2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Bi-CL: A Reinforcement Learning Framework for Robots Coordination Through Bi-level Optimization

Zechen Hu, Daigo Shishika, Xuesu Xiao, and Xuan Wang

***Abstract*— In multi-robot systems, achieving coordinated missions remains a significant challenge due to the coupled nature of coordination behaviors and the lack of global information for individual robots. To mitigate these challenges, this paper introduces a novel approach, Bi-level Coordination Learning (Bi-CL), that leverages a bi-level optimization structure within a CTDE paradigm. Our bi-level reformulation decomposes the original problem into a reinforcement learning level with reduced action space, and an imitation learning level that gains demonstrations from a global optimizer. Bi-CL further integrates an alignment penalty mechanism, aiming to minimize the discrepancy between the two levels without degrading their training efficiency. We introduce a running example to conceptualize the problem formulation. Simulation results demonstrate that Bi-CL can learn more efficiently and achieve comparable performance with traditional multi-agent reinforcement learning baselines for multi-robot coordination.**

I. Introduction

Multi-robot systems have extensive applications in various engineering fields, but their deployment relies on scalable algorithms that enable robots to make coordinated and sequential decisions using local observation [2], [3]. To this end, Centralized Training with Decentralized Execution (CTDE) [4] emerges as a promising approach, offering a balanced framework for coordinating multiple robots through centralized learning processes that guide decentralized operational decisions. A similar case applies to Proximal Policy Optimization (PPO) algorithm and its generalization Multi-agent PPO [5]. While these approaches might be concerned with state dimensionality during centralized training, recent advances, such as Value-Decomposition Networks (VDN), can be leveraged to decompose the joint value function into individual value functions for each robot [6]. Building on this, QMIX [7] further extends the framework, allowing for a more complex, state-dependent mixing of individual robot's value functions to learn coordinated behaviors. However, the effectiveness of CTDE faces significant challenges as the dimensionality of action spaces expands and each robot only has local observation of the system. These challenges complicate the training process and potentially impedes practical deployment.

In this context, we note that multi-robot cooperative missions often exhibit inherent hierarchical structures, allowing them to be decomposed into high-level and low-level tasks [8]. For instance, rescue missions utilize teams of robots to search large or hazardous areas [9], [10], where a high-level planning task may focus on area coverage, and a low-level task may address navigation and obstacle avoidance. A similar scenario arises from warehouse automation [11], where high-level tasks involve managing robot fleets for picking, packing, and sorting, and low-level tasks ensure precise and safe navigation in densely populated environments. Nevertheless, in most cases, the decomposed problems are internally coupled by state dependence, which can not be solved independently. Bi-level optimization [12] presents a solution to this issue, with the capability of enhancing learning efficiency and stability while maintaining the explicit connections between the two levels of problems.

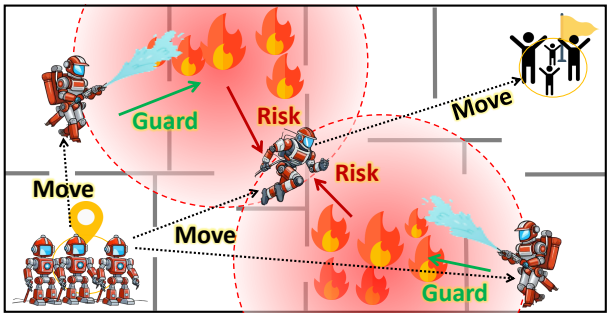

Fig. 1: A running example for a firefighting scenario. Robots can simultaneously perform two actions: move (to where) and guard (which adversary). Team reward depends on the risk of fire, which is a coupled function of both actions.

Bi-level optimization is a hierarchical mathematical formulation where the solution of one optimization task is restricted by the solution set mapping of another task [12]. In recent years, this technique has been incorporated with various machine learning methods for nested decision-making, including multi-task meta-learning [13], neural architecture search [14], adversarial learning [15], and deep reinforcement learning [16]. The applications of these bi-level learning approaches span economics, management, optimal control, systems engineering, and resource allocation problems [17]–[21], with a comprehensive review by Liu et al. [12]. Corresponding to the scope of this paper, when solving multi-robot coordination problems, bi-level decomposition has been introduced [22], [23] for actor-critic MARL to reduce the training burden of each level, thus improving the training efficiency and convergence. Le et al. [24] uses latent variables to introduce a mechanism similar to bi-level optimization that can significantly improve the performance of multi-agent imitation learning.

