Robotic Fabrication in Architecture, Art and Design 2018
Towards Automatic Path Planning for Robotically Assembled Spatial Structures

Augusto Gandia (E), Stefana Parascho (E), Romana Rust (E),
Gonzalo Casas (E), Fabio Gramazio (E), and Matthias Kohler (E)

ETH, Zurich, Switzerland
{gandia, parascho, rust, casas, gramazio, kohler}@arch.ethz.ch

Abstract. This paper discusses the integration of automatic robot path planning into the computational design environment. A path planning software interface is presented that allows to support fabrication-aware design of robotically assembled structures with discrete elements. Using the large-scale Robotic Fabrication Laboratory (RFL) as test-bed, the software interface is validated through three experiments, in which building members need to be guided around obstacles and which are fabricated using two cooperative robotic arms. Specific focus of this paper is the investigation of strategies to narrow the path search by adjusting design and path planning parameters in order to achieve a calculation time that is suitable for design applications. A close integration of automatic path planning and design is presented, which does not only enable the negotiation between design intention and fabrication feasibility, but allows for an understanding of the constraints present in robotically fabricated spatial structures. Thus, this research contributes to expand these structures’ design and fabrication space.

Keywords: Automatic path planning • Computational design
Spatial structures • Cooperative robotic processes

1 Introduction

Today, several robotic simulation and visualization tools for computational design environments (e.g. HAL [1], PowerMill Robot [2], MORSE [3]) are assisting architects in designing robotically fabricated structures. These tools however, require the designer to manually draw and adjust auxiliary curves (toolpaths) until the robot can follow the curves without any collision. This is a tedious and time-consuming process, especially if the structure consists of a high number of building elements that need to be maneuvered to intricate positions or into gaps. Apart from such CAD-based robot programming tools, there exist several offline robot-programming tools [4] (e.g. Moveit! [5] with Gazebo [6]) that are based on the Open Motion Planning Library (OMPL) [7]. These libraries could enable automatic path search while taking robot disruptions, such as collisions and robot axes limits into account, but are not easily accessible from within computational design environments used in architecture. Thus, this hinders architects to fully explore the design space offered through the integration of automated path planning, which limits their possibilities to work with over-simplified models of fabrication.

To solve these restrictions, this paper introduces a Python software interface, which connects the design environment Rhino/Grasshopper [8] with specific capabilities of the OMPL. It is developed and evolves through several experiments, from a tool for post-rationalization of a computationally designed structure, towards a tool that can be tightly integrated in the structure’s generation. This enables fabrication-aware design, which in particular considers cooperative robotic assembly with obstacle avoidance. The research is conducted in collaboration with the projects “Multi-Robotic Prefabrication of Spatial Lightweight Structures” [9] and “Spatial Timber Units” [10], which physically validate the approach through three experiments and their respective 1:1 prototypes. Experiment 1 and 3 are conducted within the first project and Experiment 2 within the second project. Using the large-scale multi-robotic laboratory RFL [11], the projects build upon the same assembly logic, in which two cooperating robotic arms alternate their function along the fabrication process. While one robot supports the structure temporarily by holding a building member, the other robot positions the next member (see Fig. 1), and conversely in the following step [9]. After placement, the member is manually joined to the existing structure through either welding (Experiment 1 and 3) or screwing the connection (Experiment 2).

Fig. 1. Diagram of a cooperative robotic assembly process showing robot 1 avoiding collisions with robot 2 and with the built structure in order to assemble a building member (blue), while robot 2 stabilizes the structure.

© Springer Nature Switzerland AG 2019
https://doi.org/10.1007/978-3-319-92294-2_5
2 Path Planning Software Interface and Integration of Collision-Free Robot Path Generation into Computational Design

The main challenge in the development of the software interface between the computational design environment and the path planning setup is the achievement of a calculation time suitable for design applications. It is also important to define the interface’s in- and outputs, which consist of path planning parameters controlled from within the design environment, and of information from the automated path planner that is useful to feed back.

