TRAPS Extended Team Description for RoboCup 2026

Yusei Naito[1], Ryoma Mitsuoka[1,2], and Kai Inagaki[1]

[1] TRAPS `traps.robocup@gmail.com`
`https://traps-official.web.app/`
[2] Nagoya University, Furo-cho, Chikusa-ku, Nagoya, Aichi, 464-8603, Japan

Abstract. This paper presents the research and development efforts of team TRAPS in the RoboCup Small Size League (SSL). Traditionally, team soccer strategies in the SSL have relied on hierarchical architectures based on heuristics. However, these systems often lack the adaptability required to handle the combinatorial complexity inherent in multi-agent interactions. To address this, we introduce an adversarial multi-agent reinforcement learning (MARL) approach designed to acquire cooperative policies. Building upon prior research, our method incorporates environment settings, training curricula, and reward designs specifically tailored for the SSL. Experimental results demonstrate that cooperative behaviors emerge between the robots, and we discuss the practical challenges encountered when deploying the learned models in actual matches.

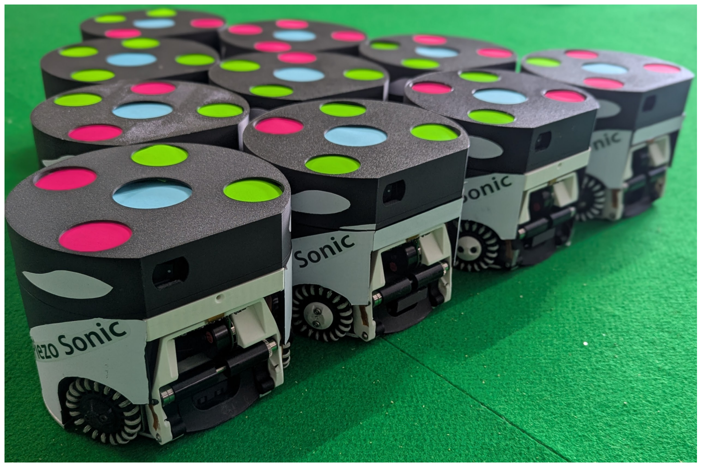


Fig. 1. TRAPS SSL robots. Each robot features a four-wheel omnidirectional drive system, a solenoid-driven kicker, and an active dribbling bar.

1 Introduction

TRAPS is a RoboCup Small Size League (SSL) team established in 2024, primarily composed of former members of team KIKS. The robots shown in Fig. 1 are equipped with omnidirectional movement, straight kick, chip kick, and dribbling capabilities, and have been competing in the league since RoboCup Japan Open 2025. In this paper, we introduce our team's efforts in multi-robot reinforcement learning for the SSL.

The problem of decision-making and control in the SSL has long been dominated by hierarchical rule-based methods, typified by the Skills, Tactics, and Plays (STP) architecture. While these methods can faithfully execute human-designed strategies, they often lack the adaptability to handle unknown opponents or dynamic environmental changes. Furthermore, designing complex co-operative behaviors requires extensive domain expertise and significant manual tuning. To overcome these limitations, the application of Deep Reinforcement Learning (DRL) and Multi-Agent Reinforcement Learning (MARL) has gained traction in recent years.

Early learning-based approaches primarily focused on acquiring specific skills for individual robots. For instance, ZJUNlict employed Deep Deterministic Policy Gradient to learn a "Self-Pass" skill, enabling high-speed maneuvers while maintaining ball possessiona task challenging for traditional path planners. Similarly, CMDragons adopted a hybrid approach by integrating learned primitives, such as kicking and positioning, into the existing STP framework. Although these methods succeed in optimizing specific sub-tasks, they have not yet achieved the learning of overall team strategies or complex coordination. The high-level decision-making layers still rely on rule-based logic, leaving human design limitations as a bottleneck for smooth transitions between skills and global tactical judgment.

TRAPS has implemented MARL based on the Vectorized Multi-Agent Simulator (VMAS) [1] [2], conducting large-scale training with thousands of parallel environments for ball-possession tasks. While this 2D environment is computationally efficient and suitable for massive parallelization, it fails to sufficiently represent physical details crucial to the SSL, such as ball spin, 3D trajectories, and the mechanical contact dynamics of the robots. Consequently, there is a significant risk of a "Reality Gap," where sophisticated ball-handling skills learned in simulation cannot be replicated on physical robots.

