# Assembly Path Planning for Stable Robotic Construction

Michael McEvoy, Erik Komendera and Nikolaus Correll

***Abstract*— We propose an algorithmic approach for assembly path planning that takes stability of the structure during construction into account. Finite Element Analysis (FEA) is used to evaluate the intermediate stages of the assembly for stability. The algorithm presented here assembles a structure by greedily taking the most stable option at each step in the assembly process, and has complexity $O(n!)$, albeit most structures are effectively assembled with complexity $O(n^2)$. We demonstrate the workings of the proposed hybrid discrete/FEA search algorithm in simulation on a series of truss structures. In particular, we show that the algorithm is able to identify correct orderings that led to stable assembly, and discuss structures for which a greedy approach with scaffolding might be advantageous over a complete approach.**

## I. Introduction

Autonomous assembly of structures is an attractive application for autonomous robots. In order to effectively assemble a structure, robots must not only plan an assembly sequence, but also whether intermediate assemblies are stable, an aspect that is mostly neglected in the literature. This paper describes a complete, greedy approach to assembly sequence planning in robotic construction applications that combines discrete search over the possible assembly sequences with physics models using established Finite Element Analysis (FEA) tools.

Current research typically ignores external forces and focuses on structures that are statically stable at all intermediate construction steps so that internal structural forces can also be ignored. Ignoring forces such as gravity limits the types of structures that can be successfully built by robotic assembly, however. Cantilevered structures, arches, and bridges are a few examples where internal and external forces strongly influence the construction sequence. These structures also illustrate the importance of being able to recognize when scaffolding is needed to complete the structure. Our greedy approach finds such instabilities in the construction sequence and performs a complete search over the space of possible assemblies.

The work described herein complements robotic construction algorithms in current literature and combines them with proven methods to evaluate static stability. Where as the proposed greedy algorithm for stable assembly can possibly work with any suitable physics simulation, e.g. to simulate the impact of wind or vibrations on the structure, this paper utilizes FEA to compare the stability of different assembly sequences with respect to gravity.

Department of Computer Science, University of Colorado, Boulder, Colorado 80309-0430 [1]michael.mcevoy@colorado.edu

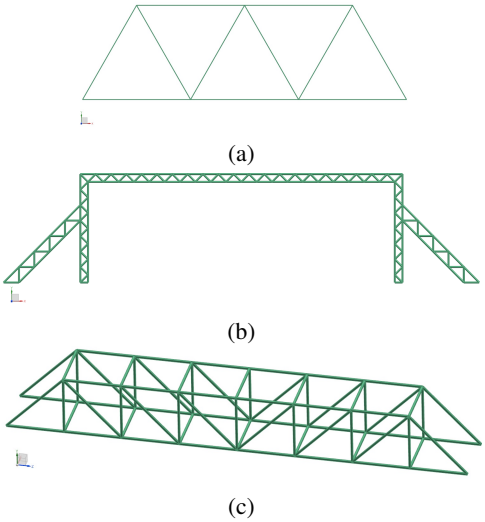


Fig. 1: Truss structures explored in simulation. (a) A simple 2D Warren style bridge, (b) a 2D example of a supporting structure for a steel building and (c) a short 3D Pratt style bridge.

### A. Related Work

Robotic assembly builds up on three distinct bodies of work: 1) the generation of an assembly path, 2) the control of the robots that will assemble the structure, and 3) the development of a robot system that can perform the construction.

The generation of automatic assembly sequences has been a topic of study in the field of manufacturing for many years. In general, assembly planning is PSPACE-Hard [1]. Algorithms to automatically generate these sequences and reduce the use of excessive fixturing already exist. Finding assembly sequences by reversing the disassembly sequences is used in many approaches, since disassembly sequencing with only geometric constraints does not require backtracking [2]. AND/OR graphs [2] assume that subassemblies can be made independently, which is only a valid assumption if each subassembly is connected to the ground. Complete and correct algorithms cannot escape the worst case $O(n!)$ for enumerating all build sequences [3], rendering these algorithms useful only for structures with tens of parts. Partial ordering [4] may be used reduce the search space for assemblies, but is still exponential in the worst case.

