Lightweight Probabilistic RL for Web-app Compatible Large-scale OHT Path Optimization

OkHwan Bae and Chung-Pyo Hong*

Division of Computer Engineering, Hoseo University, Republic of Korea
E-mail: foem954@gmail.com; cphong@hoseo.edu
**Corresponding Author*

Received 07 December 2025; Accepted 20 January 2026

Abstract

As modern smart-factory environments increasingly require real-time remote operation and lightweight cloud-based control, routing intelligence for OHT systems must be fully web-app compatible, supporting scalable deployment without reliance on high-end local infrastructure. To address these demands and the limitations of static algorithms in large-scale OHT systems, this study proposes a multi-agent reinforcement learning model based on proximal policy optimization, incorporating a state space that accounts for chain blockage probability. The key metric, "movement success probability," integrates preceding agent states to predictively assess chain-reaction congestion, enabling agents to proactively select stable detours. To enhance scalability in high-density environments, the model stabilizes learning through a lightweight policy initialization approach rather than requiring large-scale training from scratch. Moreover, the proposed decentralized structure minimizes central computational overhead, aligning naturally with web-app deployment and enabling real-time monitoring across distributed environments.

Journal of Web Engineering, Vol. 25_4, 583–598.
doi: 10.13052/jwe1540-9589.2545

In a simulation with 1333 nodes and 100 OHTs, the proposed model achieved an average task completion distance of 166,809 mm, improving efficiency by 4.1% over the rule-based Floyd–Warshall method (173,940 mm). Notably, in worst-case scenarios where the rule-based method surged to 321,753 mm due to congestion, the AI model maintained 176,268 mm, achieving a 45.2% reduction and demonstrating superior operational stability.

Keywords: Path optimization, reinforcement learning, proximal policy optimization, web-app compatible routing.

1 Introduction

Modern smart-factory logistics increasingly demand routing systems that are not only efficient but also web-app compatible, enabling lightweight deployment, cloud-based control, and real-time monitoring without reliance on high-end local infrastructure. Overhead hoist transport (OHT) systems, in particular, have evolved into large-scale multi-agent environments where hundreds of vehicles operate simultaneously as production lines grow in scale and complexity. However, widely used static pathfinding algorithms such as Floyd–Warshall, along with rule-based detour strategies that assign distance penalties for obstacle avoidance, exhibit significant limitations. These methods tend to focus on the myopic optimization of individual agents and fail to respond flexibly to fluid congestion changes or complex interactions across the entire system. In particular, distance-penalty approaches operate only after congestion has already formed, often causing frequent deadlocks in high-density regions or generating excessively long detours that rapidly degrade overall transport efficiency. To overcome these dynamic uncertainties, this study proposes a multi-agent reinforcement learning (MARL) model based on proximal policy optimization (PPO). At the core of the proposed method is the introduction of "movement success probability" as a key state variable, enabling agents to predict the likelihood of chain-reaction blockage beyond their immediate local view. This probabilistic metric links the failure probability of a preceding agent with the occupancy likelihood of its subsequent node, allowing each agent to assess not only current node availability but also future blockage potential to proactively select stable paths. To ensure robust performance in large-scale environments characterized by high-dimensional state spaces, the model adopts a fully decentralized execution structure in which each agent independently makes decisions. This minimizes central computational overhead and naturally supports scalable

operation within web-app compatible architectures. The effectiveness of the proposed method was validated in a high-precision simulation environment consisting of 1333 nodes and 100 OHTs. Experimental results demonstrated a 4.1% improvement in average task completion distance, with the rule-based approach recording 173,940 mm and the proposed model achieving 166,809 mm. More critically, in severe congestion scenarios where the rule-based method's travel distance escalated to 321,753 mm due to inefficient detours, the proposed MARL model maintained a stable 176,268 mm – representing a 45.2% reduction. These findings confirm that the proposed probabilistic MARL framework not only enhances average operational efficiency but also ensures strong system stability even under extreme traffic congestion conditions.