The software interface builds upon a library (OMPL) containing algorithms for robot path planning (e.g. RRT [12] and RRT* [13]), which, however, is independent of a specific collision checker or visualization tool [7]. Therefore, the software interface accesses the library through the simulation platform V-REP [14], which extends the functionality with fast collision detection through the definition of collidable entities [15]. A GhPython component [16] in Rhino/Grasshopper allows to interact with the software interface. Additionally, Grasshopper can be used to visualize the robot paths calculated by the planner. At a later moment in the development (see Experiment 2), the software interface was enhanced with simultaneous search capabilities through containerized instances of V-REP using the Docker platform [17] and a TCP/IP coordinator (see Fig. 2). This allowed to improve the workflow by increasing the path search success rate and contributed to the software interface’s practicality, since V-REP could run as headless software.

In order to simulate the pick and place assembly process of a computationally designed structure within a specific robotic setup, several parameters have to be set in advance. This includes (1) the computational representation of the robotic setup, (2) OMPL path planner parameters and (3) design-derived parameters.

In V-REP, the robotic setup (see Fig. 2) is defined by collision meshes including scripts that access the geometrical objects programmatically by specifying their joints and respective types in a hierarchical structure [18]. In the presented experiments, the robot model consists of four six-axis robots (ABB IRB 4600-40/2.55) mounted on a gantry system (Güdel) that adds three linear axes to each robot, resulting into nine DOF. Additionally, the geometry of the respective end-effector including its tool centre point (TCP) has to be defined.

Through the software interface it is possible to control three OMPL parameters (see Fig. 2): the name of the path planning algorithm, the collision checking resolution and the robot axes metrics. In OMPL, the 25 available path planning algorithms are categorized into geometric planners and control-based planners [19]. For the presented experiments, only geometric planners have been used since it was only necessary to consider the geometric and kinematic constraints of the system and not its dynamics (e.g. acceleration of the robot). The distance between the steps on a calculated robot path derive from the parameter collision checking resolution [20]. Furthermore, the robot axis metric can be defined for each joint as a floating-point value between 0 and 1 (from not constrained to blocked), which allows to constrain its movement.

Finally, design derived parameters (see Fig. 2) include the building member’s geometry, which is picked at a defined gripping position with a start configuration (robot’s joint and axes values) by the robot name (as defined in the robot model) and placed at the end pose (frame of reference). Additionally, other collision meshes can be added for collision checks, such as for example the representation of the already assembled structure. If the search with the described parameters is successful, the collision-free path, which consist of angular and translational values (joints and linear axes) can be visualized in Rhino/Grasshopper. On the contrary, if the path search is not successful, a message indicates if the problem occurred during the initialization of the planner (e.g. the robot in the start configuration is colliding with an obstacle), or if no solution was found due to collisions along the path between the start and the end configuration. Once all feasible paths are calculated, fabrication starts by sending each path via a custom robot control interface to the IRC5 ABB robot controller, which executes the received commands.

Finding collision-free paths is a time-intensive task since it is a highly complex problem [21] that requires to recurrently check for collisions [22]. Therefore, independent of the path planning algorithm, the resolution of the meshes, or the collision checking resolution, it is important to evaluate if design parameters can already constrain the search space of the path. Therefore, in this paper a strategy is presented, in which the path planner does not calculate the full path from the building-members’ picking station, to its final position in the structure, but only to a certain offset from it that will be later referred to as insertion pose(s).

---

1 The software interface could also be integrated in any CAD software that includes a Python interpreter (e.g. Blender [24] and Maya [25]) or through frameworks supporting algorithmic design (e.g. COMPAS) [26].

2 Flexible parts such as cables and hosepipes are not included in the model.
While for certain computationally designed structures path planning can be applied as post-processing step for fabrication (see Experiments 1 and 2), for other structures, deadlocks might occur. This means that for a given geometrical configuration no possible path can be found, which would require redesigning/regenerating the complete structure and starting the path search from the very beginning. Experiment 3 presents a case, in which the path planner is already called in the design generation process, after defining a part of a structure. This approach allows reacting locally by adjusting the geometry of one building element once a path search was not successful, thereby omitting deadlocks through fabrication-aware design.

