# Heterogeneous Robots Cooperation via Multi-Agent Reinforcement Learning

Rehab Uddin Shawon*, Siming Liu*, Randall Cox, Joe Davis, Mason Walters, and Robert Powers

Department of Computer Science, Missouri State University, Springfield, MO, USA

Email: rs424s@missouristate.edu (R.U.S); simingliu@missouristate.edu (S.L.); rc284s@missouristate.edu (R.C.); jddavis@spsmail.org (J.D.); waltersmason@gmail.com (M.W.); rpowers@thesummitprep.org (R.P.)

*Corresponding author

***Abstract*—Coordinating heterogeneous robots to perform complex tasks in dynamic environments is challenging due to differences in robot capabilities, task requirements, and coordination constraints. Multi-Agent Reinforcement Learning (MARL) provides a promising framework for enabling decentralized robots to learn cooperative behaviors through interaction with the environment. However, most existing MARL approaches focus on homogeneous robots or single-task scenarios, limiting their applicability to heterogeneous multi-robot systems that must perform multi-step sequential and collaborative tasks. In this study, we propose a MARL framework that enables heterogeneous robots to learn individual, sequential, and collaborative behaviors using a shared policy network. To distinguish robot roles while maintaining knowledge sharing, we introduce Signed Type Encoding (STE), which embeds robot identity in the observation space, enabling a shared policy to learn type-specific behaviors without separate models. Sequential and collaborative tasks are first decomposed into atomic individual tasks that are trained separately, and curriculum learning is then applied to progressively transfer knowledge to multi-step sequential and collaborative tasks, improving training efficiency and stability. Experimental results show that STE significantly improves task specialization and learning stability compared with baselines. Furthermore, task decomposition and curriculum learning enables effective transfer of learned behaviors to more complex coordination tasks, reducing training time while maintaining consistent task completion and coordinated robot behavior.**

***Keywords*—reinforcement learning, multi-agent systems, transfer learning, multi-robot coordination**

## I. Introduction

Coordinating heterogeneous robots, composed of agents with distinct capabilities, roles, and action spaces, poses fundamental challenges for autonomous task execution in dynamic environments. The growing complexity of robotic applications in industrial automation, logistics, and service systems has further intensified the need for robots capable of operating autonomously and collaboratively in uncertain, multi-agent settings. Multi-Agent Reinforcement Learning (MARL) has emerged as one of the most popular frameworks to coordinate heterogeneous robots to perform tasks collectively, efficiently, and adaptively [1]. However, training heterogeneous robots through MARL to handle tasks composed of multiple sequential steps or those requiring several diverse robots to work simultaneously remains a significant challenge (Fig.1). Traditional MARL methods often focus on homogeneous robot settings, where all robots share similar capabilities and objectives. In contrast, many real-world applications involve heterogeneous multi-agent systems (MAS) composed of robots with different physical abilities, task roles, and operational constraints. Training robots in such systems is computationally expensive, time-consuming, and prone to instability due to high-dimensional state spaces and complex robot interactions. A key limitation in existing work is the lack of generalizable learning across different task types. Existing approaches typically rely on separate per-type policies, fixed task structures, or lack explicit type-differentiation mechanisms, all of which restrict scalability and adaptability. Moreover, designing robots that can seamlessly transition from individual behaviors to coordinated group behaviors, such as performing tasks in sequence or simultaneously, is particularly difficult.


(a) Heterogeneous Tasks (b) Sequential Tasks (c) Collaborative Tasks

Fig. 1. Heterogeneous robots performing multiple task types.

In this research, we propose a curriculum learning-based reinforcement learning (RL) framework for heterogeneous MAS capable of performing heterogeneous, sequential, and collaborative tasks using a shared policy. Our approach enables multiple types of robots to share a single policy network that generalizes across different sequential and collaborative task types. To enable shared-policy learning across heterogeneous robots, we embed explicit Signed Type Encoding (STE) in observations, allowing robots to identify their own type and relevant tasks. Unlike one-hot encoding(OHE), which produces sparse binary vectors that limit gradient propagation, STE uses a dense $\pm 1$ representation that improves feature separability and encourages stronger type differentiation during training. Enabled by STE, each type of robot learns to perform its own heterogeneous task, focusing exclusively on its specific goal while ignoring tasks meant for other types. This stage adopts an atomic, per-task learning paradigm in which all robots train in parallel, significantly shortening the total training time and

Manuscript received January 08, 2026; revised February 11, 2026; accepted March 31, 2026; published June 18, 2026.

avoiding unnecessary cross-task interference. Once trained, the shared network is reused to train on more complex sequential and collaborative task structures through curriculum learning. Sequential tasks decompose into type-specific steps that transition upon completion, triggering the next robot in sequence. Collaborative tasks progressively attract different robot types through staged subtask transitions until all required robots gather and execute jointly. This approach reuses pretrained task recognition capabilities, avoiding training from scratch for complex coordination behaviors.

To evaluate the approach, we implemented experiments in the Unity ML-Agents platform across two-type and three-type robot configurations. We compare STE against two type-encoding baselines and evaluate the necessity of curriculum learning by comparing it against direct training from scratch on complex tasks. Results demonstrate that robots effectively specialize in heterogeneous tasks and reuse the same policy for complex multi-stage and multi-robot collaborative behaviors. Statistically significant improvements over all encoding baselines are confirmed through seven independent runs per configuration, with standard deviations, 95% confidence intervals, and independent $t$-tests reported throughout. The key contributions of this work are as follows:

- **Shared-Policy Learning for Heterogeneous Robots:** We propose a MARL framework in which heterogeneous robots share a policy while learning type-specific behaviors through Signed Type Encoding, which outperforms both No Type Encoding and OHE baselines.
- **Curriculum Learning for Sequential and Collaborative Tasks:** We demonstrate that heterogeneous robots trained on individual tasks can be efficiently adapted, via curriculum learning, to perform sequential and collaborative tasks without training from scratch.
- **Comprehensive Evaluation and Generalizability:** We evaluate generalization across heterogeneous, sequential, and collaborative task settings using multiple metrics, including task completion rate, correct and incorrect task interactions, task interaction delay, coordination delay, and cumulative reward, demonstrating reduced training time while maintaining stable coordinated behavior.

This study highlights the potential of combining reinforcement learning, curriculum learning, and multi-agent systems to build scalable, generalizable, and efficient robotic control architectures capable of handling real-world heterogeneity and coordination complexity.

## II. Related Work

Research on MARL has expanded rapidly, particularly in its applications to heterogeneous robotic systems, transfer learning, and collaborative task execution. This section reviews the most relevant literature across these domains and situates the contributions of this paper.

### A. Heterogeneous Multi-Agent Systems

Early investigations into heterogeneous robot coordination explored ontology-based frameworks and curriculum learning between robots of distinct capabilities [2]. More recently, Bettini *et al.* [3] demonstrated efficient cooperative behaviors in heterogeneous multi-robot RL using shared policies. Zhong *et al.* [4] formally analyzed heterogeneous-agent Markov games, providing theoretical guarantees for cooperative and competitive learning settings. Ullah *et al.* [5] proposed a cooperative deep MARL approach integrating visual-based perception for heterogeneous coordination. The applicability of MARL to robotic systems has been surveyed by Low and Zhou [6], Orr and Dutta [7], and Liang *et al.* [8], who discuss Proximal Policy Optimization(PPO) variants, Markov games, and comprehensive MARL taxonomies for multi-robot applications, and comprehensive MARL taxonomies for multi-robot applications, including explainable MARL approaches [9]. Foundational work on actor-critic methods [10] and Q-learning [11] underpins many of these modern MARL algorithms.

Recent studies such as Zhong *et al.* [4] and Yang *et al.* [12] formalized heterogeneous-agent reinforcement learning as cooperative Markov games. Yu *et al.* [13] demonstrated that actor-critic architectures achieve state-of-the-art performance in cooperative multi-agent benchmarks, while Mahajan *et al.* [14] addressed task allocation in mixed environments, recent work applies RL to scheduling and load balancing [15], [16]. Comprehensive reviews by Gautam and Mohan [17] and Wu and Suh [18] provide broader context on the evolution of multi-robot systems and robot learning for collaboration.

