Any-Time Algorithm for the Assembly of Structures with Structural and Motion Planning Constraints*

Dalan C. Loudermilk[1] and Michael W. Otte[2]

***Abstract*— For the in-space assembly of structures, there can be challenges to precomputed or teleoperated assembly solutions. These potential difficulties can be avoided by the agents reasoning in-situ about the assembly graph of the desired structure. We propose an algorithm in which the robotic agent performing an assembly considers two constraining characteristics, the first involving the structural/material properties at each step of the assembly process and the second concerning the motion/trajectory of the robot required to add an element to the partially assembled structure. Together these form an assembly sequence which represents the order in which elements are added and the associated concatenated motion plan. We propose an any-time algorithm that seeks to reconcile these two aspects of the problem while retaining the theoretical guarantees of the underlying motion planning algorithms, including probabilistic completeness and asymptotic optimality.**

Problem Context

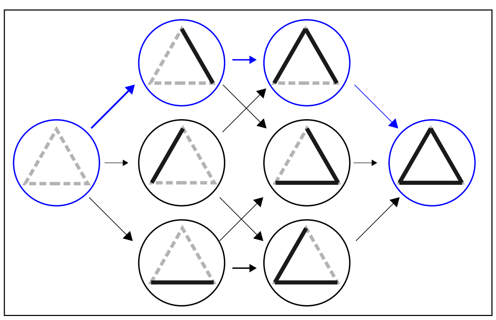

Fig. 1. Assembly graph of a planar triangle consisting of 3 elements. The set of blue edges and vertices denote an individual assembly sequence.

The assumption of being able to calculate an assembly sequence a priori is not a given for in-space assembly (ISA) tasks. Problems include communication difficulties [1], costs of teleoperators/astronauts [2], and hardware limitations [3], which must be overcome by the agents reasoning locally. Adapting to changing conditions or structural failure requires autonomous agents to replan assembly sequences using the previous assembly sequence information. Additionally, these agents can perform maintenance/servicing operations using similar logic by reversing the assembly sequences.

Assembly sequence generation can be decomposed into a series of unrelated motion planning and structural characteristics problems solved independently. However, the robot can take advantage of the well-structured nature of the assembly process to make informed decisions. Fig. 1 shows the assembly graph of a planar triangle and all possible vertices and edges, as well as a feasible assembly sequence. The primary concerns for a robot building a structure are 1) the structural properties at intermediate steps in the assembly process and 2) the actions taken by the robot in order to place an element into position. However, in most situations, these values are not known to the agent a priori.

We propose an algorithm that builds on our previous research [4] to use the structure of an assembly graph as the basis for any-time autonomous structural assembly. We believe the proposed algorithm will allow an agent to take a structural plan and find an initial sequence then refine this sequence with the remaining allotted time while retaining the theoretical guarantees of its internal motion planning algorithm.

Related Work

The field of ISA has garnered increasing interest in the last decade, for both lunar/planetary surface operations and on-orbit applications. For surface structures, there has been research into prebuilding habitats for astronauts [5], lunar power generation [6], and the construction of large-scale rail systems [7]. For on-orbit structures, the primary concerns have been focused on telescopes [8], [9], [10] and stationary platforms [11], [12], [13], [14]. Automated assembly has also received a great deal of attention, from modular structure assembly [15], [16] to other methods that try to leverage the assembly graph or structural properties in generating sequences [17], [18].

Current Research

Our previous work was limited to structures with simplified structural/vertex costs which could be precalculated for all sub-assemblies. However, sub-assembly costs are often computationally expensive in real-world situations, such as finite element analysis. We chose a vertex cost bounded set of edges to sequentially check without considering the structure of the graph. Our new work remedies those concerns by limiting the number of calls to the vertex cost function and the edge cost function, first by performing a greedier forward search to find an initial solution then switching to an exploratory search to improve upon that solution.

We present the following Algorithm 1, which takes an assembly graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V}$ is the vertex set

*This work was not supported by any organization

[1]Dalan C. Loudermilk is a PhD Student in the Department of Aerospace Engineering, University of Maryland College Park, College Park, MD, 20742, USA. dalan@umd.edu

[3]Michael W. Otte is with the Department of Aerospace Engineering, University of Maryland College Park, College Park, MD, 20742, USA. otte@umd.edu

associated with partially assembled sub-assemblies and $\mathbb{E}$ is the edge set associated with the transitional space between two sub-assemblies with a single element difference. It also takes an edge cost function $c_{edge} : e \mapsto k$ where $e \in \mathbb{E}$, k is a transitional cost associated with a trajectory through the configuration space, a vertex cost function $c_{vertex} : v \mapsto q$, where $v \in \mathbb{V}$ and q is a structural cost associated with that sub-assembly, a total time to plan t_{plan}, and $c_{vertex,max}$, the maximum structural cost. The algorithm returns an assembly sequence s^* which is the set of the vertices and edges needed to be traversed to assemble the structure. The functions in Alg. 1 `forwardSearch`, `exploreSearch`, `makeConsistent`, and `getSequence` can be considered, in order, as depth first search (DFS), upper confidence bound [19] (UCB), a standard graph update, and iteration to find vertex parents and update the assembly sequence.