The remainder of this paper is organized as follows. Section 2 reviews related works. Section 3 provides a detailed description of the architecture, state representation, and reward function design of the proposed multi-agent reinforcement learning model. Section 4 analyzes the experimental environment and results. Finally, Section 5 presents the conclusions.

2 Related Work

2.1 Table-based RL Approaches

While path optimization for overhead hoist transport (OHT) systems has traditionally relied on mathematical modeling or heuristic techniques, recent research has actively applied reinforcement learning (RL) to address the limitations posed by complex dynamic environments. Early studies modeled the OHT routing problem as a Markov decision process (MDP) and primarily adopted Q-learning-based approaches [1, 2]. These studies implemented node-based control, defining intersections (nodes) or vehicles as agents to learn directions that minimize overall delivery time at each junction. To address the slow reward propagation of simple 1-step Q-learning (TD(0)), Q(λ)-learning was introduced, incorporating eligibility traces to reflect past action trajectories and enhance convergence. Additionally, the QLBWR (Q-learning based weighted routing) algorithm was proposed to balance exploration and exploitation using a Boltzmann Softmax policy. This approach demonstrated significant throughput improvements over static routing by flexibly exploring less congested paths. However, these table-based methodologies face the fundamental limitation of the curse of dimensionality [3]. In real-world semiconductor fab environments consisting of thousands

of nodes and hundreds of vehicles, defining the entire system as a state space causes the Q-table to grow geometrically, rendering real-time updates and memory storage infeasible. Conversely, over-simplifying states for computational efficiency results in the loss of critical fine-grained control information, creating a dilemma where optimal path generation is compromised [4].

2.2 Deep RL and Multi-agent Approaches

To overcome the scalability issues of table-based methods, research integrating deep neural networks, such as deep Q-networks (DQNs), has emerged [5, 6]. DQNs utilize techniques like experience replay and target networks to decouple data correlations and stabilize learning, proving effective in generating valid paths even in environments with complex obstacles. However, as a value-based method, DQNs suffer from the tendency of Q-value overestimation, leading to instability problems where agents make overly optimistic assessments of specific routes, potentially causing unexpected deadlocks [7]. Furthermore, DQNs are structurally optimized for discrete action spaces, limiting their ability to perform precise continuous speed control or smooth acceleration/deceleration, which are essential for maintaining fluid vehicle flow [8]. Meanwhile, multi-agent algorithms like QMIX [9] have been proposed to enhance cooperation. QMIX attempted global optimization by combining individual value functions into a monotonic function. However, such centralized training with decentralized execution (CTDE) methods face a severe scalability bottleneck [10, 11]. Computational costs skyrocket when the number of agents exceeds 100. Moreover, structures specialized for game environments like StarCraft often fail to sufficiently reflect the physical constraints and dynamic characteristics of real-world logistics systems.

Therefore, this study adopts the policy-based proximal policy optimization (PPO) algorithm to overcome the persistent limitations of existing value-based and centralized approaches, optimizing it for large-scale OHT environments [12]. Our approach is distinguished by three key contributions. First is the integration of neural network-based state approximation and transfer learning [13]. By leveraging neural networks to handle infinite state spaces and transferring policy weights from a small-scale (10 agents) to a large-scale (100 agents) environment, we enable immediate adaptation to map changes or fleet expansion. Second is training stability and continuous control. PPO's clipping mechanism prevents drastic policy updates to avoid learning collapse, while its support for continuous action spaces enables precise speed control of OHTs. Third is the fully decentralized structure and

probabilistic state design. By eliminating central bottlenecks, our decentralized architecture ensures scalability for environments with over 100 agents. Additionally, we introduce a dense reward system based on "movement success probability" rather than simple location data, solving the sparse reward problem and significantly accelerating convergence speed.