A* search with pruning [5] is shown to find optimal paths subject to identifying good heuristics. Non-directional blocking graphs [6] can find assemblies in polynomial time where assembly steps are valid if they can be moved into place without violating constraints. The difficulty of solving

PSPACE-Hard problems made this type of problem fall out of fashion, but interest was renewed with the popularity of distributed robotics and stochastic methods, including genetic algorithms [7], [8] and probabilistic roadmaps [9]. A robust distributed algorithm based on opportunistic disassembly sequencing is described in [10], which uses teams of robots to assemble structures while handling exceptions due to a wide range of failures, and relying only on a human operator when failures are beyond the scope of the assembly robots. The main drawback in this body of work is the simplifying assumptions made about stability. Instead, the focus usually lies on constraints such as whether there is room to maneuver parts into their desired locations, and stability is abstracted from existing connections. For example, a connection to another member that has been previously assembled ensures the stability of the part being added, and stability relations are assumed to be local and independent of external elements on the structure [2], [4], [11].

Robotic self-assembly and reconfiguration algorithms, such as [12]–[14] tend to neglect gravity and other forces that would normally act on the structure or they focus on structures that are inherently stable. The risk with approaches such as these is that, if applied to a structure under an arbitrary loading condition, invalid assembly sequences could be generated that would destroy the structure during the construction process. For example, [15] describes a control method for a team of robots manipulating large flexible space structures. The robots are able to control the flexible components of the assembly, however, the displacements and forces that the robots impart to the structure are not considered with respect to the structural stability of the assembly. In addition to controlled movement, structural integrity of the assembly must be maintained. More recently, distributed field assembly by heterogeneous robots have been explored in large-scale assembly tasks. Three different robots are shown to successfully dock a part to an assembly [16]. An experiment performed at JPL demonstrated the precision assembly of beams by a pair of cooperative robots using highly rigid motions to ensure precision [17].

A few examples of robots developed to climb around, build, and manipulate truss structures are found in [14], [18], [19]. From these references, the most notable contributions are from [14], [18] which, respectively, demonstrate the construction of truss structures using a team of quadrotors and a humanoid robot. In the first example, the team of quadrotors builds a cubic truss structure that is inherently stable at each step of the process. In the latter example, the humanoid robot builds a truss structure under "shared control," a combination of teleoperation and assignment of low level functions to the robot. In addition, a robot that can assemble IKEA furniture, with only geometric constraints, is documented in [20] and swarms of assembly robots are shown to reliably assemble structures, while walking upon them, using only information found in the environment [12], [21]. None of these works consider stability during construction. The work that most resembles ours is seen in [22], where a robot successfully deconstructs a tower of rectangular blocks. The blocks to be removed are chosen by a set of heuristics and the choices are analyzed for stability with a physics-based simulation.

### B. Contribution and Outline

The contribution of this work is to introduce a method that determines the static stability of a structure at intermediate stages of construction by combining discrete search with FEA, and uses this information to greedily choose an assembly sequence. This capability will complement the three research areas listed above by 1) allowing for the generation of an assembly sequence that can identify the stability at each step, and 2) allowing for robot control and assembly operations that interact with the structure in a stable manner.

The next section details definitions used in the description of the stable assembly algorithm. After a description of the algorithm, the algorithm is evaluated in simulation.

## II. Definitions

A *structure* is a connected undirected graph $G = \langle V, E \rangle$. The vertices, $V = \{b, v_1, v_2, \cdots, v_n\}$, are the boundary of the environment $b$, and the nodes of the structure $v_i$. The boundary is the surface on which the structure is built, a table or the ground for example. All nodes sharing an edge with $b$ are considered to be valid starting points for construction. A truss structure is a structure constructed with straight elements called struts, represented abstractly as edges, $E \subseteq V \times V$, in a graph. The joints at which the struts are connected to each other are represented as the vertices of the graph. We chose this representation, as this allows us to limit graph construction to the addition of edges in the form of struts, which can be in the form of rods.

