. Insufficient contact surface for two branches

To develop towards the construction of a full-scale, self-standing resilient timber structure with a larger cross-sectional radius, a stronger joinery needs to be developed to combine branch elements in replacement of the resin glue.



Fig. 29 Resin glue connection between branches

4. **Rule-based Design System**

A modular test with 3 branches proved to be successful, the next challenge is to explore the possibilities of constructing a spine-based or a modular-based design strategy to form multiple modules into a desired surface. The wood chip barn provides another inspiration for constructing a primary axis for a form and aggregate on top of the spine.

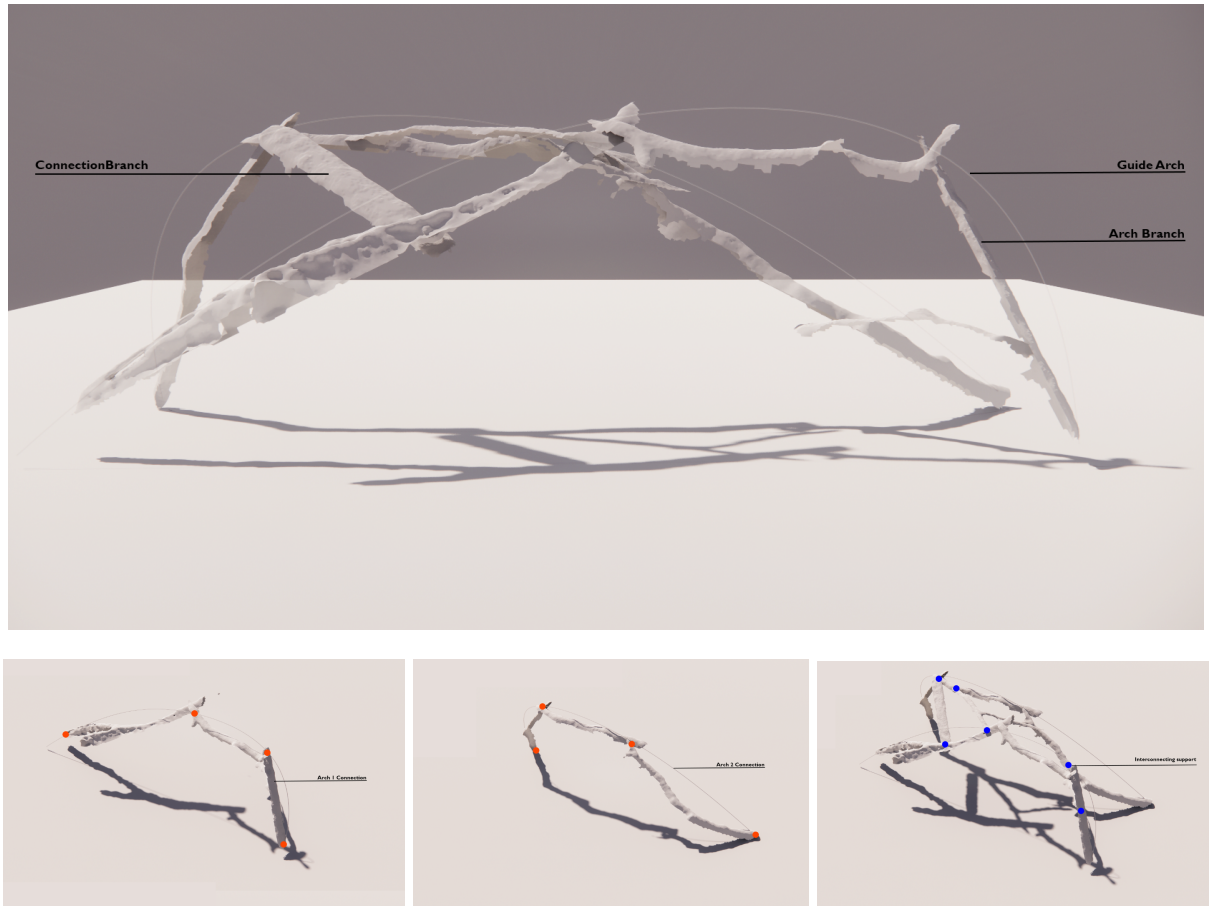


Fig. 30 This sample design shows how two formulated spines are connected by a smaller section of branches.

Apart from the spine-guided design, subsequent design algorithms are developed to construct irregular branches to a global geometry. Given a designer's surface input, a UV-guided design strategy was developed specifically for the node-edge skeleton. This technique maps irregular elements to a global geometry with linear consecutive inputs that require recurring measurements of the growing distribution of elements. The process seeks solutions for the following challenges:

1. How to position new inputs to both connect to the existing structure while fitting to the global surface.
2. How to ensure the alignment of edge conditions and Delaunay triangulation to maximize structural integrity.

The following pseudo-code delineates the constructing and iterative mapping process into 3 general conditions:

1. Bottom Spline implementation: The current strategy assumes that the start edge curve is continuous and the branch would align by its edge to fit the base curvature.

Algorithm 1: Bottom Spline fitting

Requires: the curve is divided into even space points and the points interpolate sequentially ;
Result: fitting branches onto a curve
Global last approximation;
$x \leftarrow 0$;
$y \leftarrow$ curve divided point list;
$r \leftarrow$branch's length;
while $x < len(y)$ **do**
    y = y[x:] delete all points before the current start point;
    draw a sphere with y[x] as center and r as radius;
    get the index z of the point closet to the sphere surface in grasshopper;
    if $z > x$ **then**
        $EdgeA \leftarrow x$;
        $EdgeB \leftarrow z$;
    end
    $x \leftarrow z$;
end