3 Proposed Scheme

To address the challenges of dynamic pathfinding in multi-agent systems, this study proposes a multi-agent reinforcement learning (MARL) framework based on proximal policy optimization (PPO). The most pivotal innovation of this architecture lies in the design of the state space, which fundamentally incorporates "movement success probability" to maximize the agents' situational awareness. Unlike conventional methods relying solely on positional data, this probabilistic state metric quantifies the risk of potential chain blockages by linking the status of preceding agents with the occupancy probability of subsequent nodes. This equips agents with the intelligence to proactively determine optimal paths that ensure safety and efficiency even in complex logistics environments.

To successfully apply this sophisticated state awareness to large-scale environments, we adopt a stepwise transfer learning strategy. The model first learns fundamental driving rules and collision avoidance strategies in a simplified environment with 10 agents. Subsequently, the optimized policy weights are transferred to a more challenging environment with 100 agents, thereby securing both training efficiency and convergence stability.

Furthermore, regarding practical system operation, while the training phase focuses on establishing sophisticated cooperative strategies among agents, the inference phase follows a fully decentralized execution structure where each agent makes independent decisions. This approach ensures real-time computational speed without central server bottlenecks and enables unlimited system scalability. Consequently, this study presents an integrated methodology that strictly optimizes state, action, and reward components to control the dynamic uncertainties of large-scale OHT systems.

3.1 State Space Design

As detailed in Table 1, the global state is constructed to provide the central critic with a comprehensive and granular perspective of the entire system. This state vector is composed by concatenating five distinct elements, which

Table 1 Structure of the global state.

Category	Component	Description
Global state	All local states	The complete local state vectors for all N agents, concatenated.
	All agent positions	The normalized current (x, y) coordinates for all N agents, concatenated.
	All agent destinations	The normalized current (x, y) coordinates for all N agents, concatenated.
	Cumulative collision count	The cumulative number of agent collisions accumulated until the completion of the episode.
	Task completion count	The number of agents that have reached their destinations by the completion of the episode.

Table 2 Structure of the local state

Category	Component	Description
Self state	Current coordinates	Normalized current *XY* coordinates
	Neighbor node count	Number of accessible neighbor nodes
Neighboring state	Shortest distance	Remaining node count (Dijkstra)
	Shortest path flag	Shortest path flag (1 or 0)
	Path congestion	Total agent count on Dijkstra path
	Movement success probability	Immediate movement success probability
Destination state	Destination coordinates	Normalized destination *XY* coordinates

are the local states of all OHT agents, the current positions of all agents, the destination positions of all agents, the cumulative collision count, and the number of completed tasks. This exhaustive aggregation enables the critic to analyze the specific probabilistic contexts, such as blockage risks, faced by every individual agent. Simultaneously, the inclusion of collision and task completion counts serves as a direct performance benchmark. This structural design allows the critic to correlate micro-level decisions derived from local states with macro-level outcomes like collisions or delivery successes. Consequently, this guides the agents to identify optimal paths that minimize accidents while maximizing overall system throughput.

As presented in Table 2, the local state is structured to provide specific environmental snapshots that drive the agent's learning process. The self state consists of the agent's normalized current x, y coordinates and the neighbor node count, which is the number of accessible neighbor nodes at the current position; through this, the agent can learn its absolute location within the fab

and the topological scope of available actions at the intersection. The destination state is defined by the normalized x, y coordinates of the target node; through this, the agent can learn the goal-oriented movement vector relative to its current position. The neighboring state aggregates detailed metrics for each accessible node to evaluate candidate paths. Specifically, the shortest distance is the remaining number of nodes to the target based on Dijkstra's algorithm; through this, the agent can learn a heuristic to minimize travel time. The shortest path flag is a binary value of 1 or 0 indicating whether the node lies on the static optimal route; through this, the agent can learn to prioritize globally optimized paths when no obstacles are present. The path congestion is defined as the total number of agents present on the Dijkstra shortest path extending from the specific neighbor node to the destination, thereby approximating the congestion level of that route; through this, the agent can learn to foresee and avoid routes that are macroscopically heavily burdened. Finally, the movement success probability $P_{success}$ represents the most critical contribution of this study, designed to quantify the immediate traversability of a node by accounting for the chain-reaction dynamics of inter-agent interactions. Unlike simple binary occupancy checks, this metric recursively evaluates the blockage risk posed by a preceding agent at a target neighbor node v and its intended subsequent node u. The blockage probability is mathematically defined as shown in Equation (1).