The objective of this research is to acquire advanced soccer policies in a form that is practically applicable within the SSL. We have constructed a MARL environment specifically for the SSL, building directly upon the MARLadona framework [3], which has demonstrated its effectiveness in simplified soccer simulation environments.

The primary contributions of this paper are as follows:

- A MARL environment for the SSL based on Isaac Lab [4].
- A specialized training curriculum designed for the SSL.

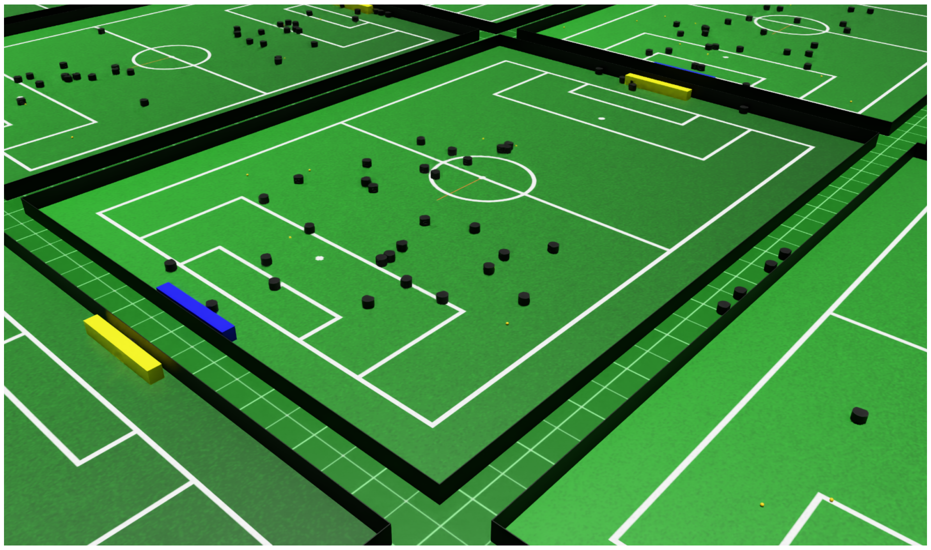


Fig. 2. Multi-Agent Reinforcement Learning Environment for SSL. Our environment models the actual robots, ball, and field dynamics.

2 Methodology

2.1 Problem Formulation

Although the standard SSL architecture allows for centralized control via a global vision system and a single workstation, modeling the joint action space of an entire team as a single agent leads to a combinatorial explosion, making convergence intractable.

Therefore, we logically decompose the problem and model it as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) defined by the tuple $\mathcal{G} = \langle \mathcal{I}, \mathcal{S}, \mathcal{A}, \Omega, \mathcal{P}, \mathcal{R}, \gamma \rangle$ [5]. We simulate decentralization by partitioning the global vision data into ego-centric local observations for each robot.

Here, $\mathcal{I} = \{1, \ldots, N\}$ denotes the finite set of robots. $\mathcal{S}$ represents the global state space, containing the positions and velocities of the ball and all robots on the field. $\mathcal{A}$ is the joint action space. At each timestep, every robot i executes an action $a_i \in \mathcal{A}_i$, which consists of target velocity commands (v_x, v_y, ω) and a kick command. Ω is the joint observation space. Each robot receives a partial observation $o_i \in \Omega_i$ via the observation function $O(s, i)$, representing the environment from its ego-centric perspective. $\mathcal{P}(s'|s, a)$ denotes the state transition probability, governed by the physics dynamics. $\mathcal{R}(s, a)$ is the global reward function shared by all teammates, primarily determined by the match outcome. $\gamma \in [0, 1)$ is the discount factor.

2.2 Robot and Field Configuration

The simulation environment is meticulously configured to adhere to the RoboCup SSL Division B regulations.