3 Experiment 1: Post-rationalization of Spatial Metal Structures Using Sequential Robot Path Search

The first experiment presents a workflow for post-processing the computational design of a spatial metal structure to enable its robotic construction using two cooperating robotic arms. The structure is generated in a computational design setup [9] and comprises 72 tubes (tube size: 16 mm $\varnothing$, 1000–1200 mm length) positioned in space that are connected through a reciprocal interlocking node, creating 23 non-regular tetrahedra that concatenate into a spiral configuration (see Fig. 3).

![Fig. 3. Visualization of placing one tube into the spatial metal structure indicating robot start configuration $i$, insertion pose $F_1$, final pose $F_2$](image)

For each tetrahedron, the computational design tool assigns a preliminary fabrication sequence, which describes the order of how to stably assemble the tetrahedron. An insertion pose (see Fig. 3, $F_1$) is calculated for each metal tube, representing the end pose of the path planner's robot path. The insertion pose is a robot end-effector position in an offset (20 cm) to the structure that defines the last section of the robot path before reaching the building member's final position. It is calculated by taking possible collisions of the inserted building member with the existing structure into account. In the fabrication process, the movement from the insertion pose to place the tube in its final pose (see Fig. 3, $F_2$) is performed through a linear motion in the robot's tool space. Despite this movement is not considered in the collision check of the planned path, collisions were not registered during the experiment.

The procedure to validate the structure's fabrication feasibility consists of iterating over the tetrahedra along the fabrication sequence and searching for collision-free assembly paths for the tube within each tetrahedron (see Fig. 4), considering the above-mentioned robot alternation of stabilizing the structure and placing a building member.

Each path starts with the robot start configuration (see Fig. 3, $i$) for picking the pre-cut tubes and ends at the robot insertion pose (see Fig. 3, $F_1$). Within the path search, the path planner does not only check for collisions of the manipulated building member with the existing structure, but also with the other robot that is temporarily stabilizing the structure. The path search is automatic, however, assigning the respective robot to each building member, selecting the next tube within the fabrication sequence and changing the algorithm name, etc. is performed manually.

![Fig. 4. Diagram of the parameters and processes involved during the sequential path planning for the first prototype](image)

Finding paths is occasionally not possible, for example, if the selected robot workspace is highly constrained by the other robot's configuration or the insertion pose is almost inaccessible. Therefore, it is necessary to change the path planner's parameter settings and restart the search. Preliminary empirical tests hint towards the use of fast algorithms, such as RRT, for paths planned in a less constrained search space and algorithms, such as SBL [19], for intricate paths for which the robot can hardly manoeuvre building members without colliding. However, changing only the algorithm does not always lead to success. In order to counteract such deadlocks it is also necessary to adjust the fabrication sequence, the gripper position/orientation along the tube and the insertion pose as well. In this experiment the total of 72 paths were found by a combined manual adjustment of the fabrication sequence, the gripping position and the insertion poses.
The successful realization of the structure proved the usability of the software interface to find collision-free assembly paths for a multi-robotic spatial assembly process (see Fig. 5). However, during the experiment several problems could be identified that require further improvements. The workflow was very slow due to the considerable amount of manual steps that required multiple adjustments until finding valid paths.

In average, it took 30 min of testing and adjusting different parameters settings to find one path, while the path’s calculation took approx. 30 s. Nevertheless, using the strategy of the insertion pose, rather than the final pose of the building member, proved advantageous since it narrowed down the search space of the path planner, by avoiding to calculate the insertion move from the insertion pose to the building member’s final position. A test comparing the use of the RRT algorithm to calculate paths to the final pose (see Fig. 3, $F_2$) and to the insertion pose (see Fig. 3, $F_1$) showed that the calculation time could be reduced from 487 s in average to 34.86 s in average. Consequently, this approach reduced the amount of trial and error to find a collision-free path and speeded up the entire path planning process. Therefore, this strategy was further used in the following experiment in a different design and material system.