In robotics, Prorok *et al.* [19] and Tan [20] showed that policy optimization combined with domain randomization enhances robustness in heterogeneous robot teams. Kurin *et al.* [21] introduced shared-policy gradient architectures enabling simultaneous training across robot morphologies. Similarly, Christiano *et al.* [22] integrated human feedback with MARL to improve agent cooperation and transferability. Specific applications like multi-robot path planning [23] and decentralized exploration [24] further illustrate the breadth of MARL in robotics. However, most existing approaches rely on separate policies or fixed task structures, limiting scalability to heterogeneous robots and multiple task types. Information-sharing mechanisms, such as those explored in semi-centralized [25], [26] and hierarchical frameworks [27], [28], aim to improve coordination but often do not address generalization across different task complexities. We address these gaps by using a shared policy with a novel Signed Type Encoding for parallel training across heterogeneous robots.

### B. Transfer Learning in MARL

Transfer learning in multi-agent settings has been a focal point for improving sample efficiency and adaptability. Da Silva and Costa [29] provided one of the earliest systematic surveys on transfer learning for MARL systems. Yang *et al.* [12] proposed a multi-agent policy-transfer framework based on policy factorization, demonstrating improved scalability and transferability across different domains. Alagha *et al.* [30] applied similar ideas for adaptive target localization under uncertainty, demonstrating performance benefits from inter-

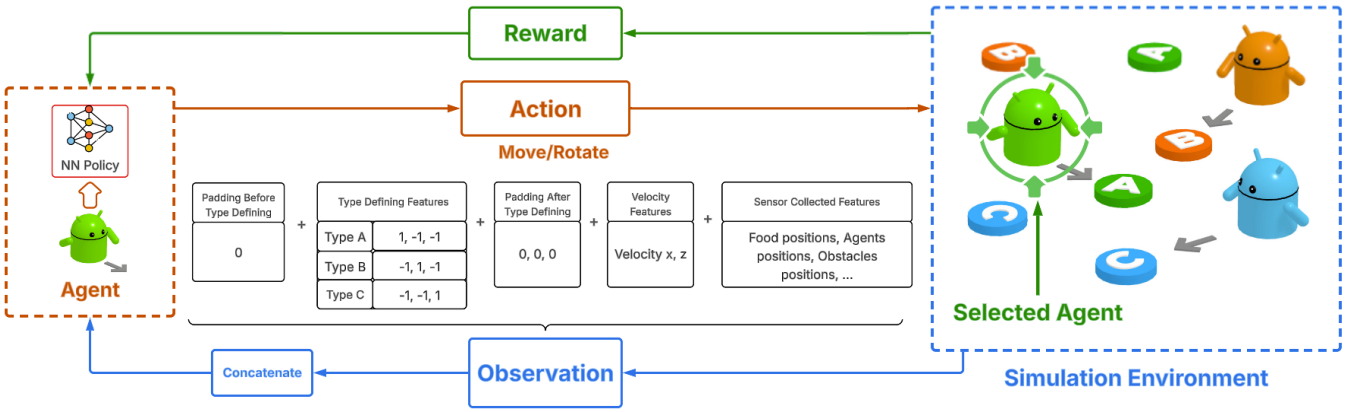


Fig. 2. Heterogeneous multi-agent reinforcement learning framework.

agent knowledge transfer. In heterogeneous systems, Kono *et al.* [2] employed ontology-based mapping to enable shared learning between dissimilar robotic agents. More recent surveys by Zhu *et al.* [31] provide a broader view of transfer learning techniques in deep reinforcement learning. Several recent studies have emphasized transfer learning for complex coordination tasks. Yu *et al.* [13] introduced the MAPPO algorithm, proving its efficiency in cooperative Markov games, while Wang *et al.* [32] and Rashid *et al.* [33] proposed value-decomposition networks that improve reward propagation among agents. Zambaldi *et al.* [34] used relational inductive biases to enhance coordination, demonstrating that knowledge transfer can emerge implicitly through shared representations. Scenario-independent representations have also been explored to facilitate transfer across different tasks and environments [35], [36]. While these methods improve sample efficiency, they typically focus on fixed task structures or homogeneous robots and rely on implicit knowledge sharing. In contrast, our work proposes an explicit curriculum learning strategy that transfers knowledge from learned atomic individual tasks to more complex sequential and collaborative tasks, leveraging a shared policy foundation.

### C. Collaborative MARL Frameworks

Sequential and collaborative task learning has become increasingly relevant for heterogeneous robotic systems. Gupta *et al.* [37] introduced parameter sharing for homogeneous and heterogeneous multi-agent policy gradients, while Foerster *et al.* [38] presented counterfactual multi-agent policy gradients (COMA) for improving credit assignment. Leibo *et al.* [39] analyzed social dilemmas in sequential cooperative games. Recent contributions by Bettini *et al.* [3] and Ullah *et al.* [5] further validated PPO-based cooperative control in physically embodied robot collectives. Foundational algorithms like MADDPG [40] and communication frameworks like DIAL [41] have significantly influenced the design of modern collaborative MARL systems. Finally, recent surveys such as Nguyen *et al.* [42] and Li *et al.* [43] provide comprehensive overviews of the evolution of MARL algorithms, from value-based to actor-critic and model-based methods, emphasizing their growing applicability in heterogeneous robotic teams and collaborative task execution. The practical challenges and applications of MARL are also extensively discussed in works like Canese *et al.* [44] and Hernandez-Leal *et al.* [45]. Nevertheless, most frameworks assume fixed robot roles or require task-specific training, limiting generalization. Information-sharing approaches like those in Siddiqua *et al.* [46] address cooperative aspects but often within a single task paradigm, while recent work has explored explainability to improve transparency of agent behaviors [47]. Our framework addresses this by transferring knowledge from individual heterogeneous tasks to sequential and collaborative behaviors through curriculum learning, enabled by a shared policy with STE. This study proposes a shared-policy MARL framework that incorporates type-defining observations (STE) and curriculum learning, enabling heterogeneous robots to learn individual tasks and progressively transfer learned policies to sequential and collaborative task execution. This approach directly addresses the limitations of existing methods by providing a scalable and generalizable solution for heterogeneous multi-robot coordination.

## III. Methodology

This section presents how MARL was employed to train heterogeneous robots to perform distinct *individual*, *sequential* and *collaborative* tasks using a shared neural network policy and curriculum learning. We model heterogeneous robot systems as Markov Games, which extend Markov Decision Process (MDP) to multi-agent systems. Formally, a Markov game is defined as:

$$G = \langle N, S, \{A_i\}, \{R_i\}, P, \gamma \rangle$$

where $N$ is the number of agents, $S$ the set of environment states, $A_i$ the action space of agent $i$, $R_i$ its reward function, $P$ the transition probability function and the term $\gamma$ represents the discount factor. At each timestep, each agent $i$ observes $o_i \in O_i$, selects an action $a_i \in A_i$ according to its policy $\pi_\theta(a_i|o_i)$, and receives a reward

$$r_i = R_i(s, \mathbf{a}, s') \qquad \mathbf{a} = \{a_1, a_2, ..., a_N\} \tag{1}$$

Here, $\mathbf{a}$ represents the joint action of all agents, and $s'$ is the resulting next state. The shared policy $\pi_\theta$ for all agents is trained using a stable policy optimization algorithm suitable for decentralized execution, which facilitates coordinated learning among the agents. In this study, heterogeneity is expressed through two dimensions: multiple robot types and multiple collaborative task types. Each robot type is specialized in a corresponding task type and is rewarded only for completing its designated task. The environment simulates all task types simultaneously, allowing any robot to interact with any task. However, a robot receives a positive reward only when completing a task of its own type and a negative reward otherwise. This reward structure enforces type-specific learning and specialization.

### A. Heterogeneous Robots Training & Type-Defining Encoding

All robots share a common policy $\pi_\theta$, though they differ in their task specializations. To allow the policy to differentiate robot types, the input observation vector was carefully designed to include each robot's type information explicitly. The observation vector for each robot consisted of:

1) **Type-defining features:** Robot type encoding.
2) **Robot velocity features:** Robot's dynamic state.
3) **Task positions:** Locations of available tasks nearby.
4) **Other robots:** Locations of surrounding robots nearby.
5) **Obstacles:** Locations of static obstacles nearby.