Fig. 31 Spline fitting strategy

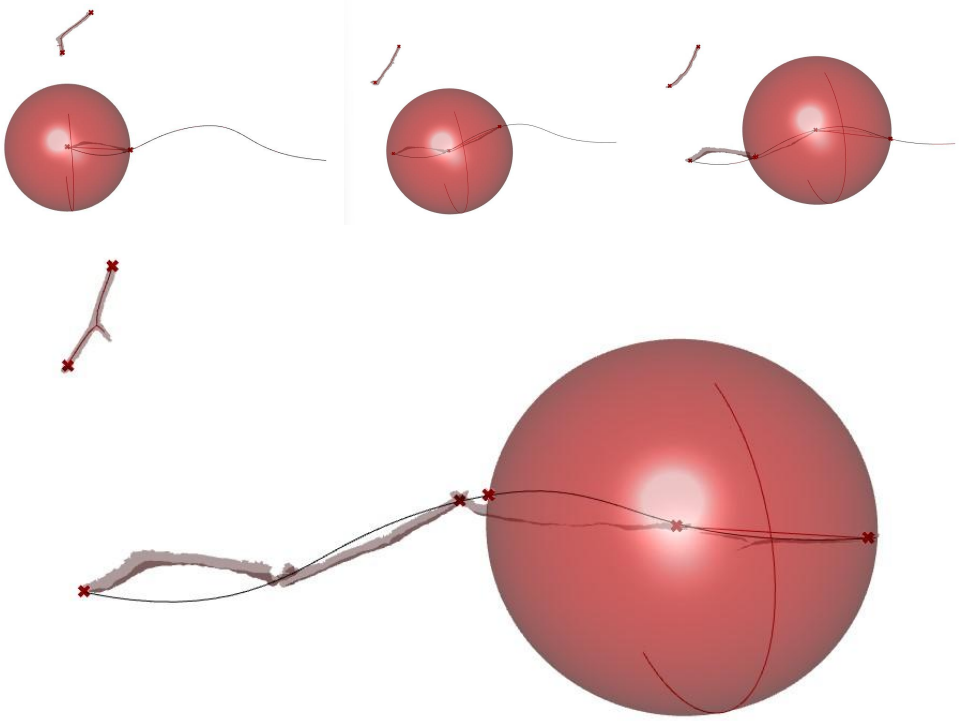


Fig. 32 Spline fitting sample process (top left: scanned input branch at its original coordinate)

2. UV layer binary search: This search method recursively searches for positions to fit branches back and forth on the same U isocurve level.
    a. Forward condition: Edge A connects with a node in the assembly skeleton in the existing cell{x,y}, Edge B search for a point on the surface in grid {x+2,y+1}
    b. Backward condition: Edge A connects with a node in the assembly skeleton in the existing cell{x,y}, Edge B search for a point on the surface in grid {x+2,y-1}
    c. Edge fitting: A Edge wrapping condition to orient the branch input to the edge of the geometry
        i. At a single level where the search has just started
        ii. At a single level where the search is ending

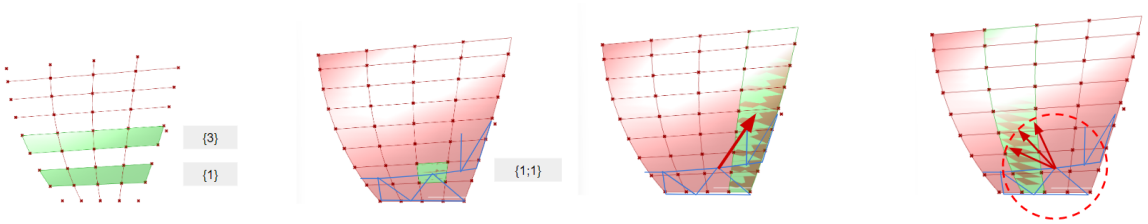


Fig. 33 visualization of forward search and backward search

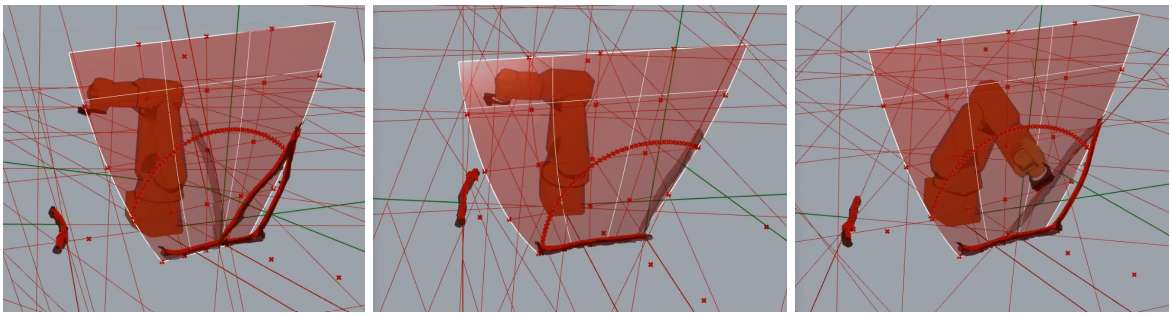


Fig. 34 reatltime visualization in Grasshopper during Assembly

Algorithm 2: UV layer binary search

```
Result: fitting branch onto UV surface diagonally
Global last node grid;
Global Current node grid;
y ← 0;
x ← 0;
while x ≥ 0 and x is odd do
    while y in current layer x do
        if y is 0 or -1 then
            EdgeA ← available assembly skeleton nodes in [x,y];
            EdgeB ← UV point at [x,y+1];
        else
            if y is even then
                perform forward search;
                EdgeA ← available assembly skeleton nodes in [x,y];
                EdgeB ← available assembly skeleton nodes in [x+2,y+1];
                last node grid ← [x + 2, y + 1];
                publish last node grid;
            else
                if y is odd then
                    perform backward search;
                    EdgeA ←available assembly skeleton nodes in [x,y+1];
                    EdgeB ←last node grid;
                end
            end
        end
        y ← y + 1;
        publish current node grid;
    end
    x ← x + 2;
    y ← 0;
    publish current node grid;
end
```

Fig. 35 UV guided input pseudo-code

3. Triangulation: After fitting each layer x, another layer search is performed to construct triangulation between members for the stability of the structure, as well as playing the role of the base node for the next layer {x+1; :}, the searching sequence is similar to UV's binary search except the fact that Edge B will search forward consecutively for grid{x, y:}only and memorize Edge B's grid and serve as Edge A for the next cycle.