A *substructure* $A \in G$, is a connected subgraph of $G$ such that $b \in A$. This definition considers separate assembly sites connected to the ground to be a single substructure, and only rules out "floating" partial structures. This allows construction to occur at multiple locations which will join together at a later step.

Let $\mathcal{S}$ be the set of all substructures of G. A constructability function $\phi \colon \mathcal{S} \times \mathcal{S} \to \{T, F\}$ is true (i.e., $\phi(A_i, A_{i+1}) = T$) if and only if $|A_{i+1}| - |A_i| = 1$ and $A_i$ and $A_{i+1}$ satisfy all construction constraints. The definition makes the assembly process both *linear*, allowing only one element to be added per step, and *monotonic*, not allowing the removal of any elements.

*Construction constraints* are the limitations placed on the substructures $A$ such that impossible assembly sequences do not arise. These constraints are determined *a priori*, and limit the number of stability analyses required. Impossible assembly sequences include enclosed and "sandwiched" struts. These examples are shown in Figures 2a and 2b with the struts that cannot be built shaded in gray. Construction constraints are arbitrary and depend on the properties of the chosen building materials and the robot systems assembling them. For example, a robot system may not have enough power or precision to place parts in a given sequence, so limitations on those sequences are necessary. Constraints

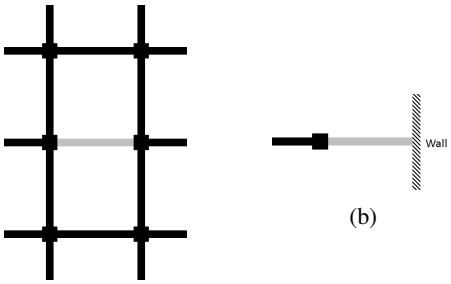


Fig. 2: Examples of construction constraints: (a) an enclosed strut inside the structure, (b) struts sandwiched between other elements or boundaries. The struts shaded in black have already been placed, blocking placement of the gray colored struts.

such as those shown in Figure 2 are common examples in the literature [12], [14]. In this paper we explicitly focus on construction of struts such as those described in [18], [19] that are free of construction constraints so that the algorithm is able to explore the complete construction space.

An *assembly path* for $G$ is a sequence of substructures $P = \{A_0, A_1, \cdots, A_n\}$ such that $A_0 = \{b\}$ and $\forall i \in \{1, 2, \cdots, |E|\}, \phi(A_i, A_{i+1}) = T$.

These definitions state that the construction of the structure must start on the boundary of the environment, that each substructure contained in the *assembly path* is created by adding one strut at a time, and that all substructures satisfy the construction constraints.

Let $\sigma(A) \rightarrow [0, \infty)$ be the instability margin for a substructure where a structure is stable if $\sigma(A) < 1 = \sigma_{limit}$, and is unstable otherwise.

The stability of a structure is defined by a set of limits placed on the displacements of the nodes, and limits placed on the resulting joint forces when when the structure is subjected to external loading. Let $M = \{\sigma(A_0), \cdots, \sigma(A_n)\}$ be the instability margins $\forall A \in P$, then an assembly path $P$ is stable if and only if $\forall A \in P, \sigma(A) < 1$. These definitions are summarized in Table I.

## III. Stable Assembly Algorithm

The algorithm used to determine the construction sequence adds one element to the structure at a time. At every step in the assembly process we greedily choose the most stable option available, while keeping track of our other possible options in case the algorithm must backtrack to find a solution. The first part placed is one that anchors the structure to the ground, $b$. The algorithm first initializes the assembly path, $P$, to include the boundary. It then analyzes each candidate element, adding the one with the lowest instability margin, $\sigma(A_{i+1})$, to the assembly path. To make this choice, the algorithm examines the stability of all possible substructures in the next step, $\{A_{i+1} \in \mathcal{S} | \phi(A_i, A_{i+1}) = T\}$. For this paper, the constructability function $\phi$ is true if and only if the added element is adjacent to any of the nodes in $A_i$.