A common method for representing categorical identities in neural networks is OHE, where each robot type is represented by a sparse vector with only one active dimension and the rest set to zero. When passed through the policy network, only the weight corresponding to the active dimension contributes to the activation. Consequently, gradient updates affect only a small subset of parameters, which can limit the network's ability to learn distinctions among robot roles as shown in Eqs. (3) and (4). To address this limitation, we propose STE, a ±1 encoding scheme for robot types. STE provides a dense, zero-centered representation in which all dimensions participate in the computation. When the encoded vector is multiplied by the first-layer weight matrix, all corresponding weights contribute to the activation, improving gradient propagation and encouraging stronger differentiation between robot types. We empirically compared STE with OHE in our experiments and the results show that STE achieves comparable or slightly improved convergence while maintaining stable learning across runs. Based on these theoretical and empirical advantages, STE is adopted in the proposed shared-policy framework.

We define STE as follows:

Let $N$ denote the total number of robot types in the scene, and let $i \in \{1, 2, \ldots, N\}$ be the index of the robot's type. The type-defining encoding vector $\mathbf{e}_i \in \mathbb{R}^{(1+N+3)}$ is defined as:

$$\mathbf{e}_i = \Big[ \underbrace{0}_{\text{pre-padding}}, \mathbf{t}_i, \underbrace{\mathbf{0}_3}_{\text{post-padding}} \Big], \tag{2}$$

where $\mathbf{0}_k$ denotes a zero vector of length $k$, and $\mathbf{t}_i \in \mathbb{R}^N$ is the type-defining component given by:

$$\mathbf{t}_i(j) = \begin{cases} +1, & \text{if } j = i, \\ -1, & \text{if } j \neq i, \end{cases} \quad j = 1, \ldots, N. \tag{3}$$

For $N = 3$, corresponding to $Robot_A$, $Robot_B$, and $Robot_C$, the type-defining encoding are:

$$\mathbf{e}_A = [\,0,\ +1,\ -1,\ -1,\ 0,\ 0,\ 0\,] \tag{4a}$$
$$\mathbf{e}_B = [\,0,\ -1,\ +1,\ -1,\ 0,\ 0,\ 0\,] \tag{4b}$$
$$\mathbf{e}_C = [\,0,\ -1,\ -1,\ +1,\ 0,\ 0,\ 0\,] \tag{4c}$$

This representation ensures that the type-defining features are distinct from all other continuous-valued features and consequently draw greater attention during gradient updates.

**Algorithm 1** Heterogeneous Robots Training - Shared Policy

**Require:** Environment $E$, robots $\{R_1, \ldots, R_N\}$, tasks $\{T_1, \ldots, T_M\}$, policy $\pi_\theta$, value network $V_\phi$, update threshold $K$, epochs $N_{epoch}$, minibatch size $B$, max episodes $E_{max}$

Initialize policy parameters $\theta$, value parameters $\phi$
Initialize experience buffer $\mathcal{B}$
**for** $e = 1$ to $E_{max}$ **do**
  Reset environment and obtain observations $\{o_i\}$
  **for** each timestep until episode termination **do**
    **for** each robot $i = 1$ to $N$ **do**
      $a_i \sim \pi_\theta(a_i|o_i), \quad v_i = V_\phi(o_i)$
    **end for**
    Execute joint action $\{a_i\}$ and observe $\{o'_i, r_i\}$
    Adjust reward based on robot–task compatibility
    Store $(o_i, a_i, r_i, o'_i, v_i)$ in $\mathcal{B}$
    **if** steps $\geq K$ **then**
      Compute advantages and returns
      **for** 1 to $N_{epoch}$ **do**
        Update $\pi_\theta$ and $V_\phi$ with minibatches
      **end for**
      Clear $\mathcal{B}$
    **end if**
    $o_i \leftarrow o'_i$
  **end for**
**end for**
**return** $\pi^*_\theta$

### B. Curriculum Learning for MARL

Training robots from scratch to perform complex behaviors, such as multi-step processes or tasks requiring simultaneous multi-agent coordination is computationally demanding and often unstable. To address this challenge, we incorporate a curriculum learning strategy. After successfully training robots on heterogeneous tasks (Algorithm 1). in parallel, we reuse the learned policy as initialization for learning more complicated sequential and collaborative tasks. This approach leverages the robots' existing abilities to recognize task types, navigate the

environment, and make high-level decisions, thereby reducing the required training time and improving sample efficiency. Once heterogeneous-task training converged, we evaluated whether the learned policy could generalize to more complex scenarios without architectural modifications. Observing that the behaviors transferred to the new environments with minimal disruption, we proceeded to fine-tune the policy for the new tasks. This framework thus enabled efficient learning of advanced behaviors by building upon previously acquired skills rather than discarding them (Algorithm 2).

*1) Sequential Tasks Training:* Sequential tasks extend the learned behaviors by requiring robots to execute multiple dependent stages in a predefined order. In this framework, the previously learned individual task policies are reinterpreted as *subtasks* within a broader sequential structure. For instance, a three-stage sequential task can be represented as:

$$Task_{Seq} = SubTask_A \rightarrow SubTask_B \rightarrow SubTask_C$$

This structure resembles real-world workflows, such as a laundry task can be broken down into washing, drying, and folding, with each step dependent on the completion of the previous one. To reflect this dependency within the simulation, the following progression was implemented:

1) Initially, a task is labeled as $SubTask_A$, which $Robot_A$ recognizes and performs.
2) After completion, the label changes to $SubTask_B$, allowing $Robot_B$ to execute the next step.
3) Finally, the task transitions to $SubTask_C$, and $Robot_C$ completes the final stage.

The sequential task design fully reuses the type-based task recognition mechanism from the initial training stage. After confirming that the agents could perform the sequential steps using the transferred policy, we conducted additional fine-tuning. Each configuration was trained for 10 million further steps to allow the agents to internalize the sequential workflow, reduce unnecessary movement, and improve temporal coordination. This refinement led to more efficient and reliable multi-step execution.

*2) Collaborative Tasks Training:* Collaborative tasks require the simultaneous presence of multiple robots for successful completion, reflecting real-world scenarios where multiple robots must coordinate to achieve shared objectives, such as lifting a heavy object. The design of these tasks parallels the dynamic task-type mechanism employed in the sequential-task framework, but introduces the additional requirement of synchronous multi-agent interaction. To enforce this cooperative behavior, each collaborative task was structured in three stages:

1) $T_{col} = Task_A$, attracting $Robot_A$ to engage the $Task_A$.
2) $Task_A \rightarrow Task_B$, prompting $Robot_B$ to join $Robot_A$.
3) $Task_B \rightarrow Task_C$, at the point $Robot_C$ joins, and $T_{col}.complete = true$.

This collaborative-task formulation leveraged the pretrained policy obtained from the heterogeneous-task training phase.

**Algorithm 2** Re-Training for Sequential/Collaborative Tasks

**Require:** Pretrained $\pi^*_\theta$, environment $E$, task set $\{T_{seq}, T_{col}\}$
**for** each training scenario in $\{T_{seq}, T_{col}\}$ **do**
  **for** each episode $e = 1$ to $E_{max}$ **do**
    **while** task $T$ not complete **do**
      **if** $T = T_{seq}$ **then**
        Execute subtask $T_i$ if $T_{i-1}$ is completed
        $T_{i+1} \leftarrow T_i$ upon completion
      **else if** $T = T_{col}$ **then**
        **for** each robot $R_i \in \{R_1, R_2, ..., R_N\}$ **do**
          **if** $R_i$ completes $T_i$ **then**
            $T_{i+1} \leftarrow T_i$, $R_i$ waits
          **end if**
        **end for**
        **if** all $N$ robots are present at task **then**
          Complete $T_{col}$
        **end if**
      **end if**
      Update $\pi^*_\theta$.
    **end while**
  **end for**
**end for**

### C. Reward Design and Asymmetric Role Balancing

For individual tasks training, a consistent reward structure was used across all experiments where robots received a reward of +2 for performing the correct task associated with their type and a penalty of -1 for performing an incorrect task. For sequential tasks, each robot executes its corresponding subtask in order, and no robot is required to wait for other robot types once its subtask becomes active. Because the task progression naturally activates the next robot in the sequence, the reward structure remains identical to that used in heterogeneous task training. In collaborative tasks, multiple robots must be present simultaneously to complete a task, introducing waiting periods. For example, in a three-robot task, $Robot_A$ typically arrives first and waits for $Robot_B$ and $Robot_C$. Once $Robot_B$ arrives, both still wait until $Robot_C$ arrives. During this period, the waiting robots remain inactive and cannot perform other tasks. When $Robot_C$ finally arrives, the collaborative task is completed, and all robots become available again to perform new tasks. To reduce waiting time and encourage faster completion, we use an asymmetric reward structure for collaborative tasks: +2 for $Robot_A$, +4 for $Robot_B$, and +6 for $Robot_C$, for correct task completion, while incorrect actions remain penalized by -0.1. This design incentivizes later-arriving robots, particularly $Robot_C$ to prioritize task completion, improving coordination efficiency.