Algorithm 3: triangulation connection

Result: fitting branch onto UV surface diagonally

$y \leftarrow 0$;
$x \leftarrow 1$;
while $x \geq 0$ *and* x *is odd* **do**
  while y *in current layer* x **do**
    $EdgeA \leftarrow$ available assembly skeleton nodes in [x,y];
    if *available skeleton nodes in* $[x, y+1]$ **then**
      $EdgeB \leftarrow$ available assembly skeleton nodes in [x,y+1];
    else
      $y \leftarrow y + 1$
    end
  end
  $x \leftarrow x + 2$;
end

Fig. 36 Triangulation Pseudo Code

The current UV-guided principle for a global geometry is subject to geometries that do not contain drastic guiding curves, or interruptions and was not tested with extensive variation of branch sizes and length due to the fact that branch samples collected vary only from 40cm to 90cm in length.

In Grasshopper, the skeleton and pointcloud sent from ROS enable designers to observe how well the structure adheres to the surface. In addition, as a temporary solution, if a single working assembly is partially outside the working range of the manipulator, re-positioning of the physical structure can be easily calibrated by manually fitting the surface to the skeleton in rhino to continue the fabrication process. For future development, it is expected that an auto-calibration should be done every iteration to automatically re-locate the surface according to the skeleton.

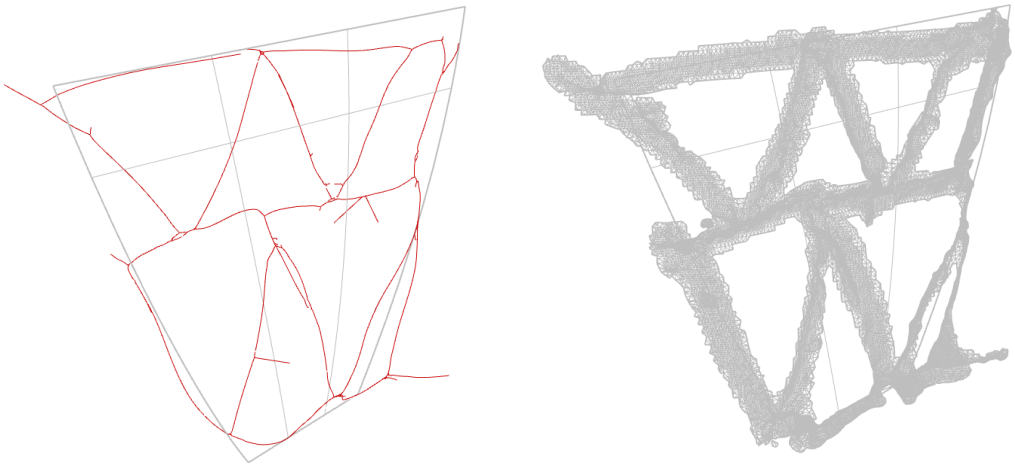


Fig. 37 Visualization of skeleton/Mesh and global surface alignment in Rhino

5. Work Range Expansion

Through constructing prototypes with multiple strategies and different scales, an optimizer was developed for accelerating the design-trajectory pipeline and enhancing the planning success rate. This optimizer plays an important role in expanding the working limit for assembly that reaches the border of the operation range. There are 4 parameters for the optimizer: 1) pick up position 2) placing orientation and position 3) position tolerance 4) orientation tolerance

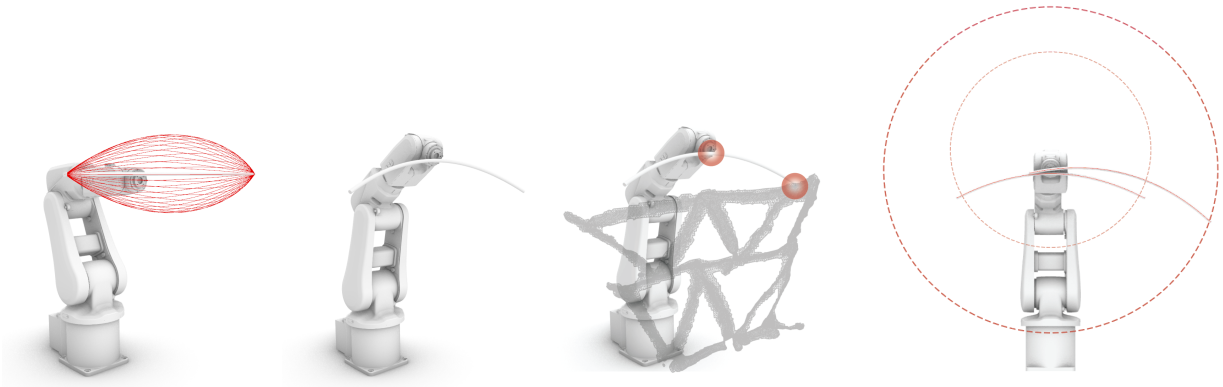


Fig. 38 1) placing orientation adjustments 2) anchored edge node 3) tolerance adjustment

1) Working boundary expansion defined by gripping pose

The optimizer is organized into two layers of filters:

1. Pick and place interactive filter with non-collision check IK solution
2. Tolerance adjustment filter with Octomap and planning scene object included IK planning trajectory

In grasshopper, the optimizer will randomly select available pick-up nodes, sorted by x direction, in between range 25%~75% of all available nodes. The intention is to ensure the stability of the branch during movement, due to the fact that picking from the far edge normally results in higher torque by gravity and contains a higher chance to fall off from the gripper during spinning movements. However, in extreme cases in which the landing position are exceeding the limits of the manipulator's working range, edge pick-up points are preferable for the operation and will therefore be taken into account. Grasshopper maps and orients the branch to the assembly structure, as well as the pickup plane. According to the design principles, only two edge nodes from the branch are anchored and therefore allow the branch to 3D rotate along the axis based on its linear nature. Therefore the placing solution would rotate the branch and placing plane to find the first solution available with an interval of 10 degrees. If there are no placing solutions available, it serves as an indication that the branch is at a challenging location that would therefore revert its pick-up plane for an edge pick-up option. Currently, there are multiple causes for the failure and the optimizer that negotiates pick and place planes would only solve this problem partially.