While calculating the stability of each possible addition to the structure, we keep track of the instability margin for the most stable option, $\sigma_{min}$, and the element addition that it corresponds to, $e_{next}$. We continue this procedure until the entire structure has been assembled or an instability is found. When an instability is found, the algorithm backtracks and initiates a search along a different assembly path. Once a stable assembly path has been found we output the resulting assembly path $P$ with its associated instability margins, $M$. This algorithm is shown in Algorithm 1.

TABLE I: Stable assembly algorithm definitions

| Symbol | Definition |
|---|---|
| $P = \{A_1, \cdots, A_n\}$ | A sequence of substructures (an assembly path) |
| $A$ | A valid substructure |
| $E$ | Edges of the graph (struts) |
| $G$ | Graph representation of the structure |
| $M = \{\sigma(A_0), \cdots, \sigma(A_n)\}$ | Instability margins $\forall A \in P$ |
| $\mathcal{S}$ | The set of all substructures |
| $V$ | Verticies of the graph (nodes) |
| $\phi$ | Constructability function |
| $\sigma$ | Stability function |

**Algorithm 1** Greedy assembly of a truss structure.

**Input:** The structure to build, $G$, and the stability limit, $\sigma_{limit}$

**Output:** An assembly sequence $P$ of length $\leq |E| + 1$ and the instability margins $M = \{\sigma(A_0), \cdots, \sigma(A_{|P|})\}$ for each step

$P = \{A_0\} = \{b\}$, $M = \{0\}$
$backtrack$ =FALSE, $path\_stack$ =NULL
**while** $P \leq |E| + 1$ **do**
  $\sigma_{min} = \infty$, $e_{next}$ =NULL
  **if** $backtrack$ =FALSE **then**
    $choices$=every $\{A_{i+1} \in \mathcal{S} | \phi(A_i, A_{i+1}) = T\}$
  **else**
    $P$, $choices$=BackTrack()
    $backtrack$ =FALSE
  **end if**
  **for** every $A$ in $choices$ **do**
    **if** $\sigma(A) < \sigma_{min}$ **then**
      $\sigma_{min} = \sigma(A)$, $e_{next} = A$
    **end if**
  **end for**
  **if** $\sigma_{min} > \sigma_{limit}$ **then**
    $backtrack$ =TRUE
  **else**
    remove $e_{next}$ from $choices$
    push $[P, choices]$ onto $path\_stack$
    $P_i = P_{i-1} + e_{next}$, $M_i = \sigma_{min}$
  **end if**
**end while**
**return** $P$, $M$

For structures that have a solution that does not requires backtracking, the upper bound for this algorithm is $O(|E|^2)$.

**Function 1** BackTrack()

```
1: while choices =NULL do
2:    pop path_stack
3:    if path_stack =NULL then
4:       terminate
5:    end if
6: end while
7: return P, choices
```

We assemble one element in every step $i$ of the outer FOR loop and we analyze at most $|E| - i$ possible additions to the assembly in inner loop. The upper bound on the number of stability analyses is given by Equation 1.

$$\sum_{i=1}^{|E|} |E| - i = \frac{|E|\left(|E|+1\right)}{2} \tag{1}$$

Equation 1 assumes that structures are complete graphs. While this in not the case in typical structures, a smaller asymptotic upper bound is not possible without placing further restrictions on the configuration of the structure.

Obviously, if the algorithm must backtrack to find a stable assembly sequence, the worst case running time would be $O\left(|E|!\right)$, requiring the examination of every possible build sequence.