4 Experiment 2: Post-rationalization of Spatial Timber Frame Structures Using Parallel Path Search

As discussed in Experiment 1, finding collision-free paths depends on multiple parameters and often requires multiple trial and error steps that considerably slow down the workflow. Thus, the path planner was enhanced with simultaneous search capabilities, which allowed checking in parallel for several paths (two paths per processor core) with different parameter settings. The validity of this approach was tested in the Experiment 2, which investigates the post-processing of a spatial timber structure to enable its construction. The structure represents 81 freely oriented beams of different lengths (2000–4600 mm) and cross-sections (200 × 80 mm, 120 × 100 mm, 120 × 60 mm).

Besides the geometry of the structure (meshes oriented in space), and the additional robot poses for cutting, drilling and milling the beams, the computational design model calculates a preliminary fabrication sequence and an insertion pose for each beam. The fabrication sequence describes to assemble the floor (beams $c_1$–$c_3$) first, then the walls (beams $a_1$–$a_3$, $b_1$–$b_3$, $c_4$ and $c_5$) and finally the ceiling (beam $c_6$) (see Fig. 7). In this experiment, all beams can be placed with only one robotic arm, except for the two corner triangulations (beams $a_1$–$a_3$ and $b_1$–$b_3$), which require a cooperative assembly process with two alternating robots. Similar to Experiment 1, the insertion pose is calculated with an offset to the structure (20 cm) and describes the last section of the robot path before reaching the building member’s final position (Fig. 6).

In the path search process the parallelization can be used to calculate the assembly paths for beams $c_1$–$c_6$ all at once, since they are placed with only one robotic arm and the other robot does not need to be considered as a potential obstacle. The robotic paths to assemble the beams of the corner triangulations ($a_1$–$a_3$ and $b_1$–$b_3$) have to be sequentially searched for, considering the fabrication approach with the two alternating robots (see Fig. 7).

The construction of the timber prototype (Fig. 8) validated the approach for post-rationalizing the spatial structure, but in contrast to Experiment 1, for a different construction system with elements of different size and geometry. The average calculation time for a sequentially planned path was 22.5 s. This however cannot be compared with Experiment 1 due to different parameters settings and differently sized building elements. In a separate benchmark test 103 beams were tested by searching 4 paths per beam for which at least one valid path was found. Therefore, the parallel search speeded up the general planning workflow by saving several manual trial and error steps. This strategy could be further developed in an automated approach for
Fig. 7. Diagram of the parameters and processes involved to sequentially and parallel planned robot assembly paths for the beams of Experiment 2.

Fig. 8. Physical prototype of Experiment 2 (left) and a detail of the cooperatively assembled corner triangulation (right) with screwed t-butt joint.

Fig. 9. Possible design adjustments for one element (blue) showing potential poses $F_1$, $F_2$, $F_3$ for a tube, respective connecting points ($[b_1, b_2]$, $[c_1, c_2]$, $[d_1, d_2]$) and vertex position $p$ that can be tested in order to find a valid path.

The computational design tool iteratively generates only three tubes with the respective robot insertion pose and a preliminary fabrication sequence before running the path planner. The insertion pose defines the position and orientation of the gripper at a distance of 20 cm before reaching the final position of the building member. This is calculated considering the movement to interlock a tube in the reciprocal connection without colliding with the structure and with the other robot.