Both the pick and place planes will be verified by an initial pass of inverse kinematic solutions without any collision check, taking baseline feedback on whether the manipulator can pose at TCP's target plane. Grasshopper will only send pick and place instructions if both picking and placing conditions are satisfied. Both target planes will be sent in the format of quaternion coordinates along with specified gripper opening angles and attach/detach information as two messages to ROS's planning interface.

At the second filter in ROS, the planning scene will plan by adjusting positions and orientation tolerance for target quaternion coordinates. The pickup pose requires both position and orientation tolerance to be smaller than 1e-4 based on extensive tests where pick-up poses are

close to no failure in planning. However, the placing position contains various factors, such as structure radius, scanning offset and calibration, Octomap resolution, and physical shifting due to human interference. Therefore for each cycle, the planner has been given at most 10 replanning trials in which the tolerance will steadily increase to a maximum of 3cm in position tolerance and 0.1 radians for orientation tolerance before successful planning. This strategy is particularly useful with the growing complexity of the assembly structure and the constrained working space for irb120. I believe the send-layer filter can be improved by a more comprehensive scanning strategy and planning solution in the future to bridge the difficulty of surface contact between materials.

Assembly Prototype

The prototype experiments start from a small scale to a larger expansion in shape and size to explore both physical limitations and global design limitations. Six pieces of prototypes are constructed in which three of which are mass-customized modular components for a larger structure. Each of these prototypes follows an iterative process of single branch input until the assembly skeleton satisfies its comparison to the global geometry: 1)rescan assembly, 2), move towards input work cell and scan new input branch, 3) extract assembly and input skeleton, update Grasshopper planning scene for a design decision, 4) update ROS planning scene and plan trajectory according to the design decision, 5) human operation to manually glue the branch to the assembly and wait for the connection to cure before the robot releases the holding.