However, despite the abundance of research on addressing such coupling in static optimization problems for both single-

Work supported by Army Research Office (W911NF-22-2-0242) and NSF (2332210). George Mason University. {zhu3, dshishik, xiao, xwang64}@gmu.edu. Extra experiments of this work appear in [1].

agent [25], [26] or multi-agent [22] cases, there remains a scarcity of studies tailored for multi-agent reinforcement learning (MARL) under CTDE with robots' local observation. This gap motivates our research to develop new approaches that leverage the benefits of bi-level optimization for reinforcement learning applications in multi-robot systems, thereby enhancing their deployment efficiency across a variety of complex and dynamic environments.

Statement of contribution: The contributions of this paper include (i) the formulation of a bi-level Decentralized Markov Decision Process (Dec-MDP) for multi-robot coordination; (ii) a Bi-level Coordination Learning (Bi-CL) framework with a novel alignment mechanism to integrate multi-agent reinforcement learning and imitation learning under the CTDE scheme; (iii) simulated experiments that verify the effectiveness of the proposed Bi-CL algorithm and a comparison with traditional MARL algorithms to solve multi-robot coordination tasks.

Running Example: We conceptualize the proposed bi-level approach using an example, as visualized in Fig. 1. Consider multiple robots traversing an area with multiple adversaries. During their traversal, each robot suffers risk or damage from adversaries accumulated over time, if it enters their impact range. Each robot has two decomposable actions: the action of *move* and the action of *guard*, which are performed simultaneously. The *move* action changes the robot's locations in the environment. In each step, the relative positions of robots to adversaries determine a base risk the robot accumulates. The *guard* is an additional action each robot can choose to perform against an adversary, which increases its own risk but reduces the risks that this adversary poses to other robots. Our goal is to minimize the accumulated risk and traveling time for all robots to arrive at a target position, given each robot only has local (partial) observation of the system.

The same scenario discussed above applies to a handful of applications for multi-robot coordination, such as firefighting considering fire as adversaries; battlefields considering enemy robots as adversaries; and team car racing or football games considering non-cooperative players as adversaries.

II. Preliminaries and Problem Formulation

A. Formulation of a Bi-level Optimization

A regular centralized Markov Decision Process (MDP) can be defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, R)$, including state, action, state transition, discount factor, and reward. The goal is to learn a policy $\pi : \mathcal{S} \to \mathcal{A}$ to maximize the expected cumulative reward over the task horizon T, i.e.,

$$\max_{\pi} \quad \mathop{\mathbb{E}}_{\boldsymbol{a}_t \sim \pi(\cdot|\boldsymbol{s}_t)} \left[\sum_{t=0}^{T} \gamma^t R_t \right], \tag{1}$$

where $\boldsymbol{a}_t \in \mathcal{A}$ and $\boldsymbol{s}_t \in \mathcal{S}$ are the action and state of the system at each step.

Let's assume the actions of the system can be decomposed as $\boldsymbol{a}_t = \{\boldsymbol{x}_t, \boldsymbol{y}_t\} \in \mathcal{X} \times \mathcal{Y}$, and the system's state transition depends only on action $\boldsymbol{x}_t$ such that $\mathcal{T} : \mathcal{S} \times \mathcal{X} \to \mathcal{S}$:

$$\boldsymbol{s}_{t+1} = \mathcal{T}(\boldsymbol{s}_t, \boldsymbol{x}_t). \tag{2}$$

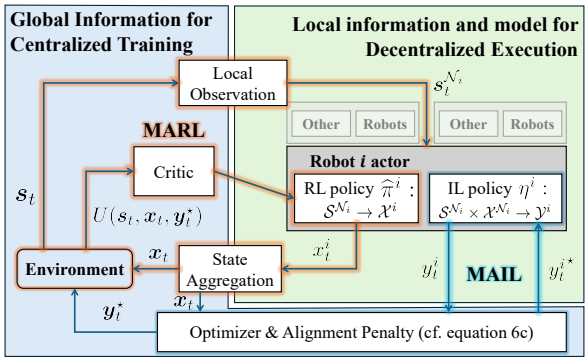

Fig. 2: A Bi-level Coordination Learning (Bi-CL) Algorithm: incorporating multi-agent reinforcement learning (MARL) and imitation learning (MAIL), guided by a global optimizer.

However, the reward function depends on both actions:

$$R_t = U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t). \tag{3}$$

If the structure of problem (1) satisfies (2) and (3), it can be formulated into a bi-level problem:

$$\max_{\widehat{\pi}} \quad \mathop{\mathbb{E}}_{\boldsymbol{x}_t \sim \widehat{\pi}(\cdot|\boldsymbol{s}_t)} \left[\sum_{t=0}^{T} \gamma^t U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t^*(\boldsymbol{s}_t, \boldsymbol{x}_t)) \right], \tag{4a}$$

$$\text{s.t.} \quad \boldsymbol{y}_t^*(\boldsymbol{s}_t, \boldsymbol{x}_t) = \arg\max_{\boldsymbol{y}_t} U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t). \tag{4b}$$

In this paper, we make the following assumption:

Assumption 1. *Problem* (4a) *is complex such that $\widehat{\pi}$ is solved using a reinforcement learning (e.g. actor-critic method), whereas problem* (4b) *can be solved explicitly and be described as a mapping $f : \mathcal{S} \times \mathcal{X} \to \mathcal{Y}$.*

The assumption can be justified by the fact that (4a) is a planning problem involving state transitions and rewards for future T steps, while (4b) only involves a one-step reward.

