Attention-Based Deep Reinforcement Learning for Task Sequence Optimization of Robotic Connection Operations

Yizhe Wang[1], Cong Zhang[2], Yihai Fang[1]

[1]Department of Civil and Environmental Engineering, Monash University, Australia
[2]College of Computing and Data Science, Nanyang Technological University, Singapore
yizhe.wang@monash.edu, cong030@e.ntu.edu.sg, yihai.fang@monash.edu

Abstract

Construction robotics has significant potential to improve the efficiency, quality, and safety of structural assembly, yet its practical deployment is constrained by the limited availability of online task sequence optimization methods. This study focuses on gantry robots performing connection tasks in floor frame assembly under limited nail magazine capacity. To improve operational efficiency, the sequencing problem is formulated as a split-delivery vehicle routing problem (SDVRP). An attention-based deep reinforcement learning (DRL) model is employed to learn routing policies, and its performance is compared against an exact solver (Gurobi), a genetic algorithm (GA), and a nearest-neighbor (NN) heuristic. Experiment results show that the DRL policy achieves solutions within 2.1% of the optimal benchmark while requiring only 0.1s per instance. Additional tests across different nail capacities demonstrate robustness to distribution shifts. Overall, the results highlight the potential of attention-based DRL for fast and high-quality task sequence optimization, representing the first application of this approach to robotic structural assembly connection operations.

Keywords –

Deep Reinforcement Learning; Optimization; Construction Robot; Task Sequence; Structural Assembly Connection

1 Introduction

The construction sector continues to face sustained challenges from skilled labour shortages, worker health and safety concerns, and modest growth in productivity [1]. Within this context, structural assembly constitutes a fundamental construction activity in which discrete components are connected to form larger systems such as walls, floors, and roofs [2,3]. These operations are particularly well suited to partial or full automation using robotic systems owing to their repetitive and physically demanding nature [4].

With recent advances in robotics, gantry robots have been adopted across diverse applications such as timber prefabrication [5], rebar tying [6], furniture manufacturing [7], and shipbuilding [8], owing to their ability to transport materials efficiently across large workspaces. These robotic systems are increasingly regarded as promising platforms for automating structural assembly tasks. An example of an overhead gantry robot performing floor-frame connection operations is illustrated in Figure 1.

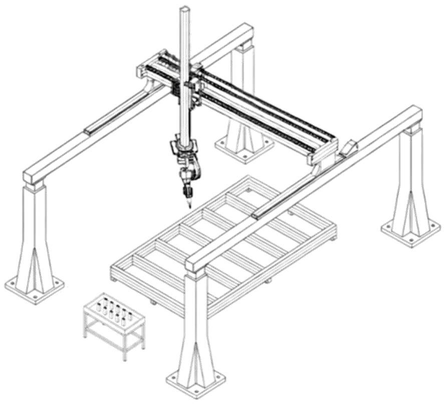

Figure 1. Example of a gantry robot performing connection tasks in timber floor-frame assembly

In practical construction settings, human workers naturally plan and adapt their own task sequences, whereas robots rely on explicitly specified task sequences to operate efficiently [9]. When gantry robots are used for connection tasks in structural assembly, their productivity depends critically on the quality of task sequence planning.

This study addresses this challenge by formulating task sequence optimization for robotic connection operations as a split-delivery vehicle routing problem (SDVRP) and solving it using an attention-based deep reinforcement learning (DRL) model. The performance of the trained policy is compared against heuristic methods and an exact solver, and its generalization capability is evaluated across varying problem settings. This proposed method achieves solutions within 2.1% of the optimal benchmark while requiring only 0.1s per instance, and maintains competitive performance when nail capacities remain close to the training dataset range.

2 Literature Review

2.1 Gantry Robot Applications in Structural Assembly

Gantry systems have long been used in the construction industry, with early applications appearing in the 1980s. During this period, gantry platforms were installed in temporary site factories on top of buildings under construction and were progressively raised as each floor was completed [10]. With continued advancements, gantry robots have been increasingly adopted for assembly operations. In industry practice, IronBot (for rebar lifting and placement) and TyBot (for rebar tying) developed by Advanced Construction Robotics are notable examples. They exhibit a high level of autonomy, including automated rebar grasping, placement, intersection detection, and tying execution [5].