## IV. Simulation Results

We evaluate the proposed methodology in simulation using three generic truss structures. The selected structures were analyzed for stability using the commercial FEA program *MSC.Nastran*. We have chosen three generic truss structures to evaluate our greedy construction algorithm in simulation. Each of these structures has multiple stable assembly paths. For each structure, construction is allowed to start at any of the grounded nodes. We do not make any restrictions on the travel that the robotic agents would make while assembling these structures.

The first structure is a very simple 2D model based on a Warren truss bridge (Figure 1a). This is a common structural design that can be found in many road and rail bridges crossing short spans. The bottom left and right nodes of this model are fixed to ground. For this analysis we have assumed that the construction robots have access to both sides of the bridge. The second structure is a simplified version of a truss structure used in steel buildings (Figure 1b). We sized the center portion such that construction would fail if the buttresses are not built first. The last example is a 3D model based on a Pratt truss bridge (Figure 1c). Again, this is a common bridge design used to span short distances and we have assumed that the robot agents have access to both sides of the bridge during construction.

The overall dimensions and the cross-sectional properties of the beams used in these models are discussed in more detail in Section **??**. For these structures, we assume that all connections to the ground are completely stable.

MSC.Nastran is a commercially available FEA program. This software allows the user to analyze a structure described by a set of nodes and elements that are subjected to a set of forces and restrained by a set of boundary conditions.

We model the truss structures using beam elements and do not consider construction constraints for these models.

The Warren bridge is modeled with elements that have a solid steel 25.4 mm square cross-section. The supported building truss and the Pratt bridge are modeled with elements that have a steel box beam cross-section. The supported building uses a 25.4 mm square beam with a wall thickness of 6.35 mm while the Pratt bridge uses a 101.6 mm square beam with a 6.35 mm wall thickness. These dimensions, as well as the overall dimensions for the structures are summarized in Table II.

Limits for the truss structures were derived from standard mechanics of materials equations using these cross-sectional properties and generic steel material properties. Stability for these structures was therefore based on the limits calculated by these equations for the maximum bending moment, shear force, and axial torque that the struts could handle.

## V. Discussion

The results show that the proposed greedy algorithm is able to find stable assembly paths for the selected structures without backtracking. Table III shows a summary of the number of stability analysis calls made to *MSC.Nastran* and compares this to the worst case.

In small structures, such as the Warren bridge, a majority of the elements can be reached after placing one or two elements. In these small structures we therefore must analyze almost all of the possible options. As the graph becomes less complete, $\frac{2|E|}{|V|(|V|-1)} \rightarrow 0$ , the number of cases departs from the maximum upper bound as evidenced in the case of the Pratt bridge and the Building structure.

While it is easy to imagine structures that will exhibit the worst case behavior, we do not believe such structures are of practical importance. Bridges, shelters, and towers do not share the design patterns that would lead the algorithm to its worst case performance. Some of the structures that we have analyzed even show that the average number of stability analyses may be more linear in nature. For example, the number of stability analyses made during the assembly of the building truss structure (Figure 1b) shown in Figure 3 hovers closely around 14.4 analyses per step.

While it will not be possible to derive a closer bound on this behavior for an arbitrary structure, it suggests that different classes of structures – for example bridges, buildings, or towers – may exhibit similar behavior and that the average

TABLE II: Dimensions of the selected truss structures.

| structure | Length (m) | Width (m) | Height (m) | Cross Section Properties (type) | (mm) |
|---|---|---|---|---|---|
| Warren | 1.829 | NA | 0.528 | Bar | 25.4 square |
| Building | 13.363 | NA | 3.017 | Box | 25.4 square x 6.35 |
| Pratt | 24.384 | 3.048 | 3.048 | Box | 101.6 square x 6.35 |

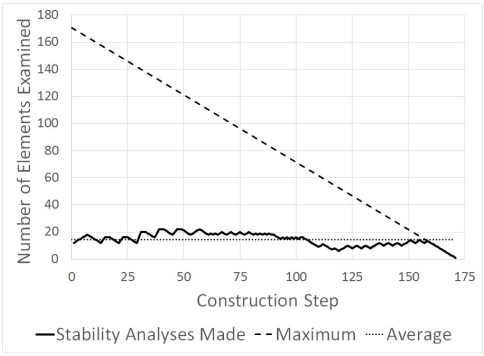


Fig. 3: Actual number of calls made to *MSC.Nastran* during each construction step of the Building truss structure.