Under Assumption 1, for the lower level problem (4b), the optimal $\boldsymbol{y}_t^*$ is solvable from $f : \mathcal{S} \times \mathcal{X} \to \mathcal{Y}$, based on the current $\boldsymbol{s}_t$ and the choice of $\boldsymbol{x}_t$. Bringing this solution back to the upper level (4a), we only need to solve a reduced-dimension MDP defined by $(\mathcal{S}, \mathcal{X}, \mathcal{T}, \gamma, R)$, to obtain a policy $\widehat{\pi} : \mathcal{S} \to \mathcal{X}$ that generates action $\boldsymbol{x}_t$. During implementation, the policy π in problem (1) can be a composition of $\pi = \widehat{\pi} \circ f$. We learn the policy $\widehat{\pi}$ to generate $\boldsymbol{x}_t$ from $\boldsymbol{s}_t$, then use f to generate $\boldsymbol{y}_t$ from $\{\boldsymbol{s}_t, \boldsymbol{x}_t\}$.

B. Bi-level Formulation for Multi-robot Coordination Learning with Local Robot Observation

In this paper, we generalize the bi-level formulation into a multi-robot system with local observation. Consider a multi-robot system with a set of $\mathbf{n} = \{1, \cdots, n\}$ robots. Suppose the centralized actions and states in (1) are decomposed into each robot's state and action as $\boldsymbol{s}_t = \{s_t^1, \ldots, s_t^n\} \in \mathcal{S}$ and $\boldsymbol{a}_t = \{a_t^1, \ldots, a_t^n\} \in \mathcal{A}$. Assume each robot's action can be further written as $a_t^i = \{x_t^i, y_t^i\}$, where the action x_t^i determines the robot's local state update $s_{t+1}^i = \mathcal{T}^i(s_t^i, x_t^i)$. The multi-robot system shares a global reward determined by all robots' states, and their two actions: $R_t = U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t)$.

For many real-world applications, the deployment of a multi-robot system faces communication and sensing constraints, where each agent may only have access to the states of its neighbors through a network, denoted by set $\mathcal{N}_i$. The neighbors' states are a subset of the full system state, denoted by $s_t^{\mathcal{N}_i} \subsetneq \boldsymbol{s}_t$. Such local observation turns the problem into a Dec-MDP. Under this constraint, the **problem of interest** in this paper is to learn a *local* policy π^i for each agent such that:

$$\max_{\pi^i} \quad \mathop{\mathbb{E}}_{a_t^i \sim \pi^i(\cdot|s_t^{\mathcal{N}_i})} \left[\sum_{t=0}^{T} \gamma^t R_t \right]. \tag{5}$$

While solving Dec-MDP problems is known to be difficult [27], in this paper, we will generalize the idea of bi-level optimization under the scheme of CTDE and use action space reduction to alleviate the challenges in high-dimensional parameter tuning for MARL. Note that such a generalization is non-trivial and the main challenge lies in the local observation and local decision-making of each robot using incomplete information. As a consequence, compared with the centralized case, the robots cannot solve an optimization problem like (4b) to determine their secondary action y_t^i.

III. Main Approach

As described in Sec. II-A, the centralized bi-level formulation leverages the optimizer (4b) to reduce the dimensionality of the action space, thereby enhancing training efficiency. However, such optimization requires global information, which is not accessible to robots during decentralized execution. Thus, robots cannot locally generate y_t^{i*}. To address this issue, in this section, we introduce a new reformulation to generalize the concept of bi-level optimization to a CTDE setup.

To present our approach, we reformulate (5) into the following Bi-level Coordination Learning (Bi-CL) problem, $\forall i \in \{1, \cdots, n\}$,

$$\max_{\widehat{\pi}^i} \quad \mathop{\mathbb{E}}_{x_t^i \sim \widehat{\pi}^i(\cdot|s_t^{\mathcal{N}_i})} \left[\sum_{t=0}^{T} \gamma^t U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t^\star(\boldsymbol{s}_t, \boldsymbol{x}_t)) \right], \tag{6a}$$

$$\min_{\eta^i} \quad \mathop{\mathbb{E}}_{y_t^i \sim \eta^i(\cdot|s_t^{\mathcal{N}_i}, x_t^{\mathcal{N}_i})} \|y_t^i - y_t^{i*}(\boldsymbol{s}_t, \boldsymbol{x}_t)\|^2, \tag{6b}$$

$$\text{s.t. } \boldsymbol{y}_t^\star(\boldsymbol{s}_t, \boldsymbol{x}_t) = \arg\max_{\boldsymbol{y}_t} \left[U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) \underbrace{- c_k \sum_{i=1}^{n} \mathcal{H}_i(y_t^i, \eta^i))}_{\text{alignment penalty}} \right] \tag{6c}$$

