Reinforcement Learning for Multi-Agent Planning in the League of Robot Runners

Anonymous submission

Abstract

Integrating planning and learning remains a central challenge in developing adaptive multi-agent systems. This work investigates how reinforcement learning (RL) can enhance multi-agent planning through experiments in the League of Robot Runners (LoRR), a benchmark traditionally rooted in classical planning. We reformulate LoRR in a multi-agent RL environment and adapt the Monte Carlo Tree Search (MCTS) algorithm for use in these settings. Because standard MCTS assumes a single agent and does not reuse experience, our approach introduces limited information sharing across episodes to improve coordination and efficiency. Preliminary results show that while the adapted method does not yet match the performance of classical planners, it provides a foundation for exploring how learning-based techniques can enhance coordination and adaptability in structured planning environments.

Introduction

Learning and automated planning represent two complementary yet traditionally distinct paradigms in artificial intelligence, differing in both objectives and underlying assumptions. Classical planning aims to synthesize action sequences that achieve the desired goals at execution time. It excels at explicit model-based reasoning, long-horizon deliberation, and online adaptability but presumes access to structured and accurate domain models. In contrast, learning-based approaches, particularly those based on RL, seek to derive policies that maximize expected returns under uncertainty and partial observability. Such methods can generalize from experience but typically employ implicit representations, which limit their ability to adapt rapidly to novel conditions, or plan over extended temporal scales.

These methodological discrepancies have led to distinct representations and algorithmic frameworks, often developed in isolation from one another (Silver et al. 2024; Lake et al. 2017). The gap becomes even more pronounced in multi-agent domains, where agents must reason not only about their own states and objectives, but also about those of others, and coordinate their behavior under decentralized information and control. Bridging this divide calls for neurosymbolic or hybrid approaches that integrate the structured reasoning capabilities of planning with the robustness and flexibility of learning, enabling agents to operate effectively in dynamic, uncertain, and socially interactive environments (Anthony, Tian, and Barber 2017).

This work seeks to bridge the gap between learning-based and planning-based approaches by applying RL techniques to a benchmark grounded in automated multi-agent planning research: the League of Robot Runners (LoRR) (LoRR Organizers 2024). LoRR is a multi-agent navigation environment in which multiple robots must complete a continuous stream of spatially distributed tasks while avoiding collisions with obstacles and one another. This setting captures key aspects of real-world multi-robot coordination problems, including those arising in warehouse automation, fleet management, and agricultural robotics.

Although LoRR presents challenges that align naturally with learning-based methods, submissions to the 2024 competition relied exclusively on classical planning algorithms (Jiang et al. 2024). This observation motivates the exploration of hybrid approaches that integrate learning with planning: leveraging learned policies for efficiency while maintaining the explicit safety and coordination guarantees characteristic of model-based methods.

In this paper, we present an initial effort to adapt LoRR for RL and demonstrate a hybrid algorithm that combines Monte Carlo Tree Search (MCTS) with online policy distillation. Our contributions are threefold: (1) a Gym-compatible implementation of the LoRR environment for multi-agent RL experimentation; (2) a compound MCTS-based algorithm that distills policies online from visit distributions during execution, enabling incremental improvement in navigation efficiency; and (3) a preliminary empirical analysis showing that the learned policy preserves deterministic collision-avoidance guarantees while improving adaptability to novel task distributions. Unlike purely reactive RL agents, our approach retains the capacity for downstream planning and search, supporting robust deployment in dynamic, real-world environments.

Background and Motivation

The League of Robot Runners (LoRR) is a planning competition benchmark in which agents navigate a discrete grid world to perform a continuous stream of delivery-like tasks. Agents must plan routes that minimize travel time and avoid collisions with static obstacles and other moving agents. The competition's rules assume full knowledge of the environ-

ment and rely on centralized or decentralized planning algorithms that compute feasible paths before execution. Multi-agent pathfinding has been extensively studied in this context, highlighting both the effectiveness of classical planners and the growing interest in learning-based and hybrid approaches (Wang et al. 2025). While classical planning approaches can produce near-optimal paths, they typically require recomputation from scratch whenever the environment changes (Li et al. 2021). This limits scalability in dynamic or partially observable settings, where repeated planning can become computationally expensive.

Sartoretti et al. (Sartoretti et al. 2019) introduced a hybrid learning approach that combines RL and imitation of an expert multi-agent path finding planner to train decentralized policies that scale to hundreds of agents in cluttered grid worlds. A recent algorithm, WPPL, uses imitation learning to scale to 10,000 agents and matches or exceeds search-based methods from LoRR (Jiang et al. 2025). It shows how learned policies can complement planning/search methods in large-scale multi-agent path-finding, providing encouraging evidence that LoRR can be solved using learning-based augmentations.