In the research domain, Chai et al. presented an integrated timber construction platform incorporating a gantry-based multi-robot system, sensing, control, and a timber machining center to enable multifunctional prefabrication of timber components [6]. Wenna et al. developed a digital-twin framework for 3D path planning of a large-span curved-arm gantry robot for automated wooden-furniture production, enabling the generation of collision-free and efficient trajectories [7]. Other studies have focused on gantry robots for connection operations. For example, Saleem et al. designed a six-degree-of-freedom gantry welding robot through mathematical modelling and mechanical design [11]. Similarly, an improved RRT* algorithm was proposed for gantry-mounted welding robots to enhance planning efficiency and reliability in complex environments [8].

The majority of prior research focuses on path planning and control methods to ensure safe and efficient operation. However, few studies address task sequencing optimization, despite its importance for assembly efficiency. This gap motivates the present study on gantry robotic connection operations in structural assembly.

2.2 Task Sequencing in Robotic Structural Assembly

Although task sequencing has been studied only rarely for gantry robotic structural assembly, research on a broader range of robotic systems, including aerial robots and robotic arms, provides useful insights.

In multi-robot structural assembly, Muñoz et al. integrate a constraint-based vehicle routing problem (VRP) solver with a hierarchical task network (HTN) planner [12]. Assembly tasks are abstracted into a VRP formulation, where the VRP solver provides an initial task allocation and coarse sequencing. The HTN planner subsequently expands these assignments into executable schedules. The two planners operate in a closed feedback loop, and the method is validated through simulations of aerial robots assembling a truss-like structure. The VRP solver part relies on heuristic search strategies rather than exact optimization.

Yang et al. address assembly sequence planning for timber wall framing using a robotic arm [13]. Their method jointly considers stud placement and fastening operations while incorporating task precedence constraints. A modified A* algorithm is used to identify an executable assembly sequence, where the accumulated end-effector travel distance forms the actual cost, and the remaining cost is estimated using a minimum spanning tree (MST)-based heuristic. This approach highlights the role of domain-specific heuristics in navigating the assembly sequence planning.

You et al. explored the use of large language models (LLMs) for structural assembly task sequencing [14]. Their system, RoboGPT, leverages ChatGPT's reasoning capabilities to autonomously generate and adapt task sequences. The approach is validated using a single robotic arm executing assembly tasks, demonstrating the feasibility of LLM-driven task planning through direct application programming interface (API) integration.

To the best of the authors' knowledge, no existing studies on robotic structural assembly have employed DRL for task sequence planning. Among the various optimization methods, attention-based DRL has shown strong potential for tackling combinatorial optimization problems, in some cases outperforming traditional metaheuristic algorithms [15], and has already been explored in related manufacturing applications [16]. A further advantage of DRL is its fast inference: once trained, the policy produces near-instantaneous solutions without a repeated optimization process, enabling online adaptation to robotic structural assembly.

In summary, task sequencing research for gantry robotic structural assembly remains limited, and DRL-based optimization in structural assembly is still lacking. This study addresses these gaps by applying an attention-based DRL model [17] to an SDVRP formulation of gantry robot connection tasks in structural assembly.

3 Methodology

3.1 Problem Formulation and Abstraction

This study investigates a common application scenario of gantry robots in structural assembly, in which a robot equipped with a nailing end-effector executes floor frame connections. The robot must periodically return to a storage location for refilling after a certain number of nailing operations because the nailing end-effector has a limited magazine capacity. Consequently, the execution sequence of the nailing tasks directly determines the robot's total travel distance and, in turn, the assembly efficiency.

During assembly, the gantry robot operates within a predefined workspace free of internal obstacles, enabling collision-free motion of the end-effector. In this study, the robot is assumed to first travel in the horizontal plane from the storage location to a point vertically above the target nailing position, and then perform a vertical approach to the connection plane to execute the nailing operation. Since this vertical motion contributes a constant and negligible portion of the overall travel effort, the travel cost between the storage and each nailing task is approximated by the two-dimensional Euclidean distance in the horizontal plane.

Based on the abstraction, the robotic nailing process is modeled as an SDVRP. The SDVRP is a variant of the classical VRP defined on a set of customer nodes and a depot, in which each customer has a demand, vehicles have limited capacity, and demands may be fulfilled through multiple vehicle visits. The goal is to determine a set of routes originating from and returning to the depot that minimizes the total travel cost while satisfying vehicle capacity constraints.