In equations (6a) and (6b), we aim to train two local policies for each robot: one policy, $\widehat{\pi}^i : \mathcal{S}^{\mathcal{N}_i} \to \mathcal{X}^i$, generates action x_t^i; another policy, $\eta^i : \mathcal{S}^{\mathcal{N}_i} \times \mathcal{X}^{\mathcal{N}_i} \to \mathcal{Y}^i$, generates action y_t^i. Note that the two policies are coupled since their inputs rely on the outputs of each other. If training two policies simultaneously, such coupling can pose a stability issue and reduce training efficiency. To address this, our idea is to introduce a global optimizer (6c) under the CTDE scheme to decouple and guide the training of both policies, as we explain next and visualized in Fig. 2.

Algorithm 1: The Bi-CL Algorithm

Initialize local RL models $\widehat{\pi}^i(s^{\mathcal{N}_i}|\theta_{\widehat{\pi}^i})$ and IL models $\eta^i(s^{\mathcal{N}_i}, x^{\mathcal{N}_i}|\theta_{\eta^i})$ for all robots. Initialize centralized critic $Q(\boldsymbol{s}, \boldsymbol{x}|\theta_Q)$ if needed;
Initialize alignment penalty co-efficient c_k;
for $k = 0$ **to** T_k **do**
  Update coefficient $c_k = \frac{c}{1+e^{-\beta(k-h)}}$;
  for $t = 0$ **to** T_t **do**
    for $i = 1$ **to** n **do**
      Get action: x_t^i by adding random perturbation to $\widehat{\pi}^i(s_t^{\mathcal{N}_i}|\theta_{\widehat{\pi}^i})$;
      Get action: $\widehat{y}_t^i = \eta^i(s_t^{\mathcal{N}_i}, x_t^i|\theta_{\eta^i})$;
      State update: $s_{t+1}^i = \mathcal{T}^i(s_t^i, x_t^i)$;
    end
    Solve $\arg\max_{\boldsymbol{y}_t}[U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) - c_k \sum_{i=1}^n \mathcal{H}_i]$ to obtain $\boldsymbol{y}_t^\star$;
    Observe reward $R_t = U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t^\star)$;
    Record $(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t^\star, R_t, \boldsymbol{s}_{t+1})$ to a fixed size buffer $\mathcal{B}$ with first-in-first-out;
    Sample $(\boldsymbol{s}_\tau, \boldsymbol{x}_\tau, \boldsymbol{y}_\tau^\star, r_\tau, \boldsymbol{s}_{\tau+1})$ from $\mathcal{B}$ as a random minibatch of size w;
    Update critic network θ_Q using the minibatch, if critic exist;
    for $i = 1$ **to** n **do**
      Update IL learning model θ_{η^i} using the minibatch;
      Update RL learning model: $\theta_{\widehat{\pi}^i}$ using the minibatch;
    end
  end
end

Bi-level optimization: First, we introduce optimization problem (6c) to solve $\boldsymbol{y}_t^\star$. If temporarily ignoring the alignment penalty term, the equations (6a) and (6c) together represent a CTDE version of the bi-level problem (4). During the centralized training process, we can utilize global information for all robots' states $\boldsymbol{s}_t$ and actions $\boldsymbol{x}_t$ to solve $\boldsymbol{y}_t^\star$. Based on $\boldsymbol{y}_t^\star(\boldsymbol{s}_t, \boldsymbol{x}_t)$, we train the local policy $\widehat{\pi}^i$ in (6a) for each agent's x_t^i using a *reinforcement learning* scheme. $\widehat{\pi}^i$ has a *reduced-dimension* action space because $\boldsymbol{y}_t^\star$ is not learned.

On the other hand, while optimizer (6c) can be used in centralized training to obtain action $\boldsymbol{y}_t^\star$ for all robots, it cannot be used by robots during decentralized execution due to the local observations. Thus, we introduce (6b) to train a local policy η^i to generate agent's y_t^i. The policy η^i can be trained conveniently using an imitation learning (IL) scheme that uses optimizer (6c) as the *expert demonstration*. Here, the imitation loss in (6b) measures the difference between the policy output y_t^i using local information and the optimal action solved from optimizer (6c) using global information.