TABLE III: Key statistics for each structure tested. The percent of stability analyses run is proportional to the completeness of the structures graph representation.

| Structure | Nodes | Elements | % Complete Graph | Stability Analyses | % of Max Possible |
|---|---|---|---|---|---|
| Warren | 7 | 11 | 52.4 | 50 | 75.8 |
| Pratt | 32 | 72 | 14.5 | 1369 | 52.1 |
| Building | 86 | 171 | 4.7 | 2470 | 16.8 |

upper bound for these structures may be much less than the $O\left(|E|^2\right)$ worst case situation.

While the greedy construction algorithm has been successful in these cases, there exist structures for which greedy construction will fail. Take for example the structure shown in Figure 4a. Greedy assembly will not be able to find a complete assembly path for this structure and must use backtracking. Greedy construction will make an incorrect choice and place the vertical elements before completing the span as they induce a smaller bending moment (Figure 4b). The proper move would be to continue the span as shown in Figure 4c. The greedy choice shown in Figure 4b leads to failure when placing the last element of the span before the two sides are connected in Figure 4d.

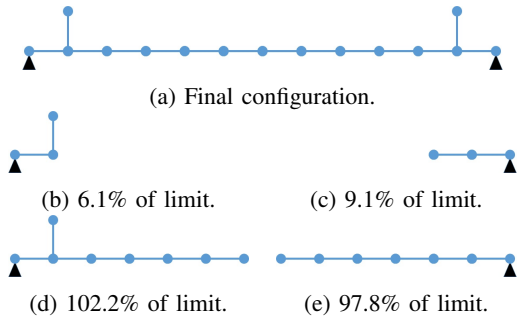


Fig. 4: A structure where greedy assembly fails to find the stable assembly path. The maximum bending moment associated with each structure is shown in the captions. The final structure (a) should be built from both sides following (c) and (e), the greedy approach incorrectly follows the path shown in (b) and (d).

Instead of performing a complete search, which might be intractable for large and complex structures, a version of the proposed algorithm without backtracking could be used to identify assembly sequences that require scaffolding. Consider an arch for example: allowing the greedy assembly to continue without backtracking, the resulting assembly sequence would show the arch to become unstable as the sides are built up, but that it returns to stable once connected. Such a sequence would show the need for scaffolding for a portion of the arch's construction. With the location of the scaffolding known it would then be possible to use the techniques outlined in [23], [24] to define the configuration of the scaffolding. Construction could then continue using greedy assembly.

Results presented in this paper use FEA to evaluate stability constraints arising from gravity. The proposed hybrid algorithm can be used with any suitable physical simulation framework, for example simulating the impact of wind gusts, earthquakes or water pressure during construction. We wish to explore such constraints in future work.

## VI. Conclusion

We presented a method for robots to reason about stable assembly paths using a greedy algorithm based on the results of a physics-based simulation of the structure's stability. A finite element analysis tool was used to evaluate the static stability of each assembly option and the most stable option was chosen at each step in the construction process. The method was demonstrated using three generic truss structures.

We were able to produce stable assembly paths for the selected structures and also showed that the number of calls to the stability analysis routine was proportional to the completeness of the structures' graph representation.

In cases where the greedy algorithm fails, a complete search of the assembly sequence space is performed. Since this might be intractable for very large structures, we believe the greedy solution should be used to identify sections of the assembly sequence that need support scaffolding during the construction. In the future, we will investigate techniques that can more reliably identify stable build paths through an analysis of the space of assembly paths: heuristics built on predicting how structures will become unstable, searching for bottlenecks, and using methods designed for exponential state spaces.