In the robotic structural assembly nailing setting, the storage location is treated as the depot, the capacity of the nailing end-effector as the vehicle capacity, and each connection task as a customer node with a corresponding demand. The objective is to determine a sequence of nailing operations that minimizes the total travel distance while satisfying the capacity constraint.

Among the available methods for solving the SDVRP, attention-based DRL offers fast inference, strong combinatorial optimization capability, and promising potential for complex construction scenarios [15,17–19]. Accordingly, this study adopts an attention-based DRL framework for the formulated SDVRP.

3.2 Optimization Framework

The Transformer is an encoder–decoder architecture built around the attention mechanism, enabling highly parallelizable sequence modeling through self-attention blocks [20]. Building on the Transformer architecture, attention-based models can effectively capture global dependencies and combinatorial structure, making them well-suited to sequential decision-making under constraints. Accordingly, routing problems such as the SDVRP can be formulated as sequence generation: instance features (e.g., locations, demands, and capacity) are embedded and encoded by an attention-based encoder [21], and a decoder then sequentially selects the next node to construct one or more feasible tours [17].

While Transformers in natural language processing are typically trained in a supervised, next-token-prediction manner using large text corpora, deep models for routing problems are often trained without access to optimal solutions. Instead, these models are commonly optimized using reinforcement learning, where the policy interacts with an environment, receives a scalar reward reflecting solution quality (e.g., total route length or cost), and is updated via policy-gradient methods [15,17,22].

This study employs the attention-based transformer model proposed by Kool et al. [17] to solve the SDVRP. Since its introduction in 2019, this model has become a milestone in attention-based DRL for combinatorial optimization and has been widely applied to routing problems, such as the traveling salesman problem (TSP), the orienteering problem (OP), the capacitated vehicle routing problem (CVRP), SDVRP and the prize-collecting TSP (PCTSP). In this framework, the routing problem is formulated as a sequential decision-making process, in which a parameterized stochastic policy is learned to construct feasible solutions. The policy is implemented using an attention-based encoder–decoder neural network and optimized with the REINFORCE algorithm [23] with a deterministic greedy rollout baseline.

The following subsections describe the proposed optimization framework in detail, including the Markov decision process (MDP) formulation, policy definition, neural network architecture, and DRL training procedure in the context of the SDVRP.

3.2.1 MDP Formulation and Policy Definition

The SDVRP is formulated as a finite-horizon Markov decision process (MDP), where the agent sequentially constructs a feasible routing solution. At each decision step t, the state s_t captures the problem instance and the current routing status, including the current location, remaining vehicle capacity, and the remaining demand of each task. An action a_t corresponds to selecting the next node to visit, subject to feasibility constraints enforced by action masking. An episode terminates when all tasks are fully served, and the solution quality is evaluated by a terminal cost corresponding to the total travel distance.

Based on this MDP formulation, a stochastic policy $p_\theta(\pi|x)$ is learned to construct a complete routing solution for a given problem instance x. A feasible solution is represented by an ordered sequence of node indices $\pi = (\pi_1, \dots, \pi_n)$, where π_t denotes the node selected at decision step t. The objective of the policy is to generate a complete sequence that satisfies all problem constraints while minimizing the overall travel cost.

The attention-based model defines this policy in an autoregressive manner. Specifically, the conditional probability of generating a complete solution sequence π is factorized as (1).

$$p_\theta(\pi|x) = \prod_{t=1}^{n} p_\theta(\pi_t \mid x, \pi_{1:t-1}) \quad (1)$$

Here, $\pi_{1:t-1}$ denotes the partial sequence constructed up to step $t-1$, and θ represents the parameters of the neural network.

3.2.2 Neural Network Architecture

The neural network adopts the standard attention-based encoder-decoder architecture proposed by [17]. The encoder consists of three stacked attention layers, each comprising a multi-head self-attention sublayer followed by a node-wise feed-forward sublayer. Skip-connections and batch normalization are applied in each sublayer to stabilize training.

Each input node is represented by its spatial coordinates and associated attributes (e.g., demand), which are first projected into a shared embedding space. Through self-attention, the encoder aggregates information across all nodes, producing context-aware embeddings that capture the global structure of the routing instance.