Alignment of the two policies: We explain the alignment term in (6c). Based on the above explanation, the policy $\widehat{\pi}^i$

is trained assuming the actions $\boldsymbol{y}_t^\star = \{y_t^{1*}, \cdots, y_t^{n*}\}$ are optimal for all robots, and the policy η^i seeks to generate such optimal actions using only local information. Clearly, during decentralized execution, the optimality of η^i can hardly be guaranteed, thus creating a mismatch between the policies $\widehat{\pi}^i$ and η^i. To address this, we introduce an alignment penalty term $c_k \sum_{i=1}^n \mathcal{H}_i(y_t^i, \eta^i))$ in (6c), where $c_k > 0$ and

$$\mathcal{H}_i = \mathop{\mathbb{E}}_{\widehat{y}_t^i \sim \eta^i(\cdot|s_t^{\mathcal{N}_i}, x_t^{\mathcal{N}_i})} \|y_t^i - \widehat{y}_t^i\|^2.$$

Here, $\widehat{y}_t^i$ is the output of the model η^i, then $\mathcal{H}_i$ evaluates how well an action y_t^i aligns with the policy η^i. Minimizing this penalty helps to reduce the mismatch between the two local policies. Mathematically, the value of $\mathcal{H}_i$ for each robot equals the loss in (6b). Hence, c_k is an important coefficient and we shall remark on its choice during the training. If $c_k \to 0$, the reformulation (6) disregards the policy mismatch, always opting for the solution that maximizes U to train $\widehat{\pi}^i$. Conversely, when $c_k \to \infty$, the penalty forces the training of $\widehat{\pi}^i$ using the action chosen by η^i, making the two policies fully coupled, and the IL (6b) no longer updates. To strike a balance, our idea is to make sure the IL model is sufficiently trained before the penalty is applied. For this purpose, we define a modified logistic function:

$$c_k = \frac{c}{1 + e^{-\beta(k-h)}} \quad \text{with} \quad c > 0, \beta > 0, h > 0 \qquad (7)$$

where k is the training episode. We let c_k start with zero to 'jump start' the training of $\widehat{\pi}^i$ and η^i using the optimal $\boldsymbol{y}_t^\star$ for maximizing U. Afterwards, c_k is gradually increased to a sufficiently large value c to fine-tune $\widehat{\pi}^i$, taking into account the output of η^i and ensuring alignment between the two policies, $\widehat{\pi}^i$ and η^i. The described training scheme is summarized in Algorithm 1.

IV. Numerical Results

In this section, we employ the running example to verify the proposed algorithm. We use simulated experiments to demonstrate: (i) the effectiveness of the proposed bi-level learning schemes, in particular, the alignment penalty term; (ii) the advantage of the proposed approach in terms of training efficiency compared with alternative MARL algorithms.

Through this section, we use the accumulated reward (5) to evaluate the proposed Bi-CL algorithm. To avoid ambiguity, we distinguish the following metrics:

RL-Reward: refers to the reward of the reinforcement learning policy $\widehat{\pi}^i$ when solving (6a) during centralized training.

T-Reward: refers to an average reward obtained by executing both the learned reinforcement learning policy $\widehat{\pi}^i$ and imitation learning policy η_i in the environment for thirty times.

R-Gap: refers to the subtraction between the RL-Reward and the T-Reward.

Remark 1. *Note that RL-Reward and T-Reward are different because they use different ways to choose robots' action y_t^i. RL-Reward uses an 'optimal' solution $\boldsymbol{y}_t^\star = \{y_t^{1\star}, \cdots, y_t^{n\star}\}$ of* (6c)*; T-Reward uses output of the learned policy η_i. Only the T-Reward reflects the true performance of the policies because it is achievable during decentralized execution, and higher means better. Since R-Gap evaluates the difference, smaller means better.*

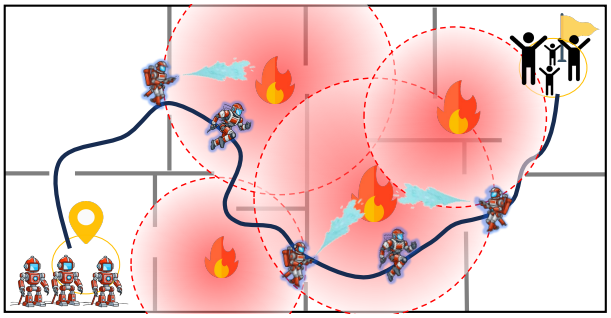

Fig. 3: Running Example (a): all robots travel along a route.

Running Example: We introduce a running example for coordinated multi-robot route traversal, by assuming all robots travel continuously through a route. Each robot can only observe a subset of full system states through an underlying communication network. Fig. 3 provides a visualization of the environment. The testing environment we use is a mathematical abstraction that may not exactly follow its physical layout and the number of adversaries may change.