## IV. Experimental Setup

This section describes the experimental setup used to evaluate the proposed multi-agent reinforcement learning framework, including the simulation environment, neural network architecture, and training configurations.

### A. Simulation Environment

All experiments were conducted using the **Unity Game Engine** integrated with the **Unity ML-Agents** toolkit [48]. Unity ML-Agents provides an interface between 3D simulation environments (Fig. 2) and Python machine learning libraries via an API, enabling reinforcement learning within physically realistic simulations. To ensure comparability, the total number of robots was fixed at twelve in every experimental setup. In the single-type configuration, all twelve robots were of $Type_A$; in the two-type configuration, six were $Robot_A$ and six were $Robot_B$; and in the three-type configuration, four robots each were assigned to $Robot_A, Robot_B$, and $Robot_C$. Likewise, although task types varied on the experiment, the overall number of tasks remained comparable to all scenarios. By maintaining these consistent scenario conditions, we ensured that variations in performance across experiments were driven by robot heterogeneity rather than by differences in scenario scale or task availability. Sequential and collaborative tasks were then designed using pretrained models. Two- and three-step sequential tasks were developed, where tasks required ordered execution, and two- and three-agent collaborative tasks were designed requiring joint participation. Each scenario was retrained for an additional 10M steps using modified reward structures that reinforced coordination and collaboration. Table I illustrates the detailed scenarios in this study.

TABLE I
TRAINING CONFIGURATIONS FOR HETEROGENEOUS ROBOT TYPES

| Type | Robot A | Robot B | Robot C | Task A | Task B | Task C |
|---|---|---|---|---|---|---|
| 2 Robots/Tasks | 6 | 6 | 0 | 25 | 25 | 0 |
| 3 Robots/Tasks | 4 | 4 | 4 | 20 | 20 | 20 |

### B. Neural Network Configuration

The robots are trained using the PPO algorithm on the Unity ML-Agents framework. The policy network maps the environment observations to discrete actions that control the robots's behavior in Unity. The observation vector obtained from the environment is directly fed into a fully connected feedforward neural network. The network consists of a single hidden layer containing 256 neurons with a Rectified Linear Unit(ReLU) activation function. The network produces two outputs: a policy output that generates the action probabilities and a value output that estimates the state-value function required for PPO optimization. Table II summarizes the neural network architecture used for training the robots.

TABLE II
NEURAL NETWORK ARCHITECTURE

| Layer | Type | Configuration |
|---|---|---|
| Input | Observation Vector | 8,010 (Environment Observations) |
| Hidden | Fully Connected | 256, ReLU activation |
| Policy Output | Fully Connected | 3, action probability distribution |
| Value Output | Fully Connected | 1, state-value estimation |

### C. Evaluation Metrics

To evaluate the effectiveness of STE, we compare it with two baseline approaches: (1) no type-defining encoding and (2) OHE. Multiple runs were performed for each configuration, and statistical analyses were used to determine whether STE provides significant performance improvements over the baselines. The trained policies from this stage were then used as initialization for learning more complex tasks using a curriculum learning strategy. To assess the impact of curriculum learning, we compared this approach with a baseline where robots were trained directly on complex tasks from scratch without curriculum progression. All parameters were kept identical to ensure a fair comparison. After training, we evaluated the best-performing model from each configuration over 1000 steps. To comprehensively assess robot behavior, we introduced several task-level metrics that measure task completion, efficiency, and multi-agent coordination. The evaluation metrics are defined as follows:

- **Cumulative Reward:** The total reward accumulated by all robots within 1000 evaluation steps. This metric reflects the overall effectiveness of the learned policy and the stability and robustness of the training process.
- **Correct Task Interactions:** The number of times a robot complete a task that belong to it's own type within 1000 steps. This metric evaluates the quality of specialization and identifies desired cross-task interactions.
- **Incorrect Task Interactions:** The number of times a robot worked on a wrong task within 1000 steps. This metric evaluates the quality of specialization and identifies undesired cross-task interactions.
- **Task Completion Rate:** The total number of tasks successfully completed within 1000 evaluation steps. This directly measures the robots' ability to accomplish the assigned tasks.
- **Task Interaction Delay:** The average number of steps each task needs to wait after spawning before getting completed by a robot. Lower values indicate greater efficiency and faster task execution.
- **Coordination Delay:** The amount of time robots waited before completing tasks. Lower delay indicates better cooperation and synchronization among robots.

Together, these metrics provide a comprehensive evaluation of task performance, specialization effectiveness, and collaborative efficiency in multi-agent settings.

## V. RESULTS AND DISCUSSION

We simulated trainings and evaluated performance of each experimental setup using different metrics and statistical analysis. All robots were trained from scratch for the heterogeneous individual task setting. The resulting trained policies were then reused for sequential and collaborative tasks, followed by additional fine-tuning as part of curriculum learning. For each scenario, we conducted seven independent training runs with different random seeds and after training, the policies were evaluated in the same simulation environment and the

best performing policy from each set was selected for both two-robot and three-robot settings.

### A. MARL for Heterogeneous Robot Learning

To evaluate the effectiveness of the proposed *Signed Type Encoding*, we compared it with two baseline approaches: *No Type Encoding* and *OHE*. Experiments were conducted using both two-type and three-type robot configurations to analyze how each representation supports heterogeneous task learning as the number of agent types increases. For each configuration, robots were trained to perform heterogeneous tasks using three encoding strategies. Seven independent training runs were conducted for every setup. The reward structure assigns +2 for correct task execution and -1 for incorrect actions. The results were statistically analyzed and evaluated using cumulative reward, correct and incorrect task interactions, task interaction delay, and task completion rate per 1000 evaluation steps.

*1) Training with No Type Encoding:* To examine the importance of explicit type information, robots were first trained without including any type-defining features in their observation vectors. All other training parameters and environmental conditions remained identical. Seven independent runs were conducted for both the two-type and three-type configurations. For the two-type setup 3 out of 7 runs failed to learn meaningful policies, with cumulative rewards remaining close to zero. In other 4 runs, robots occasionally performed correct tasks but frequently interacted with incorrect ones. Because the reward for correct actions is higher than the penalty for incorrect actions, in some runs cumulative rewards showed minor increases in the final values ranging from 65.1 to 75.9. However, the number of incorrect task interactions was comparable to correct interactions, indicating that robots failed to develop consistent task-selection behavior.

The problem becomes more severe in the three-type setup (Fig. 3e). With more robot and task types present, the probability of selecting an incorrect task increases to approximately 66.7% for each robot. As a result, robots remain highly uncertain during training and fail to learn effective policies. Most runs maintain cumulative rewards near zero, and monitoring of the simulations shows minimal task interaction, suggesting that agents did not even learn meaningful movement behaviors. These results demonstrate that without type information, robots cannot distinguish between robot types and therefore fail to associate themselves with the appropriate tasks.

*2) Training with One-Hot Encoding:* Next, we evaluated the commonly used OHE for representing robot identities.

For the two-type configuration (Fig. 3b), performance varied significantly across runs, with final cumulative rewards ranging from 37.2 to 128.6. Although some runs achieved relatively high rewards, detailed observation of the simulations revealed frequent incorrect task interactions alongside correct ones. This indicates that robots learned to move efficiently but did not consistently select the correct tasks. A similar trend appears in the three-type configuration (Fig. 3f), where final cumulative rewards ranged from 20.2 to 47.6. Robots again exhibited comparable numbers of correct and incorrect task interactions, indicating weak task specialization. Overall, OHE allows robots to partially differentiate between robot types and achieve better performance than the no-encoding baseline. However, learning remains unstable and slow, with noticeable variation across runs. This behavior suggests that the sparse nature of OHE limits the effectiveness of gradient-based policy updates.