The decoder constructs a solution in an autoregressive manner. At each decision step, it attends to the encoder embeddings and incorporates information from the partial solution constructed so far. A masking mechanism is applied to exclude infeasible actions, such as selecting already visited nodes or actions that would violate capacity feasibility. The decoder then outputs a probability distribution over all admissible nodes, from which the next node is selected.

3.2.3 DRL Training with REINFORCE

With the policy parameterized by the neural network architecture described above, the policy parameters θ are trained using the REINFORCE algorithm [23] by minimizing the expected cost of solutions sampled from the stochastic policy. For a given problem instance x, the training objective $\mathcal{L}(\theta|x)$ is defined as (2).

$$\mathcal{L}(\theta|x) = E_{p_\theta(\pi|x)}[L(\pi)] \quad (2)$$

$L(\pi)$ is a scalar cost function that evaluates the quality of a solution π. The training objective is optimized using gradient descent, which is calculated with the REINFORCE gradient estimator with a baseline in (3).

$$\nabla\mathcal{L}(\theta|x) = E_{p_\theta(\pi|x)}\left[\left(L(\pi) - b(x)\right)\nabla\log p_\theta(\pi|x)\right] \quad (3)$$

$b(x)$ is a baseline that depends only on the instance x and is used to reduce the variance of the gradient estimate. In this study, $b(x)$ is given by the cost of a deterministic greedy rollout of the current policy.

When the sampled solution π achieves a lower cost than the baseline, the term $L(\pi) - b(x)$ becomes negative, and the gradient update increases the log-probability $\log p_\theta(\pi|x)$ of the selected actions. Conversely, if the sampled solution is worse than the baseline, the update decreases this probability. This formulation supports end-to-end learning without optimal solutions or stepwise supervision.

All experiments were run on a workstation with an Intel Core Ultra 9 185H CPU and an NVIDIA GeForce RTX 4070 Laptop GPU. The DRL model was trained using a fixed set of hyperparameters throughout the study, and the main training settings are summarized in Table 1.

Table 1. Training hyperparameters of the attention-based DRL model

Hyperparameter	Value
Dataset size	1 million instances per epoch
Number of epochs	100
Learning rate	$1e^{-5}$
Batch size	512
Optimizer	Adam
Training duration	Approximately 100 minutes

4 Case Study

4.1 Application Scenario

This study applies an attention-based DRL approach to a representative floor-frame nailing task in structural assembly, with the aim of demonstrating both its effectiveness and practical advantages in structural assembly settings.

Figure 1 in Section 1 illustrates the overall gantry robot system considered in this study. The gantry structure provides overhead motion and defines the robot's operational workspace. A nailing end-effector, integrating both a nail gun and a nail magazine, is mounted on the moving carriage to enable autonomous connection operations. The target floor frame structure to be assembled is on the floor, while the nail storage area, where nail magazines are refilled, is near the boundary of the gantry system. During operation, the robot travels within the gantry workspace to access each connection location and periodically returns to the storage area when the magazine capacity is exhausted.

Figure 2 presents a plan view of the gantry robot workspace and the corresponding floor frame nailing tasks. The dashed boundary indicates the reachable workspace of the gantry system. The floor frame consists of two pairs of horizontal header joists at the top and bottom, connected by 9 vertical cross joists. Nailing demand is intentionally defined as non-uniform across the joints to reflect realistic fastening requirements. Specifically, the four corner joints between the header joists and outermost cross joists, along with the two midspan joints on the top and bottom headers, are designated as strengthened connections requiring twice as many nails as standard joints.