Algorithm 1 Any-Time Sequencing Algorithm

Require: $\mathbb{G} = (\mathbb{V}, \mathbb{E}), c_{edge}, c_{vertex}, t_{plan}, c_{vertex,max}$
Ensure: s^*
  e.cost $\leftarrow \infty \, \forall \, e \in \mathbb{E}$
  v.cost $\leftarrow \infty$, v.costToCome $\leftarrow \infty \, \forall \, v \in \mathbb{V}$
  $s^* \leftarrow \varnothing, s_{cost} \leftarrow \infty$
  while $t_{elapsed} < t_{plan}$ **do**
    if $s^* = \varnothing$ **then**
      $e_{curr} \leftarrow$ `forwardSearch`($\mathbb{G}$)
    else
      $e_{curr} \leftarrow$ `exploreSearch`($\mathbb{G}$)
    if e_{curr}.endVertex.notVisited **then**
      e_{curr}.endVertex.cost $\leftarrow c_{vertex}(e_{curr}$.endVertex$)$
    if e_{curr}.endVertex.cost $> c_{vertex,max}$ **then**
      continue
    if $(e_{curr}$.cost $> c_{edge}(e_{curr})$ **then**
      e_{curr}.cost $\leftarrow c_{edge}(e_{curr})$
      `makeConsistent`($\mathbb{G}$)
      if $\mathbb{G}$.endVertex.costToCome $< s_{cost}$ **then**
        $s^* \leftarrow$ `getSequence`($\mathbb{G}$)
        $s_{cost} \leftarrow \mathbb{G}$.endVertex.costToCome
  return s^*

One gap in our previous work was the lack of inheritance of theoretical guarantees that are typically achieved by such algorithms, e.g., probabilistic completeness and asymptotic optimality. By ensuring that these two traits are inherited by the sequence planning outer loop, the proposed research will investigate an algorithm that evolves forward in time and checks edges and vertices as they are approached to provide an iteratively better solution. The probability of selecting any individual edge will never go to zero allowing for the exhaustive exploration of the search space given enough time.

The inclusion of revisiting vertex costs warrants investigation, especially in the context of recalculating structural costs at finer levels of detail. Careful exploration of the combined Pareto frontier of the vertex and edge spaces is necessary to do so, which will be fuzzy due to the asymptotic and thus unknown costs involved. Some notion of minimums or maximums of costs to start (such as the trivial lower bound for the motion planning problem, the distance from start to goal ignoring collisions) can be used to explore inside this Pareto frontier.

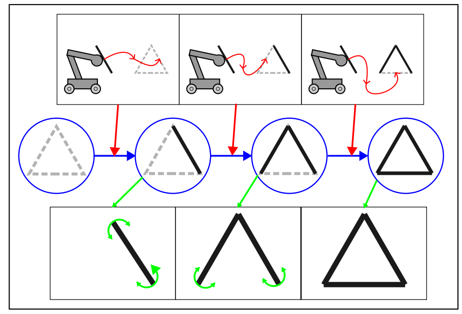

Fig. 2. For the assembly sequence colored blue in Fig. 1, the edge transitions are denoted red and vertex sub-assembly properties are denoted green. Edges represent individual motion planning problems for the placement of a single element in a configuration space given by the preceding vertex/sub-assembly. The vertices represent specific configurations of elements for sub-assemblies and their associated structural characteristics.

Our research will begin with the assumption of depth first search, but there could be advantages in using other algorithms. Once the initial solution is found, the algorithm will need to effectively balance between updating known nodes and exploring promising new nodes. If this exploratory search has weights proportional to best solutions and unexplored spaces, as long as the probability of selecting any edge never goes to zero then both probabilistic completeness and asymptotic optimality should be inherited from the edge cost function.

Our previous work and the initial work on this project employs a graph search on a directed acyclic graph (DAG). A sparse adjacency/distance matrix could be leveraged to update sequence information. This formulation might prove to be advantageous over the current DAG based search. This alternate construction allows for interesting inquiries using tropical geometry beyond the min-plus algebra used in distance matrix multiplication. An assembly graph with edge weights that have values representing probabilities could be manipulated using max-times multiplication to find the most probable assembly sequence. We are also considering other formulations that might require separate queues for each cost or an n-dimensional queue such as a KD-heap.

In addition to different data structures used to represent the problem, the authors have identified areas of interest that could use this structure to find solutions. These areas include classic simultaneous task and motion planning testbeds like sokoban and other problems that require simultaneously solving transitions between states where both have unknown values, such as critical path analysis.

We plan to run experimental trials of this initial version of the algorithm and report them at the workshop.