*3) Training with Signed Type Encoding:* Training with Signed Type Encoding demonstrates significantly more stable and consistent learning behavior across both configurations. Robots are able to reliably identify their types and select the appropriate tasks. The training curves shown in Fig. 3c and Fig. 3g illustrate faster convergence and higher cumulative rewards compared to the baseline methods. In the two-type setup, the final cumulative rewards range from 175.6 to 191.3 across runs, while the three-type setup achieves rewards between 83.1 and 99.2. Unlike the previous approaches, the training curves exhibit minimal variation among the seven runs, indicating stable and robust policy learning. These results suggest that the proposed encoding provides a more informative representation of robot identity. By enabling the shared policy network to effectively differentiate agent roles, the encoding allows agents to correctly associate themselves with their designated tasks and maximize cumulative rewards.

*4) Statistical Analysis and Performance Evaluation:* Fig. 3d and Fig. 3h show the average training performance across seven runs for each encoding strategy. In both the two-type and three-type configurations, the Signed Type Encoding consistently outperforms No Type Encoding and OHE. The comprehensive statistical results are summarized in Table III. The Signed Type Encoding achieves the highest average cumulative rewards, reaching 182.9 for the two-type setup and 92.4 for the three-type setup. It also exhibits the lowest standard deviation among the valid methods, with values of 5.8 and 5.9 for the two-type and three-type configurations, respectively, indicating stable learning across runs. Although the three-type configuration with No Type Encoding shows a near-zero standard deviation, this occurs because the robots fail to learn meaningful behaviors and consistently achieve near-zero rewards; therefore, this result does not provide meaningful insight. The proposed encoding also demonstrates the most favorable 95% confidence intervals for both configurations. Table IV presents the statistical significance analysis. Independent t-tests were conducted between the proposed encoding and each baseline method for both configurations. In all comparisons, the p-values were below 0.005, confirming that the performance improvements of the proposed encoding are statistically significant.

Further performance comparisons were conducted during the evaluation phase using the best-performing run from each configuration. The results are summarized in Table V. For the two-type setup, the proposed method achieves the highest number of correct task interactions (38.6 and 43.3), the lowest number of incorrect task interactions (2.3 and 3.3), the highest task completion rate (81.9), and the highest cumulative reward (158.2). Although OHE shows slightly lower task interaction

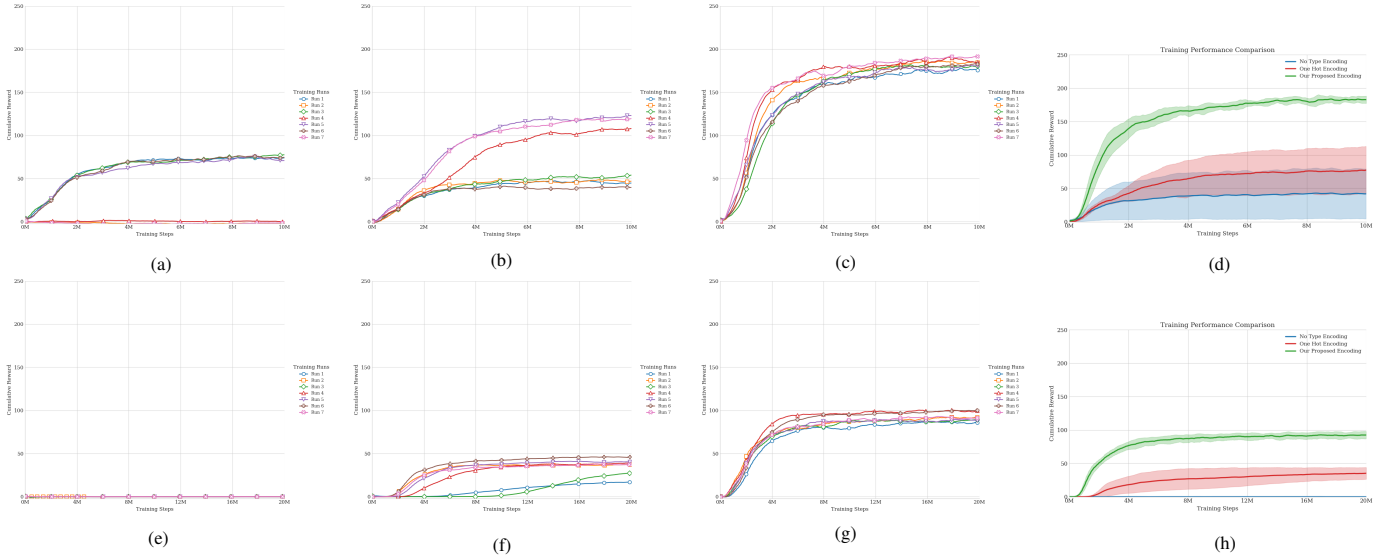


Fig. 3. Training comparison of different encodings for 2-Type and 3-Type setup. (a) No Type Encoding (2-Types);(b) One Hot Encoding (2-Types); (c) Signed Type Encoding (2-Types); (d) Comparison of the Avg. (2-Types); (e) No Type Encoding (3-Types); (f) One Hot Encoding (3-Types); (g) Signed Type Encoding (3-Types);(h) Comparison of the Avg. (3-Types).

TABLE III
STATISTICAL ANALYSIS OF PERFORMANCE ACROSS ENCODINGS

| Metric | No Type | | One Hot | | Signed Type | |
|---|---|---|---|---|---|---|
| | 2Types | 3Types | 2Types | 3Types | 2Types | 3Types |
| Cum. Reward | 41.8 | 0.0 | 77.2 | 35.1 | **182.9** | **92.4** |
| Std. Dev. | 40.3 | **0.0** | 38.6 | 9.5 | **5.8** | 5.9 |
| 95% CI | (4.4,79.1) | (0,0) | (41.5,112.9) | (26.3,43.8) | **(177.4,188.3)** | **(87.0,97.9)** |

TABLE IV
STATISTICAL ANALYSIS BETWEEN ENCODING METHODS

| Metric | No Type vs Signed Type | | One Hot vs Signed Type | |
|---|---|---|---|---|
| | 2-Type | 3-Type | 2-Type | 3-Type |
| T-Statistic | 9.14 | 41.4 | 7.15 | 13.5 |
| p-value | $0.0007 < 0.05$ | $0 < 0.05$ | $0.003 < 0.05$ | $0 < 0.05$ |
| Stat. Significance | **Yes** | **Yes** | **Yes** | **Yes** |

TABLE V
PERFORMANCE COMPARISON PER 1000 EVALUATION STEPS ACROSS DIFFERENT TYPE SETUPS

| Setup | Encoding | Right Task Interaction | | | Wrong Task Interaction | | | Task Interaction Delay | | | Task Completion | Cum. Reward |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Task A | Task B | Task C | Task A | Task B | Task C | Task A | Task B | Task C | | |
| 2-Type Setup | No Type | 14.5 | 18.2 | – | 17.2 | 18.1 | – | 6.1 | 5.6 | – | 32.7 | 30.4 |
| | One Hot | 23.4 | 23.3 | – | 25.4 | 25.9 | – | **3.9** | **4.0** | – | 46.7 | 46.1 |
| | Our Proposed | **38.6** | **43.3** | – | **2.3** | **3.3** | – | 4.9 | 4.3 | – | **81.9** | **158.2** |
| 3-Type Setup | No Type | 0.2 | 0.1 | 0 | **0** | **2.0** | **0** | 10.1 | 29.9 | 30.0 | 0.3 | 0.2 |
| | One Hot | 9.4 | 0.4 | 1.8 | 1.4 | 7.0 | 10.8 | **6.5** | 43.5 | 50.57 | 11.6 | 4.0 |
| | Our Proposed | **11.3** | **19.0** | **14.1** | 1.5 | 3.0 | 2.1 | 10.5 | **7.2** | **8.9** | **44.4** | **82.2** |

delay, this improvement is not meaningful because the agents frequently interact with incorrect tasks, indicating inefficient task selection. For the three-type setup, the proposed encoding again achieves the highest correct task interactions (11.3, 19.0, and 14.1), the highest task completion rate (44.4), and the highest cumulative reward (82.2). The No Type Encoding configuration appears to produce the lowest number of incorrect interactions; however, this occurs because the robots fail to learn meaningful movement behaviors and rarely interact with tasks. In terms of task interaction delay, the proposed encoding achieves the lowest average delay for $Task_B$ and $Task_C$ (7.2 and 8.9, respectively).