$$\begin{aligned} P_{block}(v) &= P_{block}(u) + (1 - P_{block}(u)) \cdot P_{fail} \\ P_{success(v)} &= 1 - P_{block}(v) \end{aligned} \tag{1}$$

where p_{fail} denotes the intrinsic mechanical failure probability, set to 0.1 in this study. This formulation explicitly captures the risk that even if the predecessor successfully secures the subsequent path there remains a 10% likelihood of failure to move due to mechanical issues. Through this mechanism, the agent learns to interpret low probability values not merely as immediate obstacles but as predictive warning signals of propagating gridlocks, thereby acquiring the sophisticated capability to proactively select alternative routes with higher flow stability to prevent system-wide deadlocks.

3.2 Reward Function Design

As outlined in Table 3, the reward function R is strategically engineered to shape a policy that balances individual navigational efficiency with system-wide stability. This function provides immediate feedback to reinforce

Table 3 Structure of the reward function

Category	Component	Condition	Value
Navigational efficiency & goal achievement	Shortest path inference	Select Dijkstra optimal direction	$+0.4$
		Deviate from optimal path	-0.2
	Terminal goal	Successfully reach destination	$+1.0$
		Fail to reach within time limit	-1.0
Safety & validity constraints	Invalid action inference	Attempt geometrically impossible move	-0.05
	Collision penalty	Collide with another agent	-0.1
Traffic flow optimization	Congestion inference	Select path with low agent density	$+0.1$
		Select path with high agent density	-0.3
Cooperative system stability	Global episode reward	All agents reach destinations	$+1.0$
	Global instability	Total system collisions > 100	-0.5

adherence to topological constraints, expediency, and safety, while penalizing behaviors that degrade global traffic flow. The specific components are defined as follows.

First, to enforce strict adherence to the physical layout of the fab, a penalty for invalid action inference is imposed. If an agent attempts a geometrically impossible move, such as turning onto a non-existent path at a unidirectional or T-junction node, a negative reward is applied, compelling the agent to learn the valid action space corresponding to each node type.

Second, shortest path inference is incentivized to promote rapid delivery. At branching nodes, the agent receives a positive reward if it selects the direction corresponding to the Dijkstra-optimized shortest path. Conversely, deviating from the optimal route results in a penalty, discouraging inefficient detours unless necessary for obstacle avoidance.

Third, congestion inference drives the agent to perform autonomous load balancing. At decision points, the agent evaluates the density of other agents along the projected path to the destination. Selecting a route with lower agent density yields a positive reward, whereas choosing a route with higher density incurs a penalty. This mechanism teaches the agent to sacrifice immediate shortest-path adherence in favor of underutilized routes to prevent bottleneck formation.

Fourth, collision penalties are rigorously applied to prioritize safety. In the event of a physical collision between agents, a penalty is immediately levied on the involved agents to discourage unsafe maneuvers.