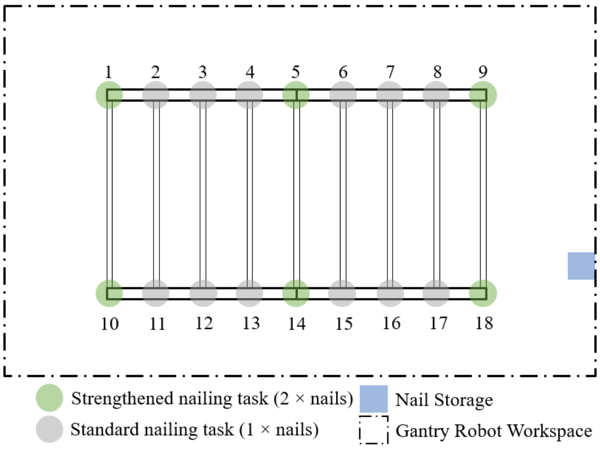

Figure 2. Plan view of the gantry robot workspace and floor-frame assembly

The training instances in this study are generated using a set of predefined rules with a fixed seed. The floor-frame to be assembled is first placed within a 4 m × 2 m rectangular region, with the length of the vertical cross joists sampled uniformly between 1 m and 2 m. The 9 vertical joists are then positioned with center-to-center spacings drawn from a uniform range of 0.3-0.5 m, yielding overall frame dimensions between 2.4 m × 1 m and 4 m × 2 m. The reachable workspace of the gantry system is defined by sampling its plan dimensions between 4 m × 2 m and 8 m × 6 m. The nail storage location is selected along the boundary of this workspace such that it always lies outside the range of the assembled floor frame. For each instance, the nail demand at a standard nailing task is drawn uniformly from 5-11 nails, while the demand at a strengthened nailing task is set to twice this value. The magazine capacity of the nailing end-effector is fixed at 100 nails for all experiments.

4.2 Results

4.2.1 Comparison of DRL with Exact and Heuristic Solvers

With all instances generated using a fixed random seed (seed = 123), 100 problem instances were created under this setting, and the solutions produced by the trained DRL model were compared against those obtained by Gurobi [24], Genetic Algorithm (GA) [25], and Nearest Neighbour (NN) baselines.

Gurobi was used as the optimal benchmark solver. It can obtain globally optimal solutions for small- to medium-scale routing problems within reasonable time, but its computational cost grows rapidly with problem size due to NP-hardness. For large-scale instances (e.g., over 100 connection tasks), solving can take hours or even days, making exact optimization impractical under limited time constraints [24,26].

Figure 3 illustrates the routing solutions obtained by different optimization methods for the first instance generated under seed=123. In this instance, each standard connection task requires 9 nails, while each strengthened connection task requires 18 nails. Overall, both the DRL and GA approaches produce solutions that are close to the optimal routes generated by Gurobi, whereas NN exhibits substantially greater deviation from optimality. Specifically, the primary discrepancy between the DRL result and the Gurobi solution occurs at Connection Task 8: in the Gurobi solution, this task is completed during the return segment of Route 1, thereby reducing the total travel distance.

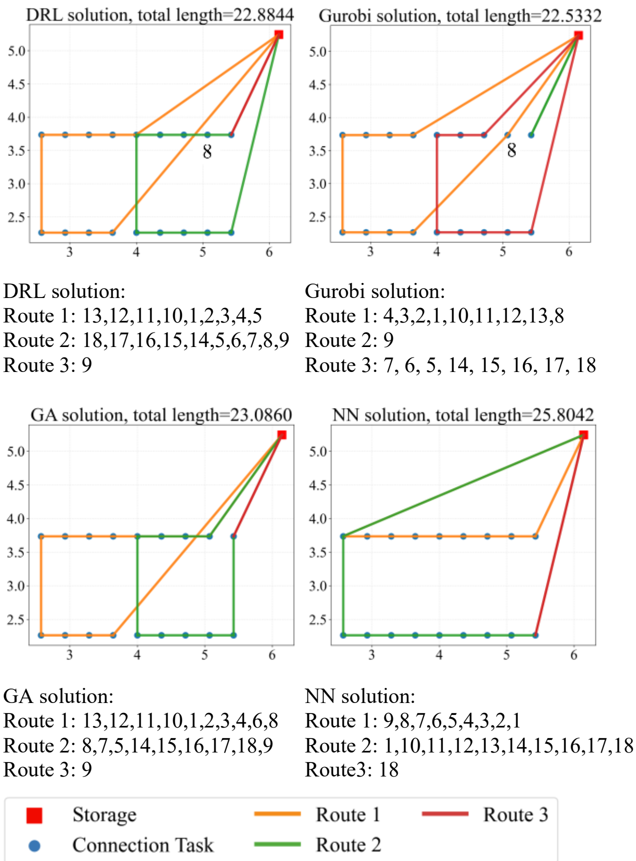

Figure 3. Comparison of routing solutions obtained by DRL, Gurobi (optimal), GA and NN for a representative problem instance (1st instance with seed=123)

To mitigate the effects of randomness and robustly evaluate the effectiveness of the DRL approach, all methods were evaluated across the full set of 100 generated instances. The average total travel distance, percentage gap to the optimal solution, and average computation time are summarized in Table 2. In addition to a deterministic greedy rollout, a stochastic DRL variant based on top-p (nucleus) Monte Carlo sampling was also evaluated, generating 128 candidate routes per instance and selecting the best-performing solution [27].