Regarding the robot model, we utilize high-fidelity 3D assets exported directly from the TRAPS hardware CAD data into the Universal Scene Description (USD) format. This guarantees accurate collision geometry for critical components such as the chassis shape and dribbling bar placement.

For locomotion control, we abstract away the low-level wheel dynamics to maximize computational efficiency and training stability. Instead of simulating individual omni-wheel rotations and tire-ground friction, we adopt a simplified dynamics model that applies forces and torques directly to the base link.

Specifically, the system calculates the necessary force and torque using a PD controller based on the target velocity vector output by the RL policy, and applies these directly to the physics engine. This abstraction significantly reduces the computational load while sufficiently reproducing the holonomic movement characteristics inherent to omni-directional robots.

2.3 Observations, Actions and Rewards

The overview of the observations is presented in Tab. 1. To ensure generalization across symmetric field positions, all observations are transformed into the robot's ego-centric frame. The local observation vector includes the robot's linear and angular velocities, the relative position and velocity of the ball, and static field information (field size and goal size). Additionally, the robot observes the relative states of the nearest teammates and opponents.

The overview of the actions is presented in Tab. 2. The robot is configured to model the custom TRAPS SSL robot platform, illustrated in Fig. 1. Key kinematic constraints and actuation parametersincluding maximum linear velocity, kick speed limits, and the geometric capture range of the dribblerare rigorously defined to match the physical hardware specifications.

The overview of the rewards is presented in Tab. 3. "Shared" denotes the common reward distributed among all team members. To accelerate initial exploration, we introduce dense rewards, such as rewarding velocity towards the ball or the opponent's goal. To adapt for the SSL, we introduced a dribble penalty term and adjusted the scaling. A dribble penalty term serves a critical role in Rule Compliance, specifically to prevent the "Excessive Dribbling" foul. We formulate this constraint as the following reward term:

$$r^i_{dribble} = -\lambda_{dribble} \cdot |\mathbf{v}_i| \cdot \mathbb{I}(i \in \mathcal{H}_{ball}) \tag{1}$$

where $\mathbf{v}_i$ denotes the linear velocity vector of agent i, and $\lambda_{dribble}$ is a scaling coefficient. The indicator function $\mathbb{I}(\cdot)$ evaluates to 1 if agent i is in the set of ball possessors $\mathcal{H}_{ball}$ (i.e., currently controlling the ball), and 0 otherwise.

This penalty term incentivizes agents to either opt for passing or limit themselves to short-distance carries, rather than dribbling over long distances.

Table 1. Observation Overview

Observation name	Noise scale for actor
Local Observation	
Robot pose	0.002
Robot velocity	0.005
Ball position	0.002
Ball velocity	0.005
Field info	0
Number of robots	0
Global Observation	
Teammate poses	0.002
Opponent poses	0.002

Table 2. Actions Overview

Action name	Range
Target velocity (m/s)	[-0.2, 0.2]
Target angular velocity (rad/s)	[-6, 6]
Kick speed (m/s)	[0, 6]

2.4 Network Architecture: Global Entity Encoder

In the SSL, the number of robots on the field changes dynamically due to penalties or hardware failures. Consequently, traditional Multi-Layer Perceptrons with fixed input dimensions are unsuitable for this task. To address this, we adopt a Global Entity Encoder (GEE) architecture as used in MARLadona which is invariant to both the order and the number of inputs.

The GEE processes these observations by concatenating the ego state with neighboring robot states. By applying a shared encoder followed by a Max Pooling operation, the network extracts a fixed-length global feature vector. This design allows a single policy network to scale seamlessly from 1v1 training scenarios to full-scale matches without retraining.

2.5 Training with Multi-Stage Curricula

Learning effective strategies in a complex continuous domain like SSL is notoriously difficult with sparse rewards. Similar to MARLadona, to facilitate the emergence of cooperative behaviors, we implement a rigorous training pipeline combining Self-Play with an Auto-Curriculum.