Overall, both the statistical analysis and evaluation metrics confirm that the proposed Signed-Type Encoding significantly improves heterogeneous task learning. Across both two-type and three-type configurations, it demonstrates superior training stability, better task specialization, and higher overall performance compared with the No Type Encoding and OHE baselines.

### B. Curriculum Learning for Sequential Tasks

After training the robots on individual tasks using the Signed Type Encoding, the learned policies were reused as initialization for sequential task learning. Additional training was then performed to enable agents to complete multi-step task sequences. Specifically, two-step sequential tasks were designed for the two-type robot setup, and three-step sequential tasks for the three-type robot setup. For the two-type configuration, training resumed from the previously trained model at 10M steps and continued to 20M steps. The training curves for all seven runs are shown in Fig. 4b. Although cumulative rewards varied across runs, the overall performance improved after curriculum learning. As shown in Fig. 4d, the mean cumulative reward increased from 182.9 (std. dev. 5.8) before curriculum learning to 206.1 (std. dev. 14.7) after sequential-task training. For the three-type configuration, training resumed from 20M steps and continued to 30M steps.

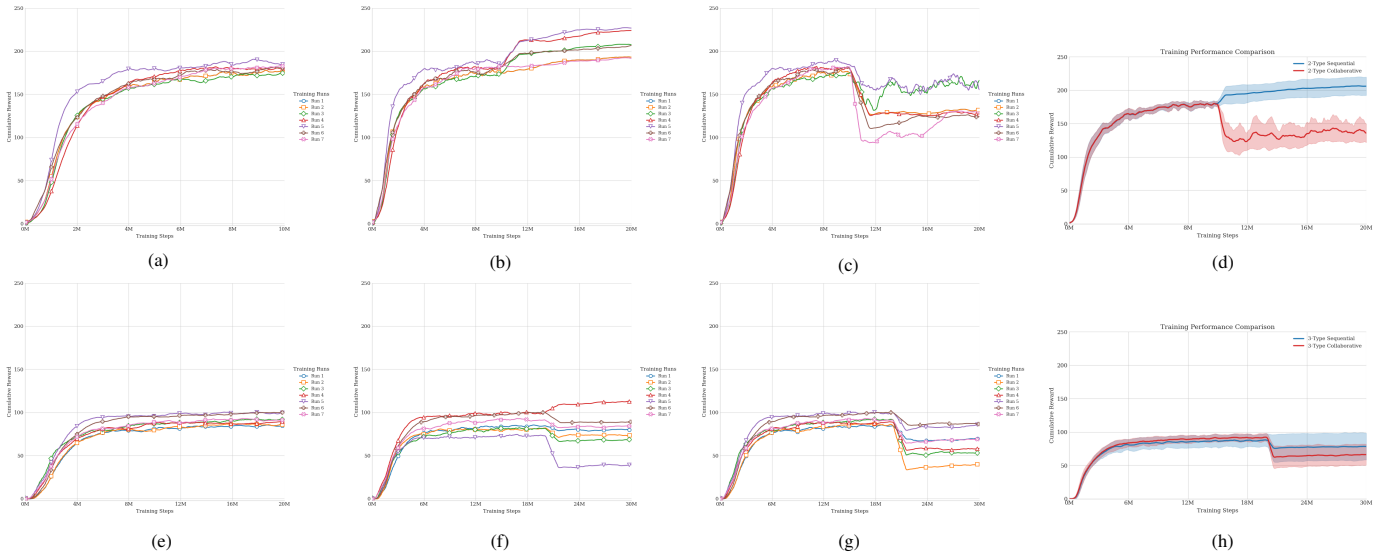


Fig. 4. Visualization of training for 2 and 3 types of robots performing heterogeneous, sequential and collaborative tasks. (a) Heterogeneous (2-Type); (b) Sequential (2-Type); (c) Collaborative (2-Type); (d) Seq. and Col. Mean values (2-Type); (e) Heterogeneous (3-Type); (f) Sequential (3-Type); (g) Collaborative (3-Type); (h) Seq. and Col. Mean values (3-Type).

TABLE VI
PERFORMANCE OF OUR PROPOSED APPROACH PER 1000 EVALUATION STEPS ACROSS DIFFERENT TASK SETTINGS

| Task Setting | Setup | Right Task Interaction | | | Wrong Task Interaction | | | Task Interaction Delay | | | Waiting Time | | | Task Completion | Cum. Reward |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Task A | Task B | Task C | Task A | Task B | Task C | Task A | Task B | Task C | Robot A | Robot B | Robot C | | |
| Heterogeneous | 2-Type | 38.6 | 43.3 | – | 2.3 | 3.3 | – | 4.9 | 4.3 | – | – | – | – | 81.9 | 158.2 |
| | 3-Type | 11.3 | 19.0 | 14.1 | 1.5 | 3.0 | 2.1 | 10.5 | 7.2 | 8.9 | – | – | – | 44.4 | 82.2 |
| Sequential | 2-Type | 46.9 | 46.0 | – | 2.1 | 1.2 | – | 6.3 | 3.6 | – | – | – | – | 92.9 | 182.5 |
| | 3-Type | 18.5 | 19.5 | 18.0 | 1.1 | 2.8 | 0.9 | 6.7 | 13.4 | 5.7 | – | – | – | 56.0 | 107.2 |
| Collaborative | 2-Type | 21.8 | 21.7 | – | 5.8 | 3.2 | – | 2.18 | 2.21 | – | 1.45 | – | – | 43.5 | 78.0 |
| | 3-Type | 6.6 | 6.4 | 6.2 | 0 | 2.7 | 5.3 | 6.1 | 6.3 | 6.4 | 3.7 | 3.2 | – | 19.2 | 30.4 |

Fig. 4f shows the training curves for all seven runs. In this case, cumulative rewards varied more significantly across runs. As shown in Fig. 4h, the mean cumulative reward decreased from 92.4 (std. dev. 5.9) before curriculum learning to 78.3 (std. dev. 21.5) after sequential-task training. This reduction reflects the increased complexity introduced by longer task sequences and a larger number of agent types. Nevertheless, the agents successfully learned to perform the sequential tasks.

*1) Performance Evaluation:* Table VI summarizes the evaluation results using the best-performing run for each configuration over 1000 evaluation steps. For the two-type setup, curriculum learning improves overall performance compared with the heterogeneous-task setting. Specifically, there is a 15% increase in correct task interactions, a 13.4% increase in task completion rate, and a 15.1% increase in cumulative reward. In addition, incorrect task interactions decrease by approximately 40%, while task interaction delay remains largely unchanged. For the three-type setup, the increased number of agents and the longer task sequences introduce greater learning complexity, leading to reduced performance across several metrics during training. However, evaluation results still indicate improvements compared with the heterogeneous-task baseline. In particular, correct task interactions increase by 26.7%, task completion rate improves by 30%, cumulative reward increases by 15.1%, and incorrect task interactions decrease by 37.5%, with no significant change in task interaction delay. Overall, these results demonstrate that our approach can extend previously learned heterogeneous-task policies to more complex sequential tasks. The evaluation results indicate improved task execution and behavioral coordination in both the two-type and three-type configurations.

*2) Learned Sequential Behavior:* Further analysis shows that the heterogeneous robots successfully execute a three-step sequential task, as illustrated in Fig. 5. Specifically, $Robot_A$ first completes $Task_A$ and transitions the task to $Task_B$, followed by $Robot_B$ executing $Task_B$ and advancing it to the final step, $Task_C$. Finally, $Robot_C$ completes $Task_C$, thereby accomplishing the full sequential $Task_{seq}$. This emergent coordination demonstrates that the shared policy can effectively capture ordered, type-dependent collaboration without explicit inter-robot communication.


Fig. 5. Learned sequential behavior.