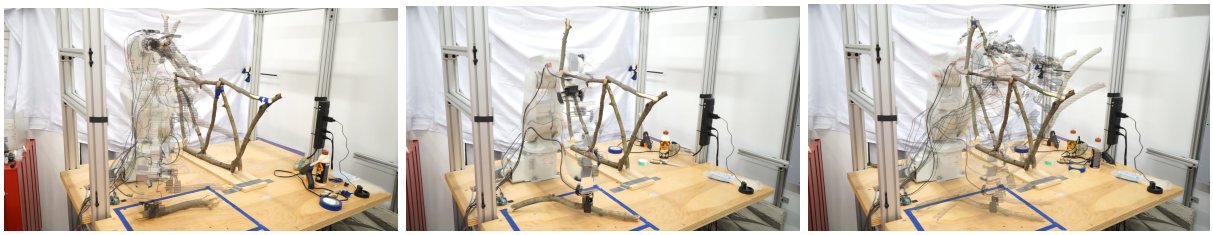


Fig. 39 Selective trajectories during assembly

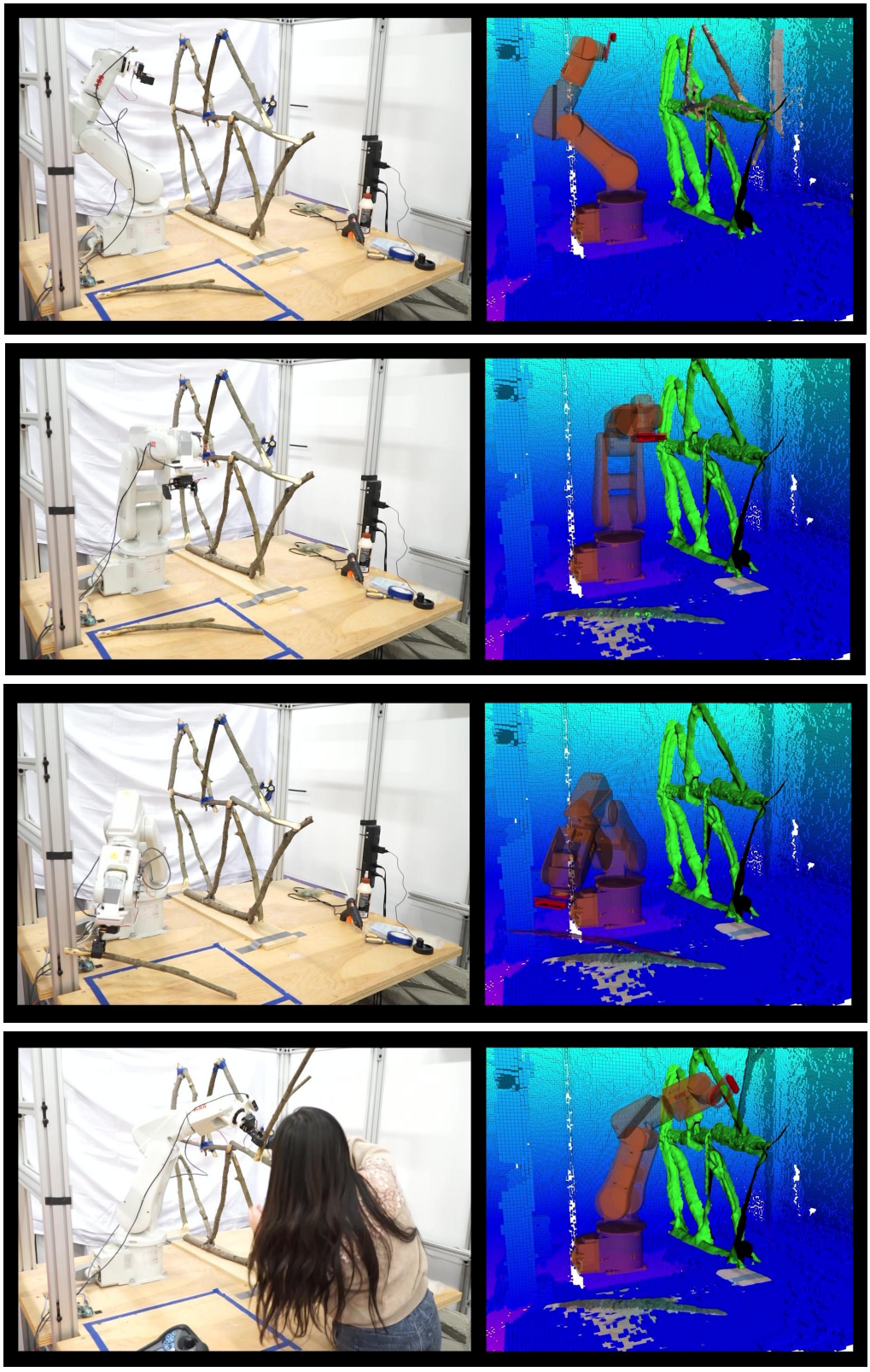


Fig. 40 Physical photos and ROS screenshot of an iterative assembly process

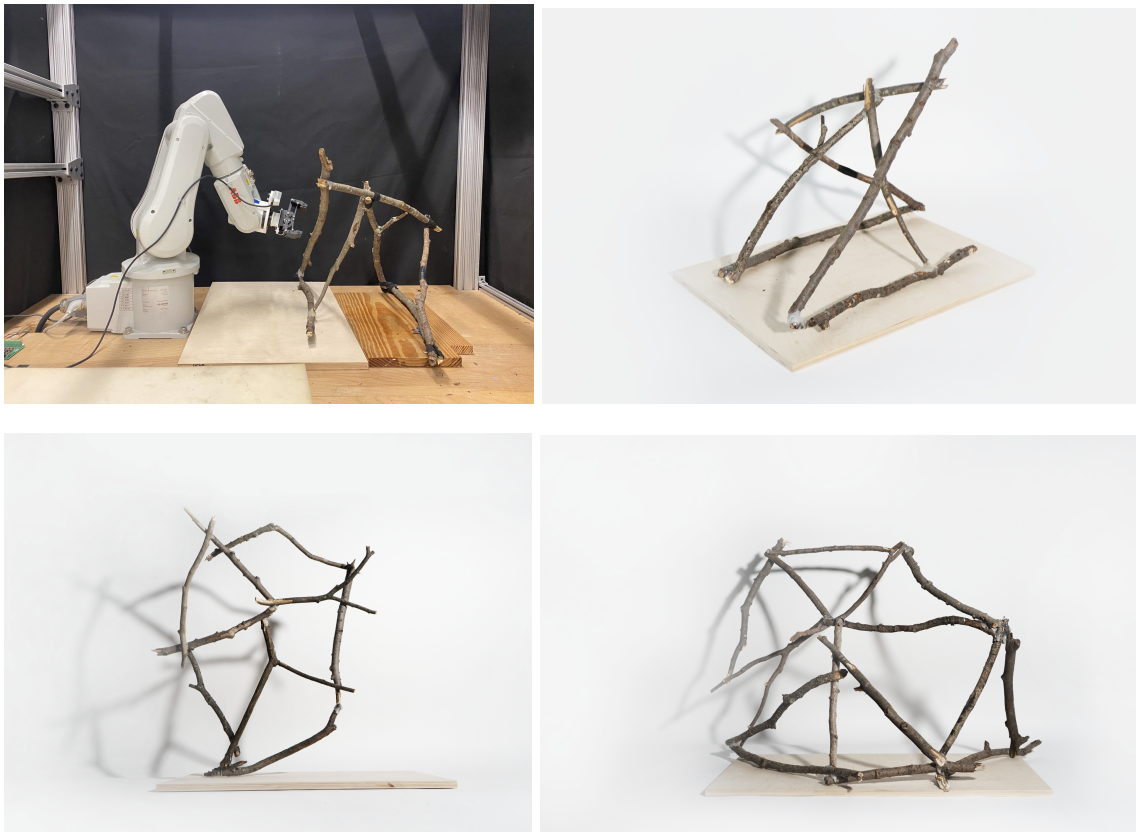


Fig. 41 Initial prototypes (A, A, B, C)



Fig. 42 Larger prototype assembled by 2/3 customized modular components

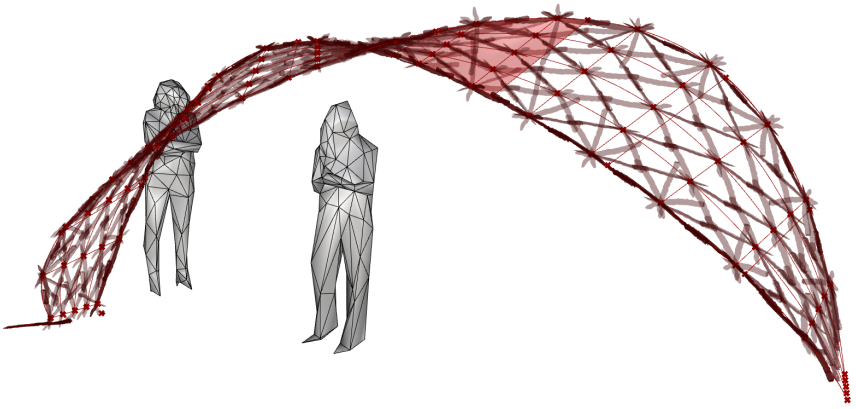


Fig. 43 Large scale structure and its modular components executed by the framework

Conclusion

Through extensive research and physical testings, a responsive robotic assembly framework, working specifically with natural wood materials, is designed and configurated. The advancement of real-time path planning and geometrical computations has enabled the potential for a robot to autonomously works in a dynamic environment and with non-standardized materials that are hard to label or calibrate. Natural, free-form material types offer inspiration for space design with their unique identity and intricate nature, which I believe will be valuable for architectural manufacturing and construction in the future. In addition, they also suggest development in mass-customization with their varying sizes and shapes. These materials desire online programming, real-time localization, and collision avoidance planning, which are essential features offered by this framework. The following sections of evaluation, contribution, and future steps, deliver an overview of the system's advantages, constraints, and potential to elaborate.