The results show that both DRL methods (greedy and sampling) achieve solutions within approximately 2.1% of the optimal benchmark, similar to the performance of GA. In contrast, NN exhibits the largest degradation from optimality, with an average gap of 9.6%. In terms of computational efficiency, the DRL inference time for both greedy and sampling remains low (around 0.1s), making it suitable for near-real-time applications. While NN provides the fastest solutions (approximately 0.001s), this advantage is negligible in practical construction planning contexts. GA requires significantly more computation time (over 9s), whereas Gurobi solves each instance in over 1s.

Table 2. Average solution and computation time over 100 generated instances (seed=123)

Solver	Distance (m)	Gap to Best	Time (s)
Gurobi	20.0429	0.00%	1.374
DRL (Greedy)	20.4682	2.12%	0.078
DRL (Sampling)	20.4518	2.04%	0.102
GA	20.4631	2.10%	9.100
NN	21.9632	9.58%	0.001

4.2.2 Generalization Performance under Varying Nail Capacities

The proposed DRL policy is trained on problem instances with a fixed nail capacity of 100. Ideally, such a policy should not only perform well on the training distribution, but also generalize to related problems with different nail capacities. To assess this generalization capability, an out-of-distribution evaluation is conducted on 100 instances generated with seed 123 by varying the nail capacity from 50 to 150 at intervals of 10.

Table 3 reports the average total travel distance obtained by Gurobi and DRL for each capacity level, along with the percentage gap between the DRL solution and the optimal Gurobi solution. When the nail capacity lies between 80 and 150, the DRL policy maintains a reasonably stable performance, with the gap to the optimum remaining below 7%. This indicates that the learned policy can adapt to moderate changes in capacity without substantial degradation in solution quality. However, as the capacity is reduced to 70 and below, the gap increases sharply, reaching 22.2% at capacity 70 and exceeding 40% at capacity 60, with the poorest performance observed at capacity 50 (66.9% gap).

Overall, these results suggest that the DRL policy exhibits satisfactory generalization for capacities within the vicinity of the training setting but deteriorates when applied to substantially different settings characterized by lower capacities. The underlying reasons are further discussed in Section 5.1.

Table 3. Generalization performance of the trained DRL policy under varying nail capacities, compared with Gurobi over 100 generated instances (seed = 123)

Nail Capacity	Gurobi Results (m)	DRL (Greedy) Results (m)	Gap to Best
150	15.6065	16.5800	6.24%
140	16.1545	16.9771	5.09%
130	16.9889	17.9866	5.87%
120	17.2348	18.4647	7.14%
110	18.9855	19.5970	3.22%
100(Trained)	20.0429	20.4682	2.12%
90	21.0676	21.8989	3.95%
80	22.8803	24.2583	6.02%
70	25.1728	30.7637	22.21%
60	27.2099	38.5350	41.62%
50	31.5817	52.7125	66.91%

5 Discussion

5.1 Interpretation of Results

Section 4.2.1 evaluates the trained attention-based DRL policy on randomly generated instances drawn from the same distribution as the training data and compares it against Gurobi, GA, and NN. The DRL policy attains solutions with an average gap of about 2.1% to the optimal benchmark, while requiring around 0.1s per instance. This runtime is compatible with online task sequence planning in structural assembly connection operations, yet still delivers high-quality routes. As problem complexity increases (e.g., to instances with 100 tasks), the constant-time inference property of the DRL policy suggests that sub-second runtimes can still be maintained, while the computational burden of Gurobi and GA is expected to grow substantially. For larger-scale or more constrained instances, Gurobi will experience a sharp increase in computational effort, with solution time extending to hours or longer. Taken together, these comparisons illustrate that the DRL approach occupies a favorable position in the speed-quality trade-off spectrum: sufficiently accurate for near-optimal planning and sufficiently fast for on-site deployment.