Finally, terminal rewards dictate the outcome of tasks and the episode. Upon reaching its specific destination, an agent receives a significant positive reward, whereas failure to complete the task within the time limit results in a negative penalty. Furthermore, a global episode reward is assessed to evaluate collective performance: a system-wide bonus is granted only if all active agents successfully reach their destinations. However, to ensure that high throughput does not come at the cost of stability, a deduction is applied if the cumulative collision count across the system exceeds a predefined threshold during the episode. Ultimately, this multi-faceted reward architecture serves as a critical shaping mechanism that enables the agent to transcend simple rule-following. By continuously optimizing its policy against these feedback signals, the agent learns to strike a sophisticated balance between individual expediency and system-wide stability. Specifically, the agent acquires the capability to dynamically determine whether to strictly adhere to its shortest path for immediate speed or to strategically execute a detour based on real-time critical metrics such as path congestion and movement success probability. This adaptive decision-making process allows the agent to autonomously navigate through complex traffic, prioritizing routes that offer high movement probability and low density over theoretically shorter but risk-prone paths, thereby maintaining optimal operational efficiency.

In summary, the proposed architecture establishes a robust autonomous navigation framework by synergizing the movement success probability, which empowers agents with predictive blockage awareness with a multi-faceted reward function that rigorously balances navigational expediency and safety constraints. To further overcome the inherent challenges of training instability and slow convergence in large scale environments characterized by vast state spaces, we successfully implemented a transfer learning strategy. By directly transferring policy weights pre-trained in a dense, small-scale environment (10 agents) to a complex large-scale system (100 agents), we secured both computational efficiency and immediate policy robustness. Consequently, this holistic integration of probabilistic state modeling, strategic reinforcement, and scalable transfer learning significantly enhances the overall system performance, demonstrating superior routing efficiency and deadlock resilience compared to conventional methods.

4 Evaluation

This section presents an analysis of the testbed environment and evaluation metrics employed to validate the efficacy of the proposed multi-agent

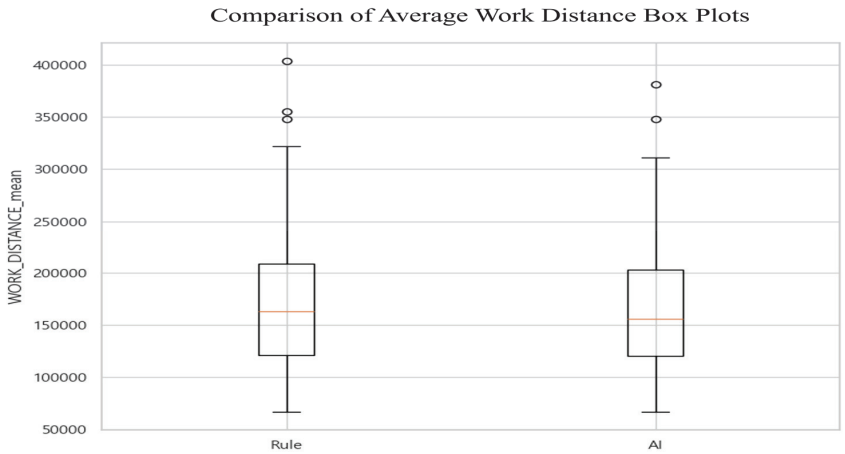

Figure 1 Comparison of average work distance box plots.

reinforcement learning model. Our comparative benchmark was a penalized Floyd–Warshall algorithm engineered for dynamic congestion deterrence by re-calculating routes every second and adding a 20 m distance penalty for each occupying OHT. It should be noted that the scope of the comparative analysis in this study is limited to this specific algorithm, and comparisons with other pathfinding methods remain a subject for future research. The simulation utilized a 1333-node unidirectional graph to represent a typical logistics layout, with the number of OHT agents fixed at 100 to assess stability in high-density operation. The proposed model is implemented using the multi-agent proximal policy optimization (MAPPO) framework. Critically, each OHT job mandates two sequential path searches: first, navigating from the agent's current position to the pickup location (source), and second, proceeding from the source to the final destination. All testing was executed on dedicated computing hardware equipped with an AMD Ryzen 9 7950X (16-Core) CPU, 128 GB of memory, and an NVIDIA RTX 5000 Ada Generation GPU, running on Windows 11 for Workstation. The parameters of PPO are shown in Table 4.