To ensure consistent adversarial pressure, agents compete against a dynamic pool of their past policies. Furthermore, to bootstrap the learning process, we utilize auxiliary dense rewards (e.g., velocity towards the ball) solely during the

Table 3. Reward Overview

Reward name	Scale	Shared
Sparse Rewards		
Score	100	True
Ball outside field	1	True
Collision	1	
Dense Rewards		
Ball to goal velocity	0.2	True
Robot to ball velocity	0.5	
Robot to ball direction	0.05	
Dribbling penalty	0.3	

initial phase, and permanently remove them once the agent's win rate reaches a predefined threshold.

Contrary to the original MARLadona approach, we fix the field dimensions to the official SSL regulations throughout the entire training process. We observed that varying the field size hinders the agents from learning consistent spatial featuressuch as the exact distance to the goal or the boundaries of the penalty areathereby degrading performance on physical robots.

We employ the Multi-Agent Proximal Policy Optimization (MAPPO) algorithm [6], which adapts PPO [7] to the Centralized Training with Decentralized Execution paradigm for policy optimization. Both the actor and critic networks are implemented as Multi-Layer Perceptrons (MLPs) with three hidden layers of 128 units each, using ELU as the activation function.

Training uses an adaptive learning rate scheduler initialized at 1.0×10^{-3}. Key hyperparameters are configured as follows: discount factor $\gamma = 0.99$, GAE parameter $\lambda = 0.95$, and entropy coefficient 0.003. The PPO clipping parameter is set to 0.2 to prevent destructive policy updates.

3 Experiments

3.1 Experimental Setup

We utilized a custom simulation environment based on NVIDIA Isaac Lab [4] for both training and evaluation. This setup allowed us to collect millions of steps of experience in a short period. All trainings were conducted on a workstation equipped with a single NVIDIA GeForce RTX 4070 Ti SUPER GPU.

We defined the primary evaluation task as a 3v3 match. We selected this configuration because this is the minimal team size where complex cooperative behaviors begin to emerge, making it an ideal testbed for validating our framework. As a baseline for comparison, we implemented a Scripted Bot. This adversary follows a simple strategy: the robot closest to the ball navigates directly towards it and kick towards the goal.