Section 4.2.2 examines the generalization of the trained policy to datasets with modified nail capacities. In the application scenario considered, the total nail demand ranges from 120 to 252 nails. When the capacity is between 90 and 120, the gantry robot typically needs two to three tours to complete all connection tasks, which is consistent with the training instances. In this setting, the policy produces routes that remain close to optimal. For capacities between 130 and 150, some instances can be completed in a single tour, but most still require two tours; the policy continues to perform reasonably well when chaining more connection tasks into each tour. However, when the capacity drops to 80 or below, the robot must undertake many more tours (up to five tours at capacity 60 and six at capacity 50), a condition that never occurred during training. As a result, performance degrades noticeably.

To enhance robustness in such low-capacity settings, future training could expand the training distribution so that the policy encounters a wider range of demand-capacity ratios and more multi-tour scenarios. In the present study, the training instances were designed to reflect the practical application setting, where the nail magazine capacity is 100, and the nail demand per connection task ranges from 5 to 11. Although this setting is realistic for the target scenario, it limits the diversity of tour structures seen during training. Future work could therefore broaden the nail demand range during training, for example, from 2 to 20 nails per connection task, so that the model is exposed to more varied routing patterns and may generalize better to low-capacity cases.

5.2 Limitations and Future Research

Despite the encouraging results, this study has several limitations that point to important directions for future research.

First, the experimental scenarios considered in this study involve a relatively small number of connection tasks (18 tasks). Although the current results demonstrate the feasibility and promise of the proposed method, its computational advantage is expected to become more evident in larger and more complex problem instances. Future studies should therefore consider larger-scale cases and compare the proposed method with exact solvers such as Gurobi and heuristic methods in terms of inference time and solution quality under practical time constraints.

Second, the current formulation assumes a two-dimensional workspace without obstacles, and travel cost is approximated using Euclidean distance. The proposed approach therefore focuses on high-level task sequencing and routing, rather than low-level motion planning. In more realistic construction settings, the resulting task sequence could be combined with lower-level path planning methods, such as RRT* or Dijkstra-based algorithms, so that travel cost is estimated from feasible collision-free trajectories. Extending the framework in this direction remains an important future task. Third, the current framework assumes static task settings and does not account for replanning. In practical construction environments, robots may need to adapt their plans in response to unexpected events, such as delays, partial task completion, or human-robot interactions. Incorporating online replanning mechanisms and dynamic task updates constitutes an important extension of this work.

Finally, the present study evaluates the proposed approach at the algorithmic level only. Future work will focus on validating the method in laboratory-based robotic experiments, which is a necessary step towards real-world deployment in construction settings.

6 Conclusion

This study presented an attention-based DRL framework for task sequence optimization in gantry robotic connection tasks. By formulating the nailing operation as an SDVRP, the proposed method enables systematic optimization of both routing and refilling behavior under nail magazine capacity constraints. Benchmark comparisons with Gurobi, GA, and NN show that the trained DRL policy achieves a favorable balance between solution quality and computational efficiency: solutions remain within 2.1% of the optimal benchmark while requiring only 0.1s of inference time per instance. This constant-time inference property underscores the suitability of the DRL approach for online planning in construction robotics scenarios.

Generalization experiments show that the policy performs reliably for nail capacities close to the training setting but deteriorates when applied to substantially different conditions that require many additional tours. These findings suggest that robustness could be enhanced by training on a broader range of nail magazine capacities and by incorporating greater variability in nail demand across tasks.

Several limitations point to directions for future research. The experiments were conducted on relatively small problem instances in obstacle-free environments, so extending the method to larger assemblies is an important next step. The current framework also assumes static task settings and does not support replanning, which is crucial for handling disruptions in real construction operations. In addition, the approach has been evaluated only at the algorithmic level; laboratory-based robotic experiments are needed to assess practical feasibility. Overall, the results highlight the potential of attention-based DRL task sequence optimization to enable more efficient robotic connection operations.

References

[1] T. Bock, The future of construction automation: Technological disruption and the upcoming ubiquity of robotics, Autom Constr 59 (2015) 113–121

[2] S. Gao, R. Jin, W. Lu, Design for manufacture and assembly in construction: a review, Building Research and Information 48 (2020) 538–550.

[3] Y. Bai, Composites for Building Assembly, Springer Tracts in Civil Engineering, 2023.

[4] Y. Wang, Y. Fang, Y. Bai, Task planning for human-robot collaboration in structure assembly, Autom Constr 179 (2025) 106464.

[5] Advanced Construction Robotics Inc., Robotic Rebar Installation Solutions, (2025). https://www.constructionrobots.com/.