### C. Curriculum Learning for Collaborative Tasks

After demonstrating that the proposed MARL framework can learn sequential multi-step tasks, we further extended the experiments to collaborative tasks that require multiple robots to act simultaneously to complete a task. The policies learned from heterogeneous task training using the Signed Type Encoding were reused as initialization for collaborative task learning. Additional training was then performed to enable coordinated task execution among multiple robots. For the two-type configuration, training resumed from the previously trained model at 10M steps and continued to 20M steps. The training curves for all seven runs are shown in Fig. 4c. Although cumulative rewards varied across runs, the overall performance decreased after introducing collaborative tasks. As shown in Fig. 4d, the mean cumulative reward decreased from 182.9 (std. dev. 5.8) before curriculum learning to 135.8 (std. dev. 15.1) after collaborative-task training. For the three-type configuration, training resumed from 20M steps and continued to 30M steps. Fig. 4g presents the training curves for all seven runs. In this case, cumulative rewards show larger variations across runs. As shown in Fig. 4h, the mean cumulative reward decreased from 92.4 (std. dev. 5.9) before curriculum learning to 66.16 (std. dev. 16.8) after collaborative-task training. The reduction in cumulative reward for both configurations reflects the increased coordination requirements of collaborative tasks. Robots often need to wait at task locations until other required agents arrive, which limits mobility and reduces the frequency of task interactions. Nevertheless, the robots successfully learned to coordinate and complete the collaborative tasks.

*1) Performance Evaluation:* Table VI summarizes the evaluation results using the best-performing run for each configuration over 1000 evaluation steps for heterogeneous, sequential, and collaborative tasks. For the two-type setup, collaborative tasks introduce a significant increase in coordination complexity compared with the heterogeneous-task setting. As a result, correct task interactions decrease by 47%, task completion rate decreases by 53.1%, and cumulative reward decreases by 50.6%. In addition, incorrect task interactions increase by approximately 33% on average, while task interaction delay decreases by about 52%. The average waiting time for $Robot_A$ is 1.45 seconds, while $Robot_B$, which arrives last at the collaboration point, experiences no waiting time. For the three-type setup, the performance reduction becomes more pronounced due to the higher coordination requirements among three agents. Correct task interactions decrease by 56%, task completion rate decreases by 56.7%, and cumulative reward decreases by 62.9%. Incorrect task interactions increase by approximately 45% on average, while task interaction delay decreases by about 29%. The average waiting times for $Robot_A$ and $Robot_B$ are 3.7 and 3.2 seconds, respectively, while $Robot_C$, which arrives last, does not experience waiting time. Overall, these results demonstrate that our approach can extend previously learned heterogeneous-task policies to more complex collaborative tasks. The observed performance reduction is mainly due to the coordination requirements of collaborative tasks, where agents that arrive earlier must wait for others before the task can be completed. This additional waiting time is unique to collaborative tasks and does not occur in heterogeneous or sequential task settings. Despite this challenge, the agents successfully learn coordinated behaviors, demonstrating effective collaboration in both the two-type and three-type configurations.

*2) Learned Collaborative Behavior:* Fig. 6 illustrates the learned collaborative behavior for $Task_{col}$, which requires three robots to act simultaneously. In the three-type setup, the task progresses as $Task_A \rightarrow Task_B \rightarrow Task_C$. Specifically, $Robot_A$ initiates the task by completing $Task_A$ and waiting, $Robot_B$ then executes $Task_B$ and waits, and finally $Robot_C$ arrives to satisfy the three-robot requirement, enabling all robots to jointly complete the collaborative task.

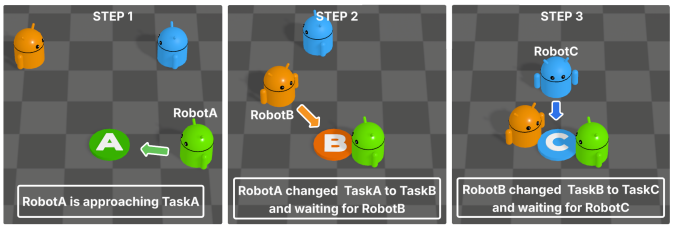


Fig. 6. Learned collaborative behavior.

### D. Effectiveness of Curriculum Learning

To evaluate the role of curriculum learning, we compared training with and without curriculum learning for collaborative tasks. In the curriculum learning setting, agents were first trained on simpler heterogeneous tasks and the learned policy was then used to initialize training for more complex collaborative tasks. As a baseline, we also trained robots directly on collaborative tasks from scratch. A three-type robot configuration was trained using the proposed type-defining encoding while keeping all other parameters unchanged. The robots were trained for 30 million steps, and seven independent runs were conducted. The results show that robots failed to learn

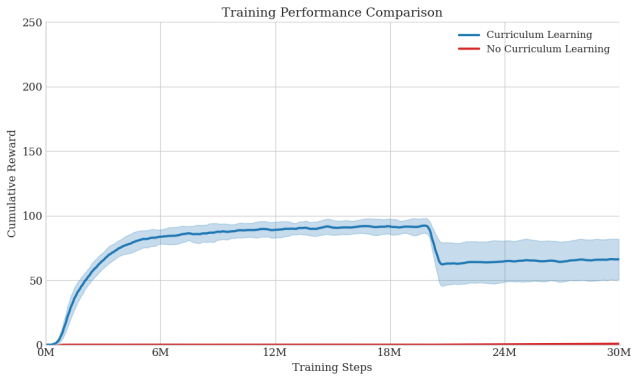


Fig. 7. Training w/o curriculum learning using STE.

meaningful collaborative behaviors when trained from scratch. Across all runs, cumulative rewards remained close to zero

throughout training, indicating that the agents were unable to discover effective coordination strategies. As shown in Fig. 7, the average cumulative reward across all runs remained near zero and stable during the entire training process. In practice, the agents did not learn meaningful movement behaviors and therefore collected almost no rewards or penalties. In contrast, when curriculum learning was applied, robots were first trained on heterogeneous tasks for 20 million steps. During this stage, they learned fundamental skills such as navigation, task recognition, and role specialization. The learned policy was then transferred and fine-tuned for collaborative tasks from 20M to 30M steps. As shown in Fig. 7, curriculum learning enabled robots to successfully learn collaborative behaviors. The cumulative reward reached 92.4 during the heterogeneous-task training phase (0-20M steps) and stabilized around 66.16 after transitioning to collaborative-task training between 20M and 30M steps. These results demonstrate that curriculum learning plays a critical role in enabling agents to learn complex collaborative behaviors. By gradually increasing task complexity, the proposed framework improves learning stability and allows robots to effectively coordinate in heterogeneous multi-agent environments.

## VI. Conclusion and Future Work

In this work, we presented a curriculum learning-based MARL framework for heterogeneous robot coordination. The framework introduces Signed Type Encoding (STE) to enable a shared policy to distinguish robot identities and learn type-specific behaviors. Compared against No Type Encoding and One-Hot Encoding (OHE) baselines, STE achieves statistically significant improvements in cumulative reward, task completion, and task specialization across all configurations for both two-type and three-type setups. Experimental results show that robots effectively learned individual tasks from scratch and transferred this knowledge to complex sequential and collaborative tasks through curriculum learning. A key finding is that increasing robot-type count amplifies task ambiguity and coordination overhead, as evidenced by reduced cumulative rewards in three-type sequential and collaborative settings. This suggests that reward shaping and STE become increasingly critical as system heterogeneity scales. Despite this, learned policies consistently demonstrated correct task specialization and coordinated behavior across all scenarios.

Several directions remain for future work. First, we plan to scale the framework to environments with a larger number of robot types to assess generalization under greater heterogeneity. Second, we aim to investigate attention-based or graph neural network architectures for the observation encoder, which may better capture inter-agent relational structure and reduce coordination delay. Third, the current asymmetric reward structure for collaborative tasks uses manually tuned values; future work will explore adaptive or learned reward shaping mechanisms to remove this dependency. Finally, we intend to evaluate the framework on physical robotic platforms, addressing sim-to-real transfer challenges including sensor noise, actuation uncertainty, and communication delays.

## Conflict of Interest

The authors declare no conflict of interest.

## Author Contributions

RUS conceived and designed the experiments, led the primary drafting of the manuscript, and revised it for significant intellectual content; RC contributed to experimental validation; JD, MW, and RP conducted simulation fabrication, data collection, sample analysis, and figure preparation; SL provided research guidance and domain expertise; all authors had approved the final version of the manuscript.

## Acknowledgment

This material is based upon work supported by the National Science Foundation under Award No. 2302060. Joe Davis, Mason Walters, and Robert Powers are secondary school teachers who contributed to this work as affiliated collaborators of the Department of Computer Science, Missouri State University. Their home institutions are: Parkview High School (J. Davis, Mathematics Teacher), Kickapoo High School (M. Walters, Business Teacher), and The Summit Preparatory School (R. Powers, Science/STEM Teacher).