Shown in Figure 1 and Table 5 is the performance evaluation of the proposed multi-agent PPO model versus the rule-based model centered on the analysis of the mean work distance. Quantitatively, the rule-based model recorded an average work distance of 173,940 mm, whereas the AI model recorded 166,809 mm, demonstrating an improved transport efficiency of approximately 4.1%. Furthermore, as illustrated by the box plot, the median

Table 4 PPO hyperparameters

Category	Rule
Learning rate	0.0001
Clipping range	0.15
Discount factor	0.99
GAE parameter	0.95

Table 5 Boxplot summary statistics

Category	Rule	AI
Mean	173,940 mm	166,809 mm
Std	69,100 mm	62,658 mm
Min	66,283 mm	66,283 mm
25%	120,698 mm	120,500 mm
50%	162,598 mm	155,349 mm
75%	208,101 mm	202,856 mm
Max	403, 997 mm	381,058 mm

of the AI model is positioned noticeably lower than that of the rule-based model. This indicates that the AI model consistently selected shorter paths in over 50% of scenarios, suggesting superior average performance. The most decisive difference emerges in the upper distribution and outliers. The rule-based model exhibits maximum outliers extending beyond 400,000 mm, a higher value than the AI model's highest outliers. The presence of these extreme outliers statistically supports the conclusion that the rule-based algorithm, in specific high-density situations, is susceptible to severe congestion or near-deadlock conditions, causing the work completion distance to surge to an uncontrollable level due to inefficient detours. In conclusion, the box plot analysis unequivocally demonstrates that the proposed AI model is significantly superior to the rule-based method, not only in terms of efficiency (lower median) but also stability (maximum outlier suppression), confirming a lower risk of sudden path length volatility in dynamic environments.

A crucial distinction between the two models is highlighted by analyzing a specific high-congestion instance, as visually represented by Figure 2 (rule-based path) and Figure 3 (AI-based path). In this extreme scenario, the rigid, locally reactive nature of the rule-based model resulted in a path length surging to 321,753 mm, demonstrating its failure to handle propagating congestion effectively. Figure 2 clearly shows the inefficient, protracted detours taken by the rule-based agent. In stark contrast, Figure 3, which illustrates the path generated by the proposed AI model, successfully mitigated this

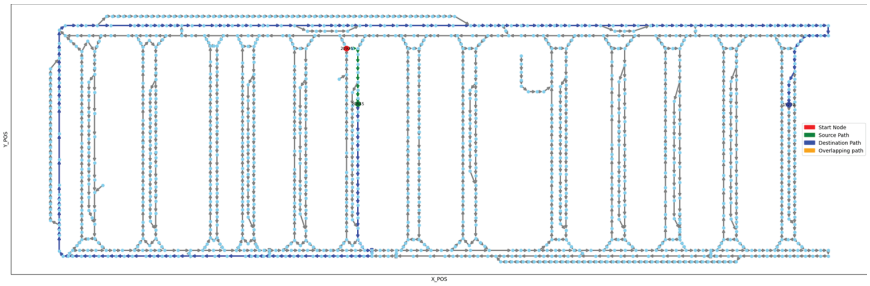

Figure 2 Path trajectory of the rule-based model in the high-congestion scenario.

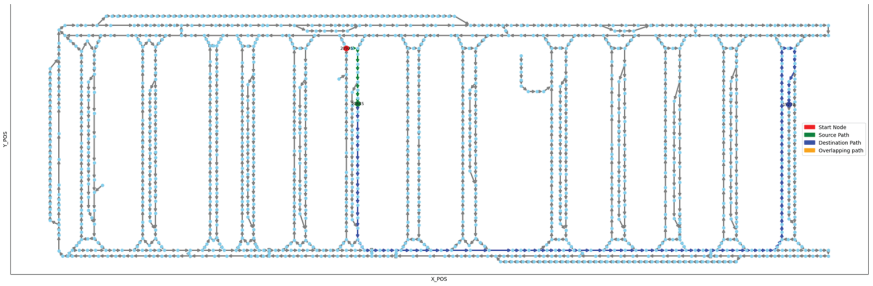

Figure 3 Path trajectory of the proposed AI model in the high-congestion scenario.