- Each robot i has a continuous ***move action*** (velocity) $x_t^i \in [-v_{\max}, v_{\max}]$, and a discrete ***guard action*** $y_t^i \in \mathcal{M}$ where $\mathcal{M}$ is the set of all adversaries. The robot i's ***state transition*** (position) follows dynamics $s_{t+1}^i = s_t^i + x_t^i$. These align with definition (2).
- Suppose adversary $j \in \mathcal{M}$ has an impacted area $\mathcal{B}_j$. Each time-step, if $s_t^i \in \mathcal{B}_j$, robot i accumulates a cost $c_j(s_t^i)$. Besides, for any robot k, if $s_t^k \in \mathcal{B}_j$, it can perform guard against adversary j, i.e., $y_t^k = j$. The guarding effect is characterized by a discount factor on the costs created by adversary j:

$$\alpha_t^{k,j}(x_t^k, y_t^k) = \begin{cases} 1 - \beta \frac{(v_{\max} - x_t^k)}{v_{\max}} & y_t^k = j, \\ 1 & \text{otherwise.} \end{cases}$$

Such a discount is more effective when the robot admits at a lower moving velocity x_t^i. The total ***team cost*** in each step is defined as:

$$R_t = U(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) = -\sum_{i=1}^n \sum_{j=1}^m \left[\prod_{k=1}^n \alpha_t^{k,j}(x_t^k, y_t^k) c_t^{i,j}(s_t^i) \right] - \delta,$$

where δ is a constant time penalty. A one-time positive reward is added when all robots arrive at the target. The definition of U aligns with definition (3).

- The goal is to minimize the team accumulated cost in the form of (5) before all robots arrive at the target position.

The alignment of the problem with (2) and (3) allows it to be reformulated into (6) and then solved by our Bi-CL.

Effectiveness of Alignment Penalty: We first implement the proposed Bi-CL in an environment with 4 robots and 4 adversaries (fire). In Bi-CL, the reinforcement learning of $\widehat{\pi}^i$ uses an independent actor in each robot with centralized critic [28, Sec. 2.4], following the structure of MADDPG [4]

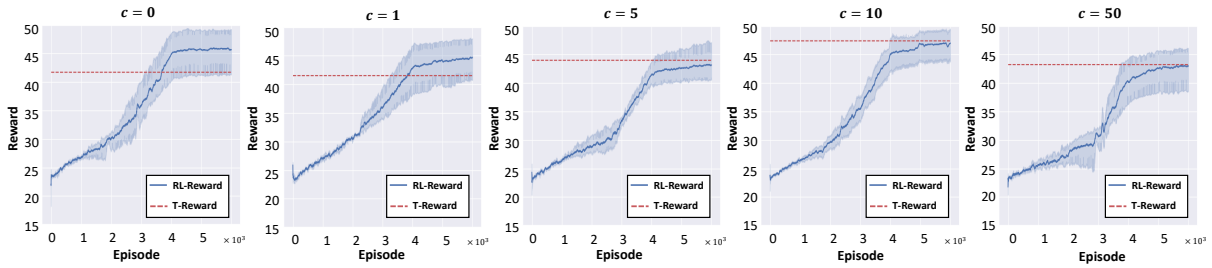

Fig. 4: Comparison of cumulative reward for different alignment penalties with four robots and four adversaries. The height of red dash lines determines implementation performance.

but over reduced action space. Each robot only observes part of the global information, i.e., its own state and the states of a subset of other robots in the system The cost function c_j is an affine function that depends on the closeness between the robot and the center of the adversary.

Fig. 4 illustrates the RL-Reward curves and T-Reward for different c_t by setting $c = 0, c = 1, c = 5, c = 10, c = 50$, all with $\beta = 2e - 3$ and $h = 3000$. This, together with (7), makes c_k gradually increase in an 'S' shape from 0 to c. Upon examining the results, it is evident that the R-Gap is large when c is small, and the gap diminishes as c increases. This verifies the effectiveness of the proposed alignment penalty term in (6c), as it motivates the policy $\widehat{\pi}_t^i$ to be tuned using a $y_t^{i\star}$ that is closer to the output of η_t^i. It is also important to note that although the MARL training reward appears high when c is small, this is misleading and unattainable in decentralized execution due to the mismatch between $\widehat{\pi}_t^i$ and η_t^i. The attainable reward (T-Reward) is much lower with smaller c.

Finally, we observe in the last plot of Fig. 4 that a very large $c = 50$ may negatively impact the training performance. This is due to two reasons. First, as we discussed in the algorithm development, larger c_k reduces the training efficiency of the imitation learning part of the algorithm. This is reflected by the lower values on both RL-Reward and T-Reward. Second, since c_k impacts the computation of $y_t^{i\star}$ during the training process, changing it too aggressively will lead to instability in training. This is evidenced by the middle part of the curve, where increased osculation is observed.