Our work continues this thread of neurosymbolic research with the aim of bridging the gap between offline learning and online planning. We explore a hybrid approach that integrates Monte Carlo Tree Search (MCTS) (Browne et al. 2012) with RL. MCTS provides efficient planning, while RL enables agents to learn reusable policies. Our motivation is that learned policies can capture structural regularities of the LoRR environment, potentially allowing agents to generalize to unseen maps or task configurations without full replanning. Unlike standard RL agents, which only react based on their learned policy, combining learning with planning allows the agent to still perform short-term search and adjust to new situations. We hypothesize that this integration could yield agents that adapt more flexibly and transfer more effectively across environments in the LoRR benchmark.

Approach and Contributions

We propose a practical combination of planning and RL for multi-agent path planning that keeps the safety and structure of a search-based planner while learning reusable policies that can guide planning across different environments and task configurations. The system decomposes the multi-agent problem into per-agent planning under time-indexed occupancy constraints, derived from the already-planned trajectories of other agents. Each agent's state is its grid coordinate $s = (x, y)$. Time acts as an external variable that limits valid moves through standard vertex and edge collision checks. Any action suggested by learning or search must still pass the same collision filter.

To coordinate multiple agents planning simultaneously, we employ a fair scheduling mechanism that alternates planning responsibility based on how much each agent has already committed. Specifically, at each iteration, the agent with the fewest planned timesteps is selected to plan its next task segment. This ensures balanced progress across agents and prevents any single agent from consuming excessive planning resources early in the process. When an agent completes planning for one subgoal, it immediately proceeds to the next location in its multi-location task; if all locations in its current task are complete, a new task is assigned from the shared input list. This mechanism ensures agents visit all errands within each task in the prescribed order.

At each decision step, an agent runs Monte Carlo Tree Search (MCTS) from its current position toward its next subgoal. The search respects time-indexed occupancy constraints: at timestep t, the agent may not move to a position that will be occupied by another agent at timestep $t + 1$ (vertex collision), nor may it swap positions with another agent moving in the opposite direction (edge collision). Formally, let $\mathcal{A} = \{1, \ldots, N\}$ denote the set of agents and $\mathcal{T} = \{0, \ldots, T\}$ the discrete time steps. We define a time-indexed allocation tensor $X \in \{0, 1\}^{|\mathcal{A}| \times |\mathcal{T}| \times |G|}$, where $X_{i,t,g} = 1$ indicates that agent i occupies grid cell g at time t. Feasibility requires that no two agents occupy the same grid cell at the same time and that edge swaps between consecutive time steps are forbidden:

$$\sum_i X_{i,t,g} \leq 1, \quad X_{i,t,g_1} X_{j,t+1,g_2} = 0 \;\; if \;\; g_1 \leftrightarrow g_2.$$

During planning, agents update X incrementally, and the scheduling mechanism selects the agent with the smallest planned horizon τ_i to extend its path. This produces a 3D occupancy grid (x, y, t) that captures both spatial and temporal coordination constraints.

Figure 1: Illustration of the time-indexed reservation process. Agent A plans first and reserves its future positions. When Agent B plans, the reserved cells marked with 1 are treated as occupied at their respective timesteps, and Agent B must avoid them when generating its own trajectory.

These constraints are enforced during both tree expansion and action validation, ensuring all generated paths remain collision-free. Let $N(a)$ be the number of times action a was visited at the root node during MCTS simulations. These visit counts are normalized into a policy distribution:

$$\pi_{MCTS}(a \mid s) = \frac{N(a)}{\sum_b N(b)},$$

where actions that led to more promising paths receive higher exploration counts and thus higher probability. This

policy reflects the actions that are most successful given the current agent's position, goal position, and surrounding constraints.

When using the hybrid configuration, we compute a lightweight observation $o(s)$ as a three-channel grid (walls, agent position, goal position) and evaluate a small policy–value network $(\pi_\theta(\cdot \mid o), V_\theta(o))$. During action selection, the search policy and learned policy are blended:

$$\pi_{blend}(a \mid o, s) = (1 - \beta)\, \pi_{MCTS}(a \mid s) + \beta\, \pi_\theta(a \mid o),$$

and the chosen action is

$$a^* = \arg \max_{a \in A_{valid}(s,t)} \pi_{blend}(a \mid o, s).$$

A small β (e.g., 0.15) keeps the system mostly search-driven, with the learned prior providing a modest bias toward actions it has learned to prefer. As the network improves through training, this small contribution becomes more effective at guiding the search.