## References

[1] L. Buşoniu, R. Babuška, B. D. Schutter, and D. Ernst, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 156–172, 2008.

[2] H. Kono, A. Kamimura, and K. Tomita, "Transfer learning using ontology for heterogeneous multi-agent systems," in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 2014.

[3] M. Bettini, A. Shankar, and A. Prorok, "Heterogeneous multi-robot reinforcement learning," *arXiv preprint arXiv:2301.07137*, 2023. [Online]. Available: https://arxiv.org/abs/2301.07137

[4] Y. Zhong, J. G. Kuba, X. Feng, S. Hu, J. Ji, and Y. Yang, "Heterogeneous-agent reinforcement learning," *Journal of Machine Learning Research*, vol. 25, 2024. [Online]. Available: http://www.jmlr.org/papers/v25/23-0488.html

[5] R. Ullah, M. Saeed, W. Ali, J. Nazar, and F. Nazar, "A cooperative heterogeneous multi-agent system leveraging deep reinforcement learning," *KADSA Journal*, 2025.

[6] D. Low and Y. Zhou, "Cooperative multi-agent reinforcement learning for robotic systems: A review," *Journal of Robotics and Autonomous Systems*, 2025.

[7] J. Orr and A. Dutta, "Multi-agent deep reinforcement learning for multi-robot applications: A survey," *Sensors*, vol. 23, no. 7, p. 3625, 2023.

[8] J. Liang, H. Miao, K. Li, J. Tan, X. Wang, R. Luo, and Y. Jiang, "A review of multi-agent reinforcement learning algorithms," *Electronics*, vol. 14, no. 4, p. 820, 2025.

[9] R. U. Shawon, S. Liu, and A. Siddiqua, "Explainable reinforcement learning for multi-agent systems," in *2025 IEEE 37th International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2025, pp. 1339–1343.

[10] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," in *Advances in Neural Information Processing Systems*, vol. 12, 1999.

[11] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, 1992.

[12] T. Yang, W. Wang, H. Tang, and J. Hao, "An efficient transfer learning framework for multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[13] C. Yu, F. Luo, Y. Yang, M. Zhang, H. Xu, J. Wang, W. Liu, and Y. Zhang, "The surprising effectiveness of mappo in cooperative multi-agent games," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 8749–8762.

[14] A. Mahajan, B. Peng, Y. Cao, R. Lowe, and J. Foerster, "Tarmac: Task allocation with multi-agent cooperation," in *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2022.
[15] S. Sakib, A. Katangur, and R. Dubey, "Reinforcement learning based adaptive task scheduling in cloud environments," in *2025 IEEE 25th International Conference on Communication Technology (ICCT)*. IEEE, 2025, pp. 1836–1843.
[16] ——, "A dynamic approach to load balancing in cloud infrastructure: Enhancing energy efficiency and resource utilization," in *2025 IEEE Cloud Summit*. IEEE, 2025, pp. 87–94.
[17] A. Gautam and S. Mohan, "A review of research in multi-robot systems," in *2012 IEEE 7th international conference on industrial and information systems (ICIIS)*. IEEE, 2012, pp. 1–5.
[18] B. Wu and C. S. Suh, "State-of-the-art in robot learning for multi-robot collaboration: A comprehensive survey," *arXiv preprint arXiv:2408.11822*, 2024.
[19] A. Prorok, "Learning robust heterogeneous robot coordination with marl," *Science Robotics*, vol. 4, no. 31, p. eaaw4513, 2019.
[20] M. Tan, "Sim-to-real transfer for multi-agent learning in robotics," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
[21] V. Kurin, J. Luketina, G. Farquhar, and J. Foerster, "Shared-policy gradients for multi-agent reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2021.
[22] P. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
[23] Y. Yang, L. Juntao, and P. Lingling, "Multi-robot path planning based on a deep reinforcement learning dqn algorithm," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 3, pp. 177–183, 2020.
[24] A. H. Tan, F. P. Bejarano, Y. Zhu, R. Ren, and G. Nejat, "Deep reinforcement learning for decentralized multi-robot exploration with macro actions," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 272–279, 2022.
[25] A. Harris and S. Liu, "Maidrl: Semi-centralized multi-agent reinforcement learning using agent influence," in *2021 IEEE Conference on Games (CoG)*. IEEE, 2021, pp. 01–08.
[26] D. Xie and X. Zhong, "Semicentralized deep deterministic policy gradient in cooperative starcraft games," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 4, pp. 1584–1593, 2020.
[27] A. Siddiqua, S. Liu, R. Iqbal, F. A. Irfan, L. Ross, and B. Zweerink, "Tim-marl: Information sharing for multi-agent reinforcement learning in smart environments," in *2024 IEEE 21st Consumer Communications Networking Conference (CCNC)*, 2024, pp. 1044–1045.
[28] A. Siddiqua, S. Liu, K. Mouser, and M. Harris, "Awareness map enabled semi-centralized marl for multi-robot collaboration," in *2025 IEEE 22nd Consumer Communications Networking Conference (CCNC)*, 2025, pp. 1–6.
[29] F. L. Da Silva and A. H. R. Costa, "A survey on transfer learning for multi-agent reinforcement learning systems," *Journal of Artificial Intelligence Research*, vol. 64, pp. 645–703, 2019.
[30] A. Alagha, R. Mizouni, S. Singh, J. Bentahar, and H. Otrok, "Adaptive target localization under uncertainty using multi-agent deep reinforcement learning with knowledge transfer," *AI Open*, 2025.
[31] Z. Zhu, K. Lin, A. K. Jain, and J. Zhou, "Transfer learning in deep reinforcement learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.
[32] Q. Wang, T. Rashid, M. Samvelyan, C. S. de Witt, Y. Yang, J. Tan, and B. Li, "Qmix: Monotonic value function factorization for deep multi-agent reinforcement learning," in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.
[33] T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorization for deep multi-agent rl," in *AAAI Conference on Artificial Intelligence*, 2018.
[34] V. Zambaldi, D. Raposo, A. Santoro, V. Bapst, Y. Li, I. Babuschkin, K. Tuyls, D. Reichert, D. Silver, and T. Lillicrap, "Relational deep reinforcement learning," *arXiv preprint arXiv:1806.01830*, 2018.
[35] A. S. Nipu, S. Liu, and A. Harris, "Enabling multi-agent transfer reinforcement learning via scenario independent representation," in *2023 IEEE Conference on Games (CoG)*. IEEE, 2023, pp. 1–8.
[36] A. Siddiqua, S. Liu, A. S. Nipu, A. Harris, and Y. Liu, "Co-evolving multi-agent transfer reinforcement learning via scenario independent representation," *IEEE Access*, 2024.
[37] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2017.
[38] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *AAAI Conference on Artificial Intelligence*, 2018.
[39] J. Z. Leibo, V. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel, "Multi-agent reinforcement learning in sequential social dilemmas," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 114, no. 43, pp. 11 354–11 359, 2017.
[40] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
[41] J. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.
[42] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "A comprehensive survey on multi-agent deep reinforcement learning," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 12, no. 4, pp. 637–659, 2020.
[43] S. Li, J. Hao, X. Lu, D. Li, Z. Wang, Y. Chen, C. Fan, and J. Zhang, "A survey of multi-agent reinforcement learning: Advances and applications," *ACM Computing Surveys*, vol. 54, no. 10s, pp. 1–38, 2021.
[44] L. Canese, G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, D. Giardino, M. Re, and S. Spanò, "Multi-agent reinforcement learning: A review of challenges and applications," *Applied Sciences*, vol. 11, no. 11, p. 4948, 2021.
[45] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 6, pp. 750–797, 2019.
[46] A. Siddiqua, S. Liu, R. Iqbal, L. Ross, B. Zweerink, and R. Eskridge, "Information sharing for cooperative robots via multi-agent reinforcement learning," in *2024 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*. IEEE, 2024, pp. 1–7.
[47] Z. Faruqui, M. S. McIntire, R. Dubey, and J. McEntee, "Explainability of cnn based classification models for acoustic signal," in *2025 IEEE 37th International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2025, pp. 1087–1094.
[48] A. Juliani, V.-P. Berges, E. Vckay, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) Demonstrations Track*, 2018. [Online]. Available: https://arxiv.org/abs/1809.02627