1. Evaluation of the current system
    a. The existing system aims to provide essential modules to enable an intelligent assembly system for natural wood branches while keeping the framework open to additional integration. All ROS packages and grasshopper cells are dedicated to information-specific channels so this digital information can be rewired and adapted to additional functions. The following narrates observations, constraints, and understanding of the performance based on physical experiments.
        i. Material localization, extraction, and analysis: the three-layer filters and extraction in ROS have proven to be successful and mature in understanding where the branches are, the number of branches and their individual geometries, and the simplified skeleton to represent each branch to support operation and design. The scale of the branch remains within

the reach of IRB120(<1.0m), larger log scan and reconstruction will strengthen this workflow in the future

ii. Pick and Place: pick and place refers to how irb120 calculates the optimal quaternion coordinates to pick up the branch effectively in place, control the gripper and re-orient the branch according to the grasshopper's design instruction. This is the most physically challenging part which heavily relies on the gripping force and compliance design to keep branches from deviation. Through several iterations of hardware updates, the branch can be picked up stably so its relative position keeps coherent with ROS's planning scene. However, heavy branches sometimes fall off due to weight.

iii. Sensor-based mapping and collision avoidance path planning: The OMPL provides a strong foundation for sampling-based path planning based on given information. While the Visual SLAM package rtab_map is not dedicated to manipulators but more to navigation, it is creating high-fidelity results for the robot to understand its surrounding environment. One of the potential upgrades is to control the refreshing rate of rtab_map. It is observed that the complexity to cover the scan of the entire assembly structure increases drastically with the expansion in scale and geometrical twists. Currently, the fastest way to perform a holistic scan is with manual jogging instead of sending a pose to the ROS interface. A specific algorithm to calculate the minimum number of poses requires to complete the scan is preferable for the next step.

iv. Human-supported operation: By having the IRB120 manipulator perform the pickup and placing of the branch based on its design judgment, human operators complete the task of linking the new branch and the rest of the assembly with resin glue, nailing, and elastic band tying. Currently, it is still in a primitive stage and requires additional testing.

2. Contributions

The thesis contributes to robotic architectural assembly in the following ways:

1. Orchestrate autonomous fabrication process for non-standardized materials with both linear input and batch input, where input is localized and indexed.
2. An optimized solution for processing non-standardized material with SLAM localization, Poisson surface rebuild, and laplacian skeleton extraction with a node-edge system to guide design inputs and strategies. With these strategies, clear and concise geometrical information of both the input branch and changing assembly is able to stream from a complex physical environment to ROS and grasshopper to assist design and planning.
3. A bottom-up design strategy that guides non-standardized material input to be mapped onto a global geometry using edge and node system

4. A two-part system to optimize data streaming and computing efficiency for environmentally sensitive path-planning in ROS and parametric design in Grasshopper with selective information exchanges. Each of the planning scenes only contains essential data for either computation or visualization for human operators.

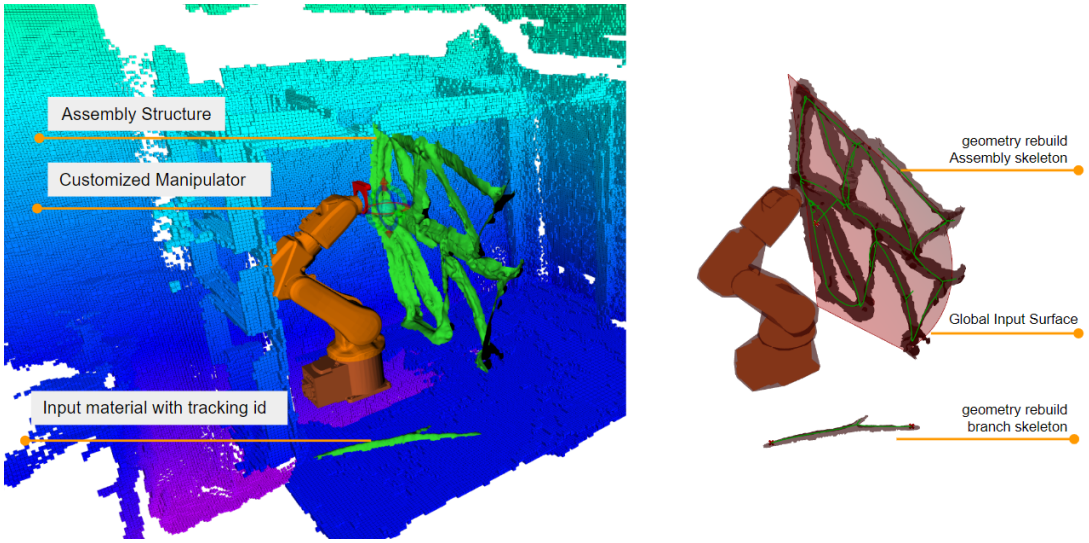


Fig. 44 Two-part system in which ROS focuses on planning semantics and Grasshopper focuses on design semantics

3. Future work

Although the scale of creations is limited due to the operation range and payload of the experiment robot irb120, which allows a maximum span of around 1.2 m per module. If equipped with a robot with higher payload capacity, larger working range, and track, many pose constraint-related issues could be resolved with the expansion of reachability and freedom of space. In addition, if multiple robots are available to integrate into a single planning group, the system can expand further about real-time joinery making and edge crafting for connection with extra degrees of freedom.

The following modules contain solid technical foundations and are available to expand immediately:

1. Scanning and surveillance sensor groups

   The current physical system depends on a singular sensor, which has to perform an assembly scan, input branch scan, pick, and place in a linear sequence. By setting up multiple scanning sensor groups and surveillance sensor groups that monitor real-time changes inside the working scene and the evolving structure, efficiency will greatly increase and the scene allow abrupt intrusion, and adaptive changes in trajectory if other dynamic objects engage in the process.