severe over-routing, resulting in a significantly shorter and controlled task completion distance of 176,268 mm. This disparity arises because the rule-based algorithm fails to assess the true congestion risk and rigidly follows excessively long detours, whereas the AI model considers predictive metrics like movement success probability and path congestion to achieve a shorter travel distance. This visual and quantitative comparison confirms that the AI model's integrated approach enables it to choose stable, globally aware routes, thereby eliminating the chaotic, excessively long paths characteristic of the baseline algorithm.

Figures 4 and 5 illustrate the node visitation frequency heatmaps for each methodology, clearly revealing the fundamental differences in how the two models manage traffic. The rule-based heatmap in Figure 4 shows a wide distribution of visitation frequency across the entire map, exhibiting a tendency to use lengthy detour routes primarily along the edges and outer perimeter. This pattern confirms the reactive nature of the rule-based algorithm, which rigidly adheres to unreasonably long alternative paths simply to avoid immediate congestion in the center. In contrast, the AI-based heatmap in Figure 5 displays a markedly different pattern: the high-frequency zones

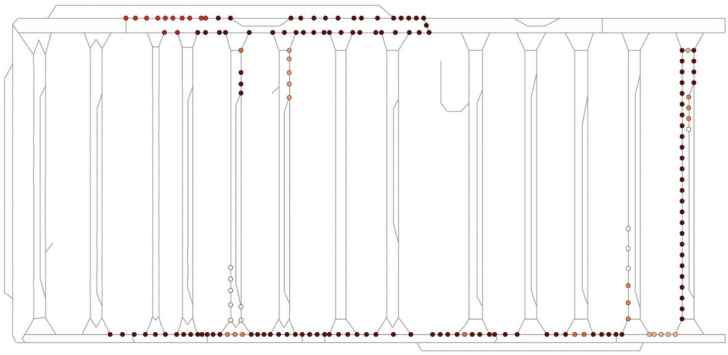

Figure 4 Visitation frequency heatmap of the rule-based model.

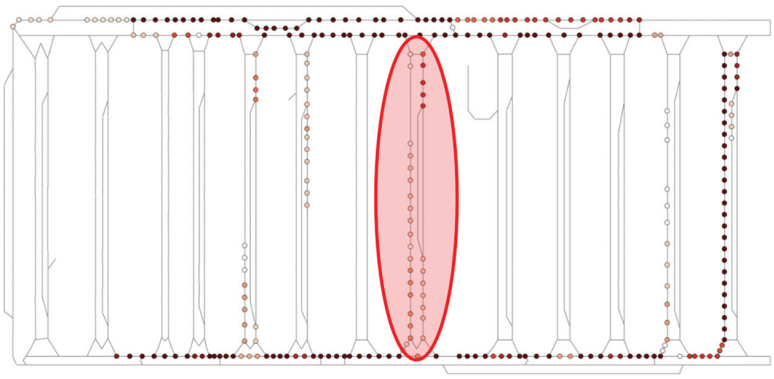

Figure 5 Node visitation frequency heatmap of the proposed AI model.

(visited 60 times or more) are concentrated on the central, critical routes (the "highway" segments) traversing the middle of the map. This efficiency is a direct result of the AI agents moving along optimal paths by comprehensively judging predictive information such as movement success probability and road blockage risks. The most crucial aspect here is the AI model's learned ability to determine the optimal timing for entering this central high-traffic corridor.