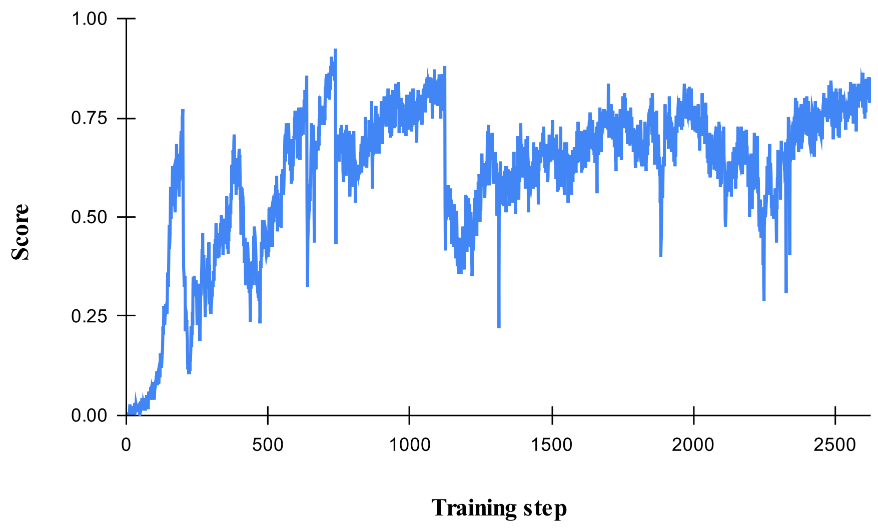


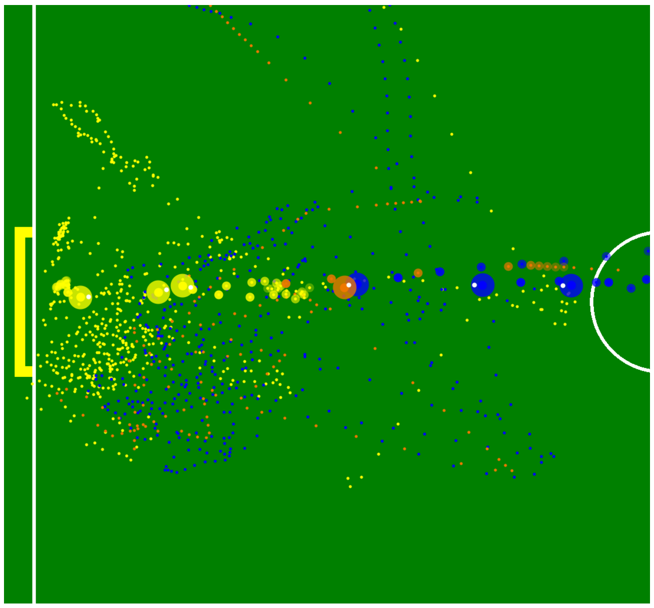
Fig. 3. Evolution of the mean episode score over training steps. Temporary score drops following opponent updates, followed by subsequent recovery.

3.2 Results

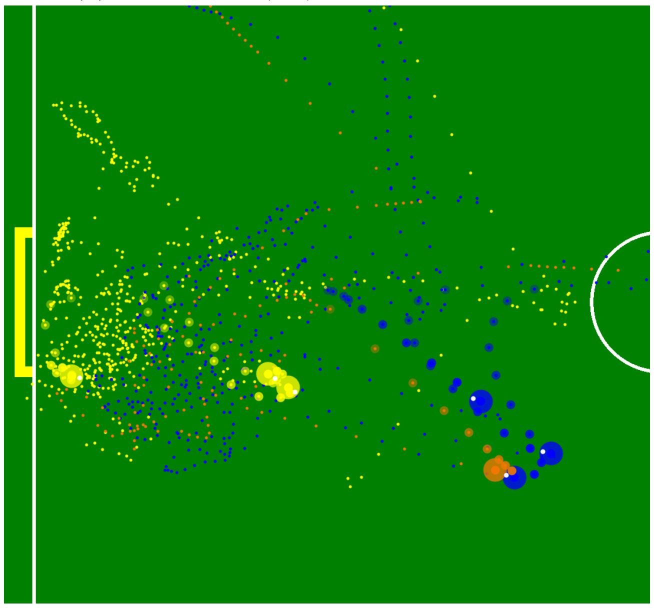
Fig. 3 illustrates the learning curve, depicting the mean episode score over training steps. Notably, a significant drop in score is observed around step 200, 600 and 1200 corresponding to an update in the opponent's policy. However, the subsequent recovery and increase in score demonstrate the agent's ability to adapt and learn counter-strategies against the improved adversary, validating the effectiveness of the continuous adaptation driven by the Self-Play mechanism.

Kicking Behavior To qualitatively evaluate the trained policy, we conducted a visual analysis of match replays in the simulation. A notable behavioral shift was observed: instead of adopting a naive strategy of "pushing" or dribbling the ball all the way into the goal, the agents converged on a strategy that frequently utilizes the kicking mechanism from appropriate distances. We attribute this emergence to the effectiveness of the Dribbling Penalty.

Cooperative Behavior Fig. 4 demonstrates the emergent cooperative backup behavior. While attacking the goal of the Scripted Bot (yellow), a teammate immediately steps in to cover and sustain the offensive when the primary ball carrier loses control. This highlights the team's resilience to individual failures, achieved purely through implicit coordination rather than explicit rules.



(a) Blue robot 1 (left) loses control of the ball.



(b) Blue robot 2 (bottom), having maintained a supportive position, reacts instantaneously to retrieve the loose ball and sustain the attack. This highlights the team's resilience to individual failures, achieved purely through implicit coordination.

Fig. 4. Cooperative backup behavior.

4 Conclusion

In this paper, we presented a multi-robot reinforcement learning approach for RoboCup SSL, building upon the MARLadona framework.

Experimental results demonstrated that our agents could acquire coordinated tactics without the need for manual rule-based tuning. However, full compliance with strict competition rules remains an open challenge. Specifically, we observed instances where agents inadvertently intrude into the defense area during high-speed maneuvers, indicating that the current reward structure alone is insufficient to guarantee hard-constraint adherence.

In future work, we will address three key challenges.