The same test is performed under various environment setups and the results are shown in Table I. Here, *Na_Mb* means

TABLE I: Average reward per episode of different c_k values

Environments	Metric	$c_k = 0$	$\beta = 2e^{-3}$			
			$c = 1$	$c = 5$	$\boldsymbol{c = 10}$	$c = 50$
N3_M3	T-Reward	51.83	53.01	**53.44**	53.17	52.13
	R-Gap	1.34	0.73	0.26	-0.45	**-0.23**
N4_M4	T-Reward	42.10	42.03	44.05	**47.38**	43.56
	R-Gap	3.98	1.55	-0.87	-0.39	**-0.11**
N5_M4	T-Reward	56.31	58.27	59.68	**60.08**	59.46
	R-Gap	4.79	4.45	1.17	**-0.04**	-0.15
*N5_M4**	T-Reward	50.67	53.39	54.09	**54.60**	54.20
	R-Gap	6.54	3.84	1.82	-0.59	**0.27**

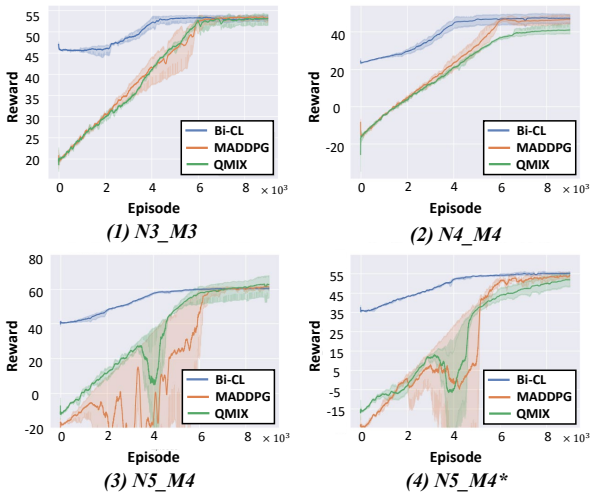

Fig. 5: Performance Comparisons in Different Scenarios.

the environment has a robots and b adversaries. Especially, in *N5_M4**, each robot further reduces the number of other robots's states it can observe. The results in the table align with our above analysis. In all cases where $c_k = c = 0$, i.e., without alignment penalty, the performance is the worst. The R-Gap generally reduces when c grows large. Since T-Reward is the core metric for performance, the best result mostly occurs at $c = 10$ with only one exception and the difference is small. Furthermore, to read the table column-wise, environments with the same number of adversaries, i.e., *M4* are comparable. From *N4_M4* to *N5_M4*, the T-Rewards generally increase because more coordination behaviors can be generated. When comparing *N5_M4** and *N5_M4*, the T-Rewards generally decrease due to the reduced sensing distances of the robots. This also leads to a larger R-Gap when $c = 0$, necessitating the introduced alignment penalty.

Comparing Training Efficiency with Baselines: We also compare the proposed Bi-CL algorithm with two well-established MARL algorithms, MADDPG [4] and QMIX [7], respectively. MADDPG and QMIX are implemented in complete action space $\mathcal{A}^i = \mathcal{X}^i \times \mathcal{Y}^i$ of each robot. The same learning rates with Bi-CL are used. The comparison is visualized in Fig. 5 for the four cases in the above discussion. Here, we choose $c = 10$ as the R-Gaps are small. Thus, the RL-Reward curve can represent the convergence of our algorithm and can approximate the true reward our algorithm can achieve. It can be observed that in all cases, Bi-CL can

achieve similar final rewards compared with baselines, which justifies the effectiveness of the proposed algorithm. The efficiency of Bi-CL is evidenced by its faster convergence speed compared with baselines. Furthermore, the starting reward of our algorithm is significantly better because it uses the optimization (6c) to boost and guide the policy training, while other methods are purely based on exploration. For a similar reason, we observe that both MADDPG and QMIX suffer from training stability issues for complex scenarios, while the proposed Bi-CL does not.

V. Conclusion

We presented a bi-level formulation for multi-robot coordination learning with local observation, wherein robots' state transitions and their cooperative behaviors are abstracted and optimized on different levels, significantly enhancing learning efficiency. A key enabler of our Bi-CL algorithm was an alignment penalty that enables upper-level learning to account for potential discrepancies arising from local observations in lower-level optimization. We validated our algorithm through a running example. Experimental results demonstrated that our algorithm can effectively learn in the environment. We evaluated the performance enhancement of the Bi-CL using different alignment penalty parameters. Comparative analysis with baselines verified the efficiency of our algorithm.