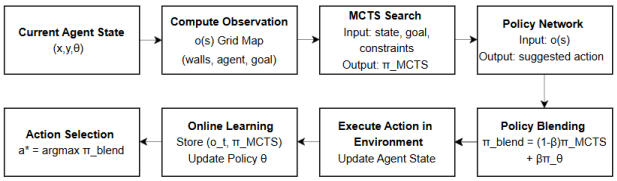

Figure 2: Overview of the hybrid MCTS + RL system for multi-agent planning in the League of Robot Runners. Each agent's state holds its x and y coordinates and its rotation θ. The agent observes its environment to obtain an observation $o(s)$, runs Monte Carlo Tree Search (MCTS) using the current state, goal, and multi-agent constraints, and combines the resulting search policy π_{MCTS} with the learned policy network π_θ through a blending step. The selected action is executed in the environment with rotation penalties. Observations and MCTS visit counts are stored for online policy distillation.

The system also accounts for rotation penalties that arise when agents change direction. Each agent maintains an orientation (North, East, South, or West), and turns incur time penalties: a 90° turn adds one timestep, while a 180° turn adds two. These penalties are tracked globally and included in the final time computation. This cost model encourages the planner and learned policy to prefer forward motion and minimize unnecessary rotations or waiting.

Learning happens online during normal execution. For each task segment (the sequence of decisions to reach a current goal), the system stores pairs $(o_t, \pi_{MCTS,t})$ at the root of each search. When a segment ends, a binary result $z \in \{0, 1\}$ indicates whether the goal was reached. The network parameters θ are updated by minimizing:

$$L(\theta) = CE(\pi_{MCTS}, \pi_\theta) + \lambda(V_\theta(o) - z)^2 - \alpha H(\pi_\theta),$$

where CE is cross-entropy and H is entropy. A few stochastic gradient steps are applied after each segment, and the parameters are saved so later runs start from improved policies.

The policy distillation step converts the local search behavior of MCTS, which is already aware of moving agents and obstacles, into a learned prior that can guide future planning. The learned policy suggests likely good actions, and the value estimate helps the search allocate its effort more efficiently. All actions remain subject to the planner's collision checks, so the system maintains the same safety guarantees as the base method. By planning independently for each agent and representing others through a time-indexed occupancy grid, the approach avoids the full joint-state complexity while still responding to multi-agent interactions.

Experiments and Preliminary Results

We conducted preliminary experiments using small grid environments adapted from the League of Robot Runners. Each scenario consisted of agents navigating to assigned goals while avoiding other agents. We evaluated our method on two configurations:

1. **Pure MCTS baseline**: standard search-based planner without learning.
2. **Hybrid MCTS + RL**: our proposed approach with a small learned prior ($\beta = 0.15$).

We focused on small-scale scenarios where both methods could reliably complete all tasks within reasonable computational time. This enables us to assess the potential benefits of learning while avoiding the computational overhead and potential non-terminating behavior that can arise in larger-scale scenarios.

Experimental Setup

We evaluated both methods on a 10×10 grid world with 4 agents, averaged over 10 independent runs. Each run consisted of a continuous stream of tasks assigned to agents, with the planner computing paths online. The pure MCTS baseline uses standard UCB exploration without any learned guidance. The hybrid MCTS+RL method starts with a randomly initialized policy network and learns online during execution, distilling the MCTS search policy after each completed task segment. Both methods use identical collision-avoidance constraints and scheduling mechanisms to ensure fair comparison.

Results

Table 1 summarizes the average performance across 10 runs. Both methods achieved 100% task completion with zero collisions, demonstrating that the hybrid approach maintains the safety and reliability guarantees of pure MCTS while introducing learning capabilities. The hybrid planner achieves comparable path efficiency (21.44 steps vs. 22.69 steps), showing that online policy learning does not degrade performance even when starting from a randomly initialized network.

This result establishes a crucial baseline: the hybrid system successfully integrates learning without compromising the planner's effectiveness. While the immediate performance gains are modest, the framework introduces capabilities that pure MCTS alone cannot provide. The learned policy network, developed through online distillation of MCTS

Table 1: Preliminary performance comparison on a 10×10 grid with 4 agents (averaged over 10 runs). Both methods achieved 100% success with zero collisions, demonstrating that the hybrid approach maintains performance while enabling learning.

Agents	Method	Avg. Steps
4	MCTS	22.69
4	MCTS+RL	21.44

search behavior, potentially captures spatial navigation patterns that could be reused across different task configurations or map variations, which might reduce computational overhead in future deployments. Additionally, by serving as a compressed representation of effective navigation strategies extracted from MCTS search distributions, the policy network could facilitate transfer learning or provide warm-start initialization for new environments. These preliminary results suggest that incremental policy improvement through online distillation is feasible, offering a foundation for exploring more sophisticated learning strategies in future work.