2. Parallel detection and operation
   As addressed for the limitation of the current physical setup, only one camera and one robot manipulator are available to perform all the material scans and collision detection during the non-operation period, which greatly hinders efficiency and subsequently results in slower assembly per branch. Through the documentation of the prototyping, every single branch takes from 15min-20 mins for a full cycle. However, with an advanced setup with multiple sensors, both human operation, robot operation, and collision detection can perform parallelly to increase efficiency and potentially contribute to better integration between human-robot collaboration. Moveit enables replanning ability for trajectory during execution once an update in Octomap informs potential new collision in route, therefore even manipulators will be able to perform dynamic collision avoidance during motion. One of the potential challenges could be caused by the speed of the Octomap, as a small test of Octomap update speed was performed by using real sense pointcloud input. The test showed slow update frequency (>5s) and therefore might require either parameter tuning or a decrease in Octomap resolution. Moveit also allows the setup of multiple robots in a single move group and may plan trajectories simultaneously, this is an exciting aspect that could be realized to demonstrate a distributed robotic fabrication system soon.
3. HRI interface
   Currently the most approachable HRI interfaces in the framework are Rhino and Grasshopper which allows designers to assign design instructions with curves, surfaces, and parameters for the manipulator to plan and execute. There is no intuitive interface yet like gesture motion capture or computer vision to receive an immediate response from the human operator. The framework is set up to mostly fulfill high-precision and careful design decisions by designers. However, an intuitive HRI interface could potentially add interesting interaction between the operator and robot manipulators. Considering the open integrative nature of ROS, an additional HRI interface could serve as an assistive element, driven by microcontrollers in the future.

Bibliography

"Abb_driver - ROS Wiki." Accessed December 18, 2022. http://wiki.ros.org/abb_driver.

"Before Building." Accessed November 10, 2022. https://before-building.com.

Cao, Junjie, Andrea Tagliasacchi, Matt Olson, Hao Zhang, and Zhinxun Su. "Point Cloud Skeletons via Laplacian Based Contraction." In *2010 Shape Modeling International Conference*, 187–97. Aix-en-Provence, France: IEEE, 2010. https://doi.org/10.1109/SMI.2010.25.

Casas, Gonzalo, Romana Rust, Beverlylytle, Anton Tetov, Matteo Pacher, Kathrin Dörfler, Chen Kasirer, et al. "Compas-Dev/Compas_fab: COMPAS FAB v0.27.0." Zenodo, October 10, 2022. https://doi.org/10.5281/ZENODO.3469478.

Coleman, David T., Ioan A. Sucan, Sachin Chitta, and Nikolaus Correll. "Reducing the Barrier to Entry of Complex Robotic Software: A MoveIt! Case Study," 2014. https://doi.org/10.6092/JOSER_2014_05_01_P3.

Durrant-Whyte, H., and T. Bailey. "Simultaneous Localization and Mapping: Part I." *IEEE Robotics & Automation Magazine* 13, no. 2 (June 2006): 99–110. https://doi.org/10.1109/MRA.2006.1638022.

Eversmann, Philipp, Julian Ochs, Jannis Heise, Zuardin Akbar, and Stefan Böhm. "Additive Timber Manufacturing: A Novel, Wood-Based Filament and Its Additive Robotic Fabrication Techniques for Large-Scale, Material-Efficient Construction." *3D Printing and Additive Manufacturing* 9, no. 3 (June 1, 2022): 161–76. https://doi.org/10.1089/3dp.2020.0356.

Gil Pérez, M, C Zechmeister, F Kannenberg, P Mindermann, L Balangé, Y Guo, S Hügle, et al. "Computational Co-Design Framework for Coreless Wound Fibre–Polymer Composite Structures." *Journal of Computational Design and Engineering* 9, no. 2 (February 25, 2022): 310–29. https://doi.org/10.1093/jcde/qwab081.

Hack, Norman Peter. "Mesh Mould: A Robotically Fabricated Structural Stay-in-Place Formwork System." Application/pdf. ETH Zurich, 2018. https://doi.org/10.3929/ETHZ-B-000263345.

Hägele, Martin, Klas Nilsson, J. Norberto Pires, and Rainer Bischoff. "Industrial Robotics." In *Springer Handbook of Robotics*, edited by Bruno Siciliano and Oussama Khatib, 1385–1422. Springer Handbooks. Cham: Springer International Publishing, 2016. https://doi.org/10.1007/978-3-319-32552-1_54.

HANNAH. "Ashen." Accessed March 15, 2023. https://www.hannah-office.org/work/ashen.

Hooke Park. "Wood Chip Barn." Accessed March 15, 2023. http://hookepark.aaschool.ac.uk/woodchip-barn/.

Hornung, Armin, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees." *Autonomous Robots* 34, no. 3 (April 2013): 189–206. https://doi.org/10.1007/s10514-012-9321-0.

Labbé, Mathieu, and François Michaud. "RTAB-Map as an Open-Source Lidar and Visual Simultaneous Localization and Mapping Library for Large-Scale and Long-Term Online Operation: LABBÉ AND MICHAUD." *Journal of Field Robotics* 36, no. 2 (March 2019): 416–46. https://doi.org/10.1002/rob.21831.

"Layered Costmaps for Context-Sensitive Navigation | IEEE Conference Publication | IEEE Xplore." Accessed March 12, 2023. https://ieeexplore.ieee.org/document/6942636.

Lin, Wenwei, Peidong Liang, Guantai Luo, Ziyang Zhao, and Chentao Zhang. "Research of Online Hand–Eye Calibration Method Based on ChArUco Board." *Sensors* 22, no. 10 (January 2022): 3805. https://doi.org/10.3390/s22103805.