[6] H. Chai, L. Zhang, P.F. Yuan, Advanced Timber Construction Platform Multi-Robot System for Timber Structure Design and Prefabrication, in: Architectural Intelligence, Springer Nature Singapore, 2020: pp. 129–144.

[7] W. Wenna, D. Weili, H. Changchun, Z. Heng, H. Feng, Y. Yao, A digital twin for 3D path planning of large-span curved-arm gantry robot, Robot Comput Integr Manuf 76 (2022).

[8] X. Wang, J. Gao, X. Zhou, X. Gu, Path Planning for the Gantry Welding Robot System Based on Improved RRT*, Robot Comput Integr Manuf 85 (2024).

[9] J.G. Everett, A.H. Slocum, Automation and robotics opportunities: construction versus manufacturing, J Constr Eng Manag 120 (1994) 443–452.

[10] Y. Ikeda, T. Harada, Application of the Automated Building Construction System Using the Conventional Construction Method Together, in: Proceedings of the 23rd ISARC, Tokyo, Japan, 2006: pp. 722–727.

[11] M. Saleem, S. Jameel, A. Hassan, Z.H. Khan, Design and Fabrication of a Six DOF Gantry Robot for Welding Applications, in: ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems, 2024: pp. 1940–1946.

[12] J. Muñoz-Morera, F. Alarcon, I. Maza, A. Ollero, Combining a hierarchical task network planner with a constraint satisfaction solver for assembly operations involving routing problems in a multi-robot context, Int J Adv Robot Syst 15 (2018). 10.1177/1729881418782088.

[13] C.-H. Yang, L.-T. Tsai, Y. Chen, S.-C. Kang, Assembly sequence planning method for robotic timber wall prefabrication, Autom Constr 181 (2026) 106669.

[14] H. You, Y. Ye, T. Zhou, Q. Zhu, J. Du, Robot-Enabled Construction Assembly with Automated Sequence Planning Based on ChatGPT: RoboGPT, Buildings 13 (2023). 10.3390/buildings13071772.

[15] Y.-D. Kwon, J. Choo, B. Kim, I. Yoon, Y. Gwon, S. Min, POMO: Policy Optimization with Multiple Optima for Reinforcement Learning, (2020).

[16] Y. Chen, M. Chen, F. Yu, H. Lin, W. Yi, An Improved Ant Colony Algorithm with Deep Reinforcement Learning for the Robust Multiobjective AGV Routing Problem in Assembly Workshops, Applied Sciences (Switzerland) 14 (2024).

[17] W. Kool, H. Van Hoof, M. Welling, Attention, learn to solve routing problems!, 7th International Conference on Learning Representations, ICLR 2019 (2019) 1–25.

[18] R. Zhang, A. Prokhorchuk, J. Dauwels, Deep Reinforcement Learning for Traveling Salesman Problem with Time Windows and Rejections, Proceedings of the International Joint Conference on Neural Networks (2020).

[19] F. Berto, et al, RL4CO: An Extensive Reinforcement Learning for Combinatorial Optimization Benchmark, in: Association for Computing Machinery (ACM), 2025: pp. 5278–5289.

[20] A. Vaswani, et al, Attention Is All You Need, in: NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, US, 2017: pp. 6000–6010.

[21] C. Zhang, Y. Wu, Y. Ma, W. Song, Z. Le, Z. Cao, J. Zhang, A review on learning to solve combinatorial optimisation problems in manufacturing, IET Collaborative Intelligent Manufacturing 5 (2022).

[22] I. Bello, H. Pham, Q. V. Le, M. Norouzi, S. Bengio, Neural combinatorial optimization with reinforcement learning, 5th International Conference on Learning Representations, ICLR 2017 - Workshop Track Proceedings (2017) 1–15.

[23] R.J. Williams, Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning, Mach Learn 8 (1992) 229–256.

[24] Gurobi Optimization LLC, Gurobi Optimizer Reference Manual, (2025). https://www.gurobi.com.

[25] J.H. Holland, Genetic Algorithms, Sci Am 267 (1992) 66–73.

[26] P. Toth, D. Vigo, The Vehicle Routing Problem, Society for Industrial and Applied Mathematics, 2002.

[27] A. Holtzman, J. Buys, L. Du, M. Forbes, Y. Choi, The Curious Case of Neural Text Degeneration, (2020).