We plan to incorporate other FEA solutions into the analysis. Modal analysis could be used to evaluate the resonant frequencies of the system and then limit certain interactions with the assembly. Inertia relief solutions could be used to analyze unconstrained structures, such as structures in flight or in space. Transient response analysis could be used to determine stability under time varying loads.

We intend to test the method presented in this paper on a truss assembly experiment using the Intelligent Precision Jigging Robot (IPJR) paradigm [25], [26], the most recent version of which is shown in Figure 5. Previous IPJR experiments were performed with 2D trusses, so stability

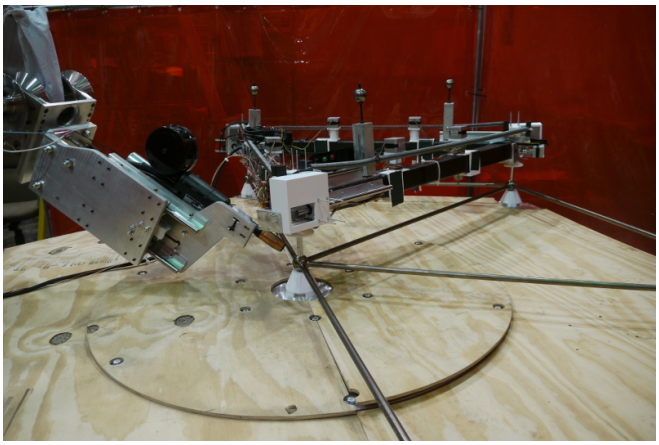

Fig. 5: Truss assembly with the Intelligent Precision Jigging Robot (right) positioning and grasping free parts while an external manipulator (left) welds the free parts to the truss.

was not a concern. However, the next prototype of the IPJR will be responsible for construction in 3D, requiring assembly planning to maximize stability and minimize nodal placement error at every step. Additional considerations will include the masses of the IPJRs, the geometric constraints prohibiting IPJRs and manipulators from placing struts on various locations on the intermediate structures, the variability of the structure's physical parameters, and the usage of real-time sensor data to modify the model and the assembly sequence.

A additional objective is to evaluate different stable assembly paths as the robot systems dynamically interact with the assembly during construction. This will also include the analysis of multi-robot systems. We are interested in characterizing subsequences that can be assembled in multiple orders, allowing distributed robots to work on different parts of the structure, possibly independently, uncertainty in the state of the structure and the environment, and handling unexpected problems including broken components and obstacles.

## ACKNOWLEDGEMENTS

This work was supported by a NASA Office of the Chief Technologist's Space Technology Research Fellowship and the Airforce Office of Scientific Research.

## REFERENCES

[1] B. K. Natarajan, "On planning assemblies," in *Proc. of the Symposium on Computational Geometry*, 1988.

[2] L. S. Homem de Mello and S. Lee, *Computer Aided Mechanical Assembly Planning*. Springer, 1991.

[3] L. S. Homem de Mello and A. C. Sanderson, "Correct and complete algorithm for the generation of mechanical assembly sequences," *IEEE Trans. on Robotics and Automation*, vol. 7, no. 2, pp. 228–240, 1991.

[4] T. De Fazio and D. Whitney, "Simplified generation of all mechanical assembly sequences," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 6, pp. 640–658, 1987.

[5] J. Wolter, "On the automatic generation of assembly plans," in *Proc. of the Int. Conf. on Robotics and Automation*, 1991, pp. 62–68.

[6] R. H. Wilson, "On geometric assembly planning," Ph.D. dissertation, Stanford University, 1992.

[7] B. Lazzerini and F. Marcelloni, "A genetic algorithm for generating optimal assembly plans," *Artificial Intelligence in Engineering*, vol. 14, pp. 319–329, 2000.