Market Research Reports® Inc. "World's Top 10 Industrial Robot Manufacturers | Market Research Blog." Text, May 8, 2019. https://www.marketresearchreports.com/blog/2019/05/08/world%E2%80%99s-top-10-ind

ustrial-robot-manufacturers.
“Realsense2_camera - ROS Wiki.” Accessed December 18, 2022. http://wiki.ros.org/realsense2_camera.
“ROS Noetic Ninjemys,” May 23, 2020. http://wiki.ros.org/noetic.
“Roslibpy: ROS Bridge Library — Roslibpy 1.3.0 Documentation.” Accessed December 15, 2022. https://roslibpy.readthedocs.io/en/latest/.
“Rtabmap_ros - ROS Wiki.” Accessed November 10, 2022. http://wiki.ros.org/rtabmap_ros.
Sparks, D. N. “Algorithm AS 58: Euclidean Cluster Analysis.” *Applied Statistics* 22, no. 1 (1973): 126. https://doi.org/10.2307/2346321.
Sucan, Ioan A., Mark Moll, and Lydia E. Kavraki. “The Open Motion Planning Library.” *IEEE Robotics & Automation Magazine* 19, no. 4 (December 2012): 72–82. https://doi.org/10.1109/MRA.2012.2205651.
Wu, Kaicong, and Axel Kilian. “Designing Natural Wood Log Structures with Stochastic Assembly and Deep Learning.” In *Robotic Fabrication in Architecture, Art and Design 2018*, edited by Jan Willmann, Philippe Block, Marco Hutter, Kendra Byrne, and Tim Schork, 16–30. Cham: Springer International Publishing, 2019. https://doi.org/10.1007/978-3-319-92294-2_2.

Appendix

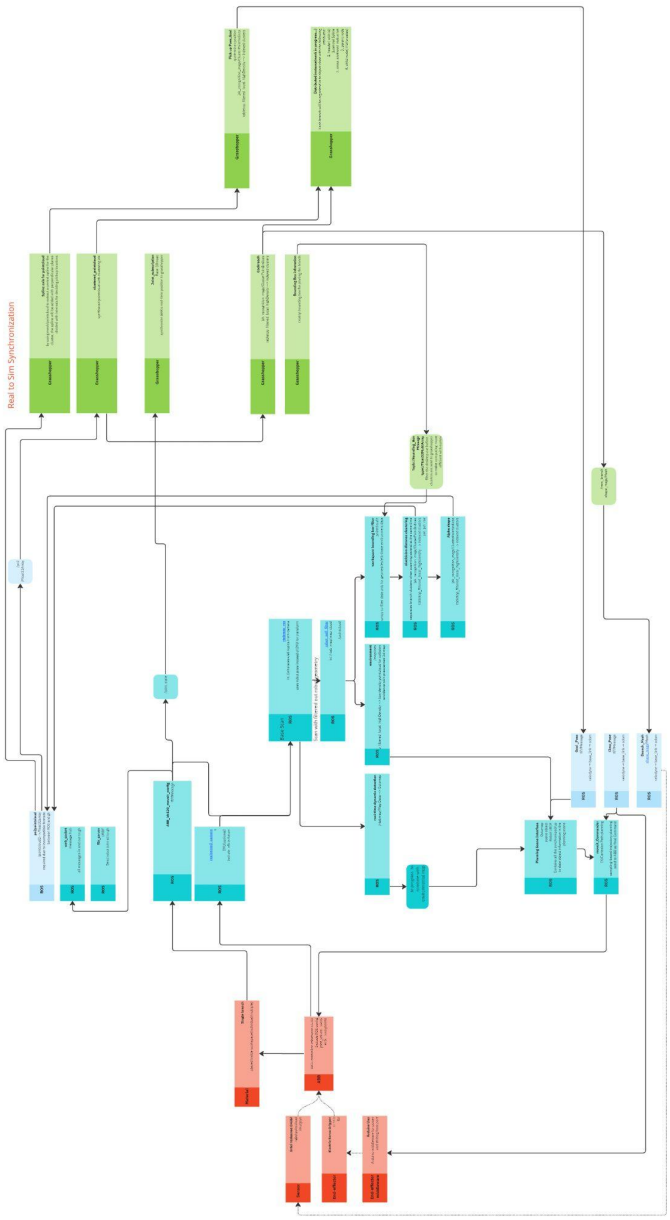


Fig.45 System Atlas

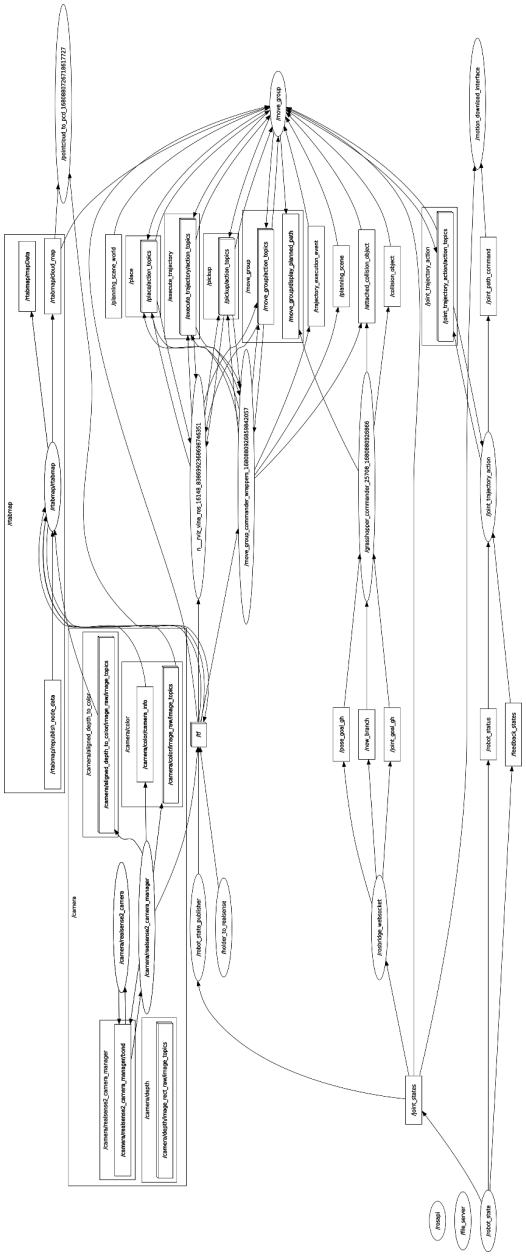


Fig.46 ROS rqt_graph during operation, listing all nodes and topics relationship