Similar to Experiment 1, solution paths are searched for each tube in the order of the fabrication sequence inside the tetrahedron considering the robot alternation (see Fig. 10). However, this time the adjustment of parameters is automated, except for adjustments of the insertion pose. Also, the path planning algorithm was not changed since SBL provided a high success-rate for intricate path planning with high resolution. If even the automated search was not successful, the designed geometry was adjusted. In the context of this experiment this was performed manually through changing design parameters of the tetrahedron, such as the vertex (the approximate position of the tubes' meeting point, see Fig. 10, $p$) or the connection points (see Fig. 10, $[a_1, a_2]$, $[b_1, b_2]$, $[c_1, c_2]$), which are geometrically defined within the design script by considering requirements of the connection system.

At a later state of implementation, it is foreseen to automate these changes.

The experiment shows the potential of using the software interface to explore fabrication-aware design of cooperatively assembled spatial structures. It exemplifies

---

3 Adjustments of the design do not take structural considers into account.
that for a certain structural complexity the early integration of path planning in the design process is inevitable and contributes to expand the design space of feasible structures.

The strategy of automating fabrication parameter adjustments was successful and allowed to reduce the required manual input. Twenty-three of the thirty-three paths were found automatically through adjustments in the fabrication parameters, such as fabrication sequence and gripping position on the tube. In four cases, the path search had to be paused since no path was found, but could continue after adjusting the insertion pose. Finally, six tubes required changes in the overall design, since no other measures led to a successful path generation. The changes involved choosing one of the different connection pairs (see Fig. 10, \([b_1, b_2], [c_1, c_2], [d_1, d_2]\)) which allowed to find a valid path. Further advancement could comprise the integration of evaluation criteria for the effects of parameter changes to speed up the search with more effective adjustments. Additionally, a modular setup of the search procedure could facilitate a faster identification of high-impact parameters and optimize the number of iterations needed to find a path (Fig. 11).

**Fig. 11.** Physical prototype of Experiment 3 (left) and detail of the reciprocal welded node (right)

6 Conclusion

The presented work demonstrates the use of automatic path planning for enabling the robotic assembly of spatial structures showcased through three experiments. The fabricated prototypes built with different assembly logics (individually and cooperatively assembled members) and consisting of building members in varied scales, show its successful application for constructive systems with linear building members, thus expanding the design and fabrication space of robotically assembled spatial structures.

However, several aspects of the path planning interface still require further development. While over the course of development the software interface was already enhanced by simultaneous search capabilities (see Experiment 2) and the possibility to automate parameter combinations (see Experiment 3), the performance could be improved further. Instant design adjustments according to the interface's feedback requires a more radical reduction of calculation time. This could be pursued through the mentioned strategy for the identification of high-impact parameters for specific design scenarios. Additionally, the definition of multiple optimization objectives for the optimal planners available in OMPL (e.g. RRT* and PRM*) would allow to automate the adjustments of the OMPL parameters. More radical approaches towards improving performance would include the optimization of OMPL path planner algorithms through fine-grained parallelism [23]. Additionally, to improve the interface's usability, the ability to specify the robotic setup from the computational design environment would allow using the software without any knowledge of V-REP.

The conceptualization and materialization of robotically assembled spatial structures that are truly aware of fabrication require the consideration of tolerances that sum up not only vertically as for layer-based assemblies, but also in multiple directions. In Experiment 2 sensors measured deviations of material deformations and of the structure during fabrication. These tolerances were fed back into the design environment and the design was adapted accordingly. Topic of further investigation is to simulate tolerances and to visualize deadlocks in the design in order to investigate strategies to counteract thereof.

**Acknowledgements.** This research presented here is part of and supported by the research programme NCCR, funded by the Swiss National Science Foundation. Special thanks goes to Ammar Mirjan for his advice during the early stages of the PhD Fund and for the people that contributed to the realization of the first experiment: Gergana Rusenova, Petrus Aejmelaeus-Lindström, Marco Palma and Martin Rusenova. Also thanks for the people that were part of the second experiment: Andreas Thoma, Arash Adel and Matthias Helmreich. Additional thanks goes to Marc Freese for collaborating as consultant of robotic simulation and for the integration of OMPL in V-REP and to Philippe Fleischmann and Michael Lyrenmann for their support in the RFL.
References