[8] G. C. Smith and S. S.-F. Smith, "An enhanced genetic algorithm for automated assembly planning," *Robotics and Computer-Integrated Manufacturing*, vol. 18, no. 5, pp. 355–364, 2002.

[9] S. Sundaram, I. Remmler, and N. M. Amato, "Disassembly sequencing using a motion planning appproach," in *Proc. of the Int. Conf. on Robotics and Automation*, 2001.

[10] F. W. Heger, "Assembly planning in constrained environments: Building structures with multiple mobile robots," Ph.D. dissertation, Carnegie Mellon University, 2010.

[11] F. Röhrdanz, H. Mosemann, and F. M. Wahl, "A high level system for generating, representing and evaluating assembly sequences," in *In Proc. of the Int. Joint Symposia on Intelligence and Systems*, 1996.

[12] J. Werfel and R. Nagpal, "Three-dimensional construction with mobile robots and modular blocks," *Int. journal of robotics research*, vol. 3-4, no. 27, pp. 463–479, 2008.

[13] S.-k. Yun, D. A. Hjelle, E. Schweikardt, H. Lipson, and D. Rus, "Planning the reconfiguration of grounded truss structures with truss climbing robots that carry truss elements," in *Int. Conf. on Robotics and Automation*. IEEE, 2009, pp. 1327–1333.

[14] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction of cubic structures with quadrotor teams," *Proc. Robotics: Science & Systems VII*, 2011.

[15] S. Dubowsky and P. Boning, "The coordinated control of space robot teams for the on-orbit construction of large flexible space structures," in *2007 IEEE Int. Conf. Robotics and Automation*, 2007.

[16] R. Simmons, S. Singh, D. Hershberger, J. Ramos, and T. Smith, "First results in the coordination of heterogeneous robots for large-scale assembly," in *Experimental Robotics VII*. Springer, 2001, pp. 323–332.

[17] A. Stroupe, T. Huntsberger, A. Okon, H. Aghazarian, and M. Robinson, "Behavior-based multi-robot collaboration for autonomous construction tasks," in *International Conference on Intelligent Robots and Systems*. IEEE, 2005, pp. 1495–1500.

[18] M. A. Diftler, J. Mehling, P. Strawser, W. R. Doggett, and I. M. Spain, "A space construction humanoid," in *Int. Con. on Humanoid Robots*. IEEE, 2005, pp. 92–97.

[19] W. Doggett, "Robotic assembly of truss structures for space systems and future research plans," in *Aerospace Conf. Proc.*, vol. 7. IEEE, 2002, pp. 7–3589.

[20] R. A. Knepper, T. Layton, J. Romanishin, and D. Rus, "Ikeabot: An autonomous multi-robot coordinated furniture assembly system," in *Proc.\ IEEE Int'l Conf.\ on Robotics and Automation, Karlsruhe, Germany, in submission*, 2013.

[21] K. Petersen, R. Nagpal, and J. Werfel, "Termes: An autonomous robotic system for three-dimensional collective construction," in *Robotics: Science and Systems VII*, 2011, pp. 257–264.

[22] J. Wang, P. Rogers, L. Parker, D. Brooks, and M. Stilman, "Robot jenga: autonomous and strategic block extraction," in *RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE, 2009, pp. 5248–5253.

[23] G. Ye and K. Ishii, "Incorporation of topology optimization capability in msc/nastran," in *Proceedings of*, 1999, pp. 155–168.

[24] S. Christodoulou, "Optimal truss design using ant colony optimization," in *5th GRACM Int. Congress on Computational Mechanics*. Citeseer, 2005.

[25] E. Komendera, D. Reishus, J. T. Dorsey, W. R. Doggett, and N. Correll, "Precise truss assembly using commodity parts and low precision welding," in *Proceedings of the Fifth Annual IEEE International Conference on Technologies for Practical Robot Applications*, 2013.

[26] ——, "Precise truss assembly using commodity parts and low precision welding," *Journal of Intelligent Service Robotics*, vol. Accepted for Publication, 2014.