5 Conclusion

In this study, we proposed a multi-agent reinforcement learning (MARL) framework based on proximal policy optimization (PPO) to fundamentally overcome the limitations of static pathfinding algorithms in large-scale OHT

systems. The core contribution of this research is the design of a probabilistic state space incorporating movement success probability, which enables proactive congestion prevention by estimating chain-reaction blockage risks in advance. This awareness allows each agent to safely select more reliable routes, thereby preventing system-wide gridlocks and enhancing dynamic routing intelligence in multi-agent logistics operations.

A notable advantage of the proposed framework lies in its strong compatibility with web-app based architectures. The fully decentralized execution design greatly reduces dependence on central computational resources, enabling scalable deployment and real-time control without requiring high-performance local infrastructure. Such lightweight integration is ideal for modern cloud-connected smart factories, where distributed accessibility and operational scalability are essential.

The effectiveness of the proposed model was validated in a simulation with 1333 nodes and 100 OHTs. Experimental results demonstrated an average task completion distance of 166,809 mm, representing a 4.1% improvement over the rule-based Floyd–Warshall method (173,940 mm). Most notably, in severe congestion scenarios where the rule-based approach surged to 321,753 mm due to inefficient detours, the proposed MARL model maintained a stable 176,268 mm – achieving a 45.2% reduction. These results conclusively demonstrate that the proposed probabilistic MARL approach delivers superior routing efficiency, strong operational stability, and clear feasibility for web-enabled remote logistics management in dynamic, large-scale industrial environments.

Acknowledgement

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation(IITP)-Innovative Human Resource Development for Local Intellectualization program grant funded by the Korea government(MSIT)(IITP-2025-RS-2024-00436765)

References

[1] Hwang, Illhoe, and Young Jae Jang. “Q (λ) learning-based dynamic route guidance algorithm for overhead hoist transport systems in semiconductor fabs.” International Journal of Production Research 58.4 (2020): 1199–1221.

[2] Watkins, Christopher JCH, and Peter Dayan. "Q-learning." Machine learning 8.3 (1992): 279–292.
[3] Bellman, Richard. "Dynamic programming and stochastic control processes." Information and control 1.3 (1958): 228–239.
[4] Kober, Jens, J. Andrew Bagnell, and Jan Peters. "Reinforcement learning in robotics: A survey." The International Journal of Robotics Research 32.11 (2013): 1238–1274.
[5] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).
[6] Liao, Haiguang, et al. "A deep reinforcement learning approach for global routing." Journal of Mechanical Design 142.6 (2020): 061701.
[7] Van Hasselt, Hado, Arthur Guez, and David Silver. "Deep reinforcement learning with double q-learning." Proceedings of the AAAI conference on artificial intelligence. Vol. 30. No. 1. 2016.
[8] Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971 (2015).
[9] Shen, Zi-Zhen, Rui Yu, and Yang-Yang Chen. "Rules-PPO-QMIX: multi-agent reinforcement learning with mixed rules for large scene tasks." 2021 China Automation Congress (CAC). IEEE, 2021.
[10] Lowe, Ryan, et al. "Multi-agent actor-critic for mixed cooperative-competitive environments." Advances in neural information processing systems 30 (2017).
[11] Yang, Yaodong, et al. "Mean field multi-agent reinforcement learning." International conference on machine learning. PMLR, 2018.
[12] Schulman, John, et al. "Proximal policy optimization algorithms." arXiv preprint arXiv:1707.06347 (2017).
[13] Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." IEEE Transactions on knowledge and data engineering 22.10 (2009): 1345–1359.

Biographies

OkHwan Bae received his master's degree in computer engineering from Hoseo University in 2025. His research interests include computer vision, deep learning, and reinforcement learning.

Chung-Pyo Hong received his B.Sc. and M.Sc. degrees in computer science from Yonsei University, Seoul, Korea, in 2004 and 2006, respectively. In 2012, he received his Ph.D. degree in computer science from Yonsei University, Seoul, Korea. He is currently an associate professor of computer engineering at Hoseo University, Asan, Korea. His research interests include machine learning, explainable AI, and data science.