ACKNOWLEDGMENT

D. L. thanks Yi-Hsuan Chen and Khalid M. Jaffar for their valuable feedback on this manuscript, as well as Dr. John R. Cooper for his involvement with the previous work.

References

[1] S. A. Jefferies, D. C. Arney, C. A. Jones, F. Stillwagen, P. Chai, C. D. Hutchinson, M. Stafford, R. Moses, J. A. Dempsey, E. Rodgers, *et al.*, "In-space assembly capability assessment for potential human exploration and science applications," in *AIAA SPACE and Astronautics Forum and Exposition*, p. 5359, 2017.

[2] L. Purves, "A method for estimating costs and benefits of space assembly and servicing by astronauts and robots," in *Proceedings, IEEE Aerospace Conference*, vol. 7, pp. 7–7, 2002.

[3] M. B. Obenchain, A. Trujillo, J. Rome, J. H. Heying, A. Villegas, V. K. Goyal, and D. Taylor, "Toward a fully capable in-space manufacturing ecosystem," in *AIAA SCITECH 2025 Forum*, p. 1781, 2025.

[4] D. C. Loudermilk, J. R. Cooper, and M. W. Otte, *Assembly Sequence Optimization for Space Structures*. AIAA ASCEND, 2024.

[5] M. L. Grande, R. W. Moses, P. A. Cosgrove, R. P. Mueller, T. J. Prater, and A. J. Blanchard, "Protecting crew and surface systems with a long-duration lunar safe haven," *ASCEND 2021*, p. 4070, 2021.

[6] D. Tiffin and M. Mahlin, "Tall lunar towers: Systems analysis of lunar-surface-assembled power, communication, and navigation infrastructure," in *ASCEND 2023*, p. 4754, 2023.

[7] J. Thangavelautham and Y. Xu, "Modeling excavation, site preparation, and construction of a lunar mining base using robot swarms," in *Earth and Space 2021*, pp. 1310–1325, 2021.

[8] W. R. Doggett, O. R. Stohlman, J. Teter, K. Song, B. White, C. J. Oh, and M. Mikulas, *In-Space Modular Assembly: An Approach for Reliable, Affordable, Precision Space Apertures*.

[9] M. H. Nair, M. C. Rai, M. Poozhiyil, S. Eckersley, S. Kay, and J. Estremera, "Robotic technologies for in-orbit assembly of a large aperture space telescope: A review," *Advances in Space Research*, 2024.

[10] J. Martínez-Moritz, I. Rodríguez, K. Nottensteiner, J.-P. Lutze, P. Lehner, and M. A. Roa, "Hybrid planning system for in-space robotic assembly of telescopes using segmented mirror tiles," in *2021 IEEE Aerospace Conference (50100)*, pp. 1–16, 2021.

[11] E. Wang, S. Wu, and Z. Wu, "Dynamic multi-constrained assembly sequence planning of large space structures considering structural vibration," *Acta Astronautica*, vol. 195, pp. 27–40, 2022.

[12] J. Moser, J. Hoffman, R. Hildebrand, and E. Komendera, "An autonomous task assignment paradigm for autonomous robotic in-space assembly," *Frontiers in Robotics and AI*, vol. 9, p. 709905, 2022.

[13] X. Zhihui, L. Jinguo, W. Chenchen, and T. Yuchuang, "Review of in-space assembly technologies," *Chinese Journal of Aeronautics*, vol. 34, no. 11, pp. 21–47, 2021.

[14] R. H. Freeman, "In-space assembly of the gateway-lunar surface development and protection: Lunar operations conceptual design i.," *Journal of Space Operations & Communicator (ISSN 2410-0005)*, vol. 19, no. 2, 2023.

[15] M. A. Post, X.-T. Yan, and P. Letier, "Modularity for the future in space robotics: A review," *Acta Astronautica*, vol. 189, pp. 530–547, 2021.

[16] Y. Jiang, M. Li, D. Guo, W. Wu, R. Y. Zhong, and G. Q. Huang, "Digital twin-enabled smart modular integrated construction system for on-site assembly," *Computers in Industry*, vol. 136, p. 103594, 2022.

[17] E. Mendez, O. Ochoa, D. Olivera-Guzman, V. H. Soto-Herrera, J. A. Luna-Sánchez, C. Lucas-Dophe, E. Lugo-del Real, I. N. Ayala-Garcia, M. Alvarado Perez, and A. González, "Integration of deep learning and collaborative robot for assembly tasks," *Applied Sciences*, vol. 14, no. 2, p. 839, 2024.

[18] M. McEvoy, E. Komendera, and N. Correll, "Assembly path planning for stable robotic construction," in *2014 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, pp. 1–6, 2014.

[19] P. Auer, "Using confidence bounds for exploitation-exploration trade-offs," *Journal of Machine Learning Research*, vol. 3, no. Nov, pp. 397–422, 2002.