First, we aim to enforce stricter Rule Compliance. While current penalty-based rewards mitigate obvious violations, they are insufficient to prevent infractions like illegal defense area entry. We plan to adopt Constrained Reinforcement Learning approaches to guarantee safety and rule adherence as hard constraints without compromising the team's strategic performance.

Second, we will extend the action space to incorporate Chip Kicks. Integrating 3D ball manipulation capabilities will unlock advanced tacticssuch as lob passes over opponents and aerial clearancesthereby enriching the strategic complexity of the emergent gameplay.

Finally, we will focus on comprehensive Sim-to-Real Deployment. Our ultimate goal is to deploy the trained policies onto the physical TRAPS fleet for full 3v3 competitive matches. We will refine our domain randomization strategies to ensure the policy's robustness against real-world friction variability, vision latency, and mechanical imperfections.

References

1. Matteo Bettini, Ryan Kortvelesy, Jan Blumenkamp, and Amanda Prorok. Vmas: A vectorized multi-agent simulator for collective robot learning. In *International Symposium on Distributed Autonomous Robotic Systems*, pages 42–56. Springer, 2022.
2. Ryoma Mitsuoka, Daichi Miyajima, Naoya Nishiura, Taisuke Tane, Hajime Teruoka, Yasutaka Tsuruta, Masahiro Uchida, Eiki Nagata, Ryuhei Fukuta, Satoru Sakakibara, et al. Traps team description for robocup 2025, 2025.
3. Zichong Li, Filip Bjelonic, Victor Klemm, and Marco Hutter. Marladona – towards cooperative team play using multi-agent reinforcement learning, 2025.
4. Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, Lukasz Wawrzyniak, Milad Rakhsha, Alain Denzler, Eric Heiden, Ales Borovicka, Ossama Ahmed, Iretiayo Akinola, Abrar Anwar, Mark T. Carlson, Ji Yuan Feng, Animesh Garg, Renato Gasoto, Lionel Gulich, Yijie Guo, M. Gussert, Alex Hansen, Mihir Kulkarni, Chenran Li, Wei Liu, Viktor Makoviychuk, Grzegorz Malczyk, Hammad Mazhar, Masoud Moghani, Adithyavairavan Murali, Michael Noseworthy, Alexander Poddubny, Nathan Ratliff, Welf Rehberg, Clemens Schwarke, Ritvik Singh, James Latham Smith, Bingjie Tang, Ruchik Thaker, Matthew Trepte, Karl Van Wyk, Fangzhou Yu, Alex Millane, Vikram Ramasamy, Remo Steiner, Sangeeta Subramanian, Clemens Volk, CY Chen, Neel Jawale, Ashwin Varghese Kuruttukulam, Michael A. Lin, Ajay Mandlekar, Karsten Patzwaldt, John Welsh, Huihua Zhao, Fatima Anes, Jean-Francois Lafleche, Nicolas Moënne-Loccoz, Soowan Park, Rob Stepinski, Dirk Van Gelder, Chris Amevor, Jan Carius, Jumyung Chang, Anka He Chen, Pablo de Heras Ciechomski, Gilles Daviet, Mohammad Mohajerani, Julia von Muralt, Viktor Reutskyy, Michael Sauter, Simon Schirm, Eric L. Shi, Pierre Terdiman, Kenny Vilella, Tobias Widmer, Gordon Yeoman, Tiffany Chen, Sergey Grizan, Cathy Li, Lotus Li, Connor Smith, Rafael Wiltz, Kostas Alexis, Yan Chang, David Chu, Linxi "Jim" Fan, Farbod Farshidian, Ankur Handa, Spencer Huang, Marco Hutter, Yashraj Narang, Soha Pouya, Shiwei Sheng, Yuke Zhu, Miles Macklin, Adam Moravanszky, Philipp Reist, Yunrong Guo, David Hoeller, and Gavriel State. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025.
5. Christian A. Schroeder de Witt, Jakob N. Foerster, Gregory Farquhar, Philip H.S. Torr, Wendelin Böhmer, and Shimon Whiteson. Multi-agent common knowledge reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.
6. Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. In *Advances in Neural Information Processing Systems*, volume 35. Neural information processing systems foundation, 2022.
7. John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.