We also tested a higher-density configuration with 15 agents on a 10×10 grid. The system remains collision-free and the agents ultimately complete their assignments, but paths are suboptimal, with unnecessary local loops and occasional detours before converging to the goal. These results highlight emerging scalability limitations under higher agent densities, motivating ongoing efforts to enhance coordination efficiency in densely populated environments.

Discussion and Future Work

The preliminary results demonstrate that integrating learning into a search-based planner achieves comparable efficiency while maintaining the safety and structure that make classical methods reliable. While immediate performance gains are modest, the hybrid approach successfully introduces learning capabilities without degrading baseline performance. Even with limited training, the learned policy network begins to capture spatial navigation patterns that could be reused across different task configurations or map variations.

One of the central challenges ahead is understanding how the learned components generalize beyond the training environments. Because LoRR involves repeated tasks in a shared space, policies may capture environment-specific regularities that do not immediately transfer to new layouts or agent densities. Systematically evaluating this transfer will be an important next step to validate the reusability claims of our framework. A critical direction for future work is scaling the approach to handle more agents or more crowded spaces, where coordination complexity increases significantly.

Another open question concerns the balance between planning depth and learned guidance. Our current experiments use a fixed blending parameter ($\beta = 0.15$) that provides modest influence from the learned policy. Exploring adaptive mechanisms for adjusting this balance based on policy confidence or search efficiency could help amplify the benefits of the hybrid approach in future work.

The primary value of this early work lies in creating an extensible framework that enables future capabilities, such as policy reuse, transfer learning, and adaptive behavior. The learned policy network, even when providing modest immediate benefit, creates a reusable asset that can be refined through additional training, applied to new environments, or used to warm-start planning in novel scenarios.

This hybrid approach points toward a middle ground between pure search and pure learning. It preserves the interpretability and guarantees of planning while gaining the adaptability of RL. Continued experiments and ablations will help clarify how these two processes can best support each other in dynamic, multi-agent environments, particularly as we scale to more agents and more crowded spaces where coordination complexity increases significantly.

References

Anthony, T.; Tian, Z.; and Barber, D. 2017. Thinking fast and slow with deep learning and tree search. In *Advances in Neural Information Processing Systems (NeurIPS)*, 5366–5376.

Browne, C. B.; Powley, E.; Whitehouse, D.; Lucas, S. M.; Cowling, P. I.; Rohlfshagen, P.; Tavener, S.; Perez, D.; Samothrakis, S.; and Colton, S. 2012. A Survey of Monte Carlo Tree Search Methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1): 1–43.

Jiang, H.; Wang, Y.; Veerapaneni, R.; Duhan, T.; Sartoretti, G.; and Li, J. 2025. Deploying Ten Thousand Robots: Scalable Imitation Learning for Lifelong Multi-Agent Path Finding. arXiv:2410.21415.

Jiang, H.; Zhang, Y.; Veerapaneni, R.; and Li, J. 2024. Scaling lifelong multi-agent path finding to more realistic settings: Research challenges and opportunities. In *Proceedings of the International Symposium on Combinatorial Search*, volume 17, 234–242.

Lake, B. M.; Ullman, T. D.; Tenenbaum, J. B.; and Gershman, S. J. 2017. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40: e253.

Li, J.; Tinka, A.; Kiesel, S.; Durham, J. W.; Kumar, T. K. S.; and Koenig, S. 2021. Lifelong Multi-Agent Path Finding in Large-Scale Warehouses. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI '21)*.

LoRR Organizers. 2024. League of Robot Runners (LoRR). https://lorr-competition.org. Accessed: 2025-11-02.

Sartoretti, G.; Kerr, J.; Shi, Y.; Wagner, G.; Kumar, T. S.; Koenig, S.; and Choset, H. 2019. Primal: Pathfinding via reinforcement and imitation multi-agent learning. *IEEE Robotics and Automation Letters*, 4(3): 2378–2385.

Silver, T.; Dan, S.; Srinivas, K.; Tenenbaum, J. B.; Kaelbling, L.; and Katz, M. 2024. Generalized planning in pddl domains with pretrained large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, 20256–20264.

Wang, S.; Xu, H.; Zhang, Y.; Lin, J.; Lu, C.; Wang, X.; and Li, W. 2025. Where Paths Collide: A Comprehensive Sur-

vey of Classic and Learning-Based Multi-Agent Pathfinding. *arXiv preprint arXiv:2505.19219*.