For future work, we aim to explore the scalability of our Bi-CL to accommodate larger multi-robot systems and more complex environments, further refining the alignment penalty mechanism to enhance its adaptability and efficiency. Moreover, we intend to extend our way of handling robots' information loss to effectively manage dynamic, stochastic, and noisy scenarios, thereby enhancing its resilience and performance in unpredictably evolving multi-robot coordination environments.

References

[1] Z. Hu, D. Shishika, X. Xiao, and X. Wang, "Bi-cl: A reinforcement learning framework for robots coordination through bi-level optimization," *arXiv preprint arXiv:2404.14649*, 2024.

[2] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, pp. 1–31, 2019.

[3] X. Wang, J. Hudack, and S. Mou, "Distributed algorithm with resilience for multi-agent task allocation," in *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, pp. 112–117, IEEE, 2021.

[4] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[5] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative multi-agent games," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24611–24624, 2022.

[6] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, *et al.*, "Value-decomposition networks for cooperative multi-agent learning," *arXiv preprint arXiv:1706.05296*, 2017.

[7] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[8] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *International Journal of Advanced Robotic Systems*, vol. 10, no. 12, p. 399, 2013.

[9] J. P. Queralta, J. Taipalmaa, B. C. Pullinen, V. K. Sarker, T. N. Gia, H. Tenhunen, M. Gabbouj, J. Raitoharju, and T. Westerlund, "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision," *Ieee Access*, vol. 8, pp. 191617–191643, 2020.

[10] X. Xiao, J. Dufek, and R. R. Murphy, "Autonomous visual assistance for robot operations using a tethered uav," in *Field and Service Robotics: Results of the 12th International Conference*, pp. 15–29, Springer, 2021.

[11] Y. Lian, Q. Yang, Y. Liu, and W. Xie, "A spatio-temporal constrained hierarchical scheduling strategy for multiple warehouse mobile robots under industrial cyber–physical system," *Advanced Engineering Informatics*, vol. 52, p. 101572, 2022.

[12] R. Liu, J. Gao, J. Zhang, D. Meng, and Z. Lin, "Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 10045–10067, 2021.

[13] A. Antoniou, H. Edwards, and A. Storkey, "How to train your maml," in *International Conference on Learning Representations*, 2018.

[14] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," in *International Conference on Learning Representations*, 2018.

[15] D. Pfau and O. Vinyals, "Connecting generative adversarial networks and actor-critic methods," *arXiv preprint arXiv:1610.01945*, 2016.

[16] M. Hong, H. Wai, Z. Wang, and Z. Yang, "A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic. arxiv e-prints, art," *arXiv preprint arXiv:2007.05170*, 2020.

[17] X. Wang, S. Mou, and B. D. Anderson, "Consensus-based distributed optimization enhanced by integral feedback," *IEEE Transactions on Automatic Control*, vol. 68, no. 3, pp. 1894–1901, 2022.

[18] A. Sinha, P. Malo, and K. Deb, "A review on bilevel optimization: From classical to evolutionary approaches and applications," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 276–295, 2017.

[19] Y. Mintz, J. A. Cabrera, J. R. Pedrasa, and A. Aswani, "Control synthesis for bilevel linear model predictive control," in *2018 Annual American Control Conference (ACC)*, pp. 2338–2343, IEEE, 2018.

[20] T. Nisha, D. T. Nguyen, and V. K. Bhargava, "A bilevel programming framework for joint edge resource management and pricing," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 17280–17291, 2022.

[21] X. Wang, S. Mou, and B. D. Anderson, "Scalable, distributed algorithms for solving linear equations via double-layered networks," *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 1132–1143, 2019.

[22] H. Zhang, W. Chen, Z. Huang, M. Li, Y. Yang, W. Zhang, and J. Wang, "Bi-level actor-critic for multi-agent coordination," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 7325–7332, 2020.

[23] L. Zheng, T. Fiez, Z. Alumbaugh, B. Chasnov, and L. J. Ratliff, "Stackelberg actor-critic: Game-theoretic reinforcement learning algorithms," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, pp. 9217–9224, 2022.

[24] H. M. Le, Y. Yue, P. Carr, and P. Lucey, "Coordinated multi-agent imitation learning," in *International Conference on Machine Learning*, pp. 1995–2003, PMLR, 2017.

[25] A. Biswas and C. Hoyle, "A literature review: solving constrained non-linear bi-level optimization problems with classical methods," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 59193, p. V02BT03A025, American Society of Mechanical Engineers, 2019.

[26] M. Limbu, Z. Hu, S. Oughourli, X. Wang, X. Xiao, and D. Shishika, "Team coordination on graphs with state-dependent edge costs," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 679–684, IEEE, 2023.

[27] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of markov decision processes," *Mathematics of operations research*, vol. 27, no. 4, pp. 819–840, 2002.

[28] Y. Xiao, W. Tan, and C. Amato, "Asynchronous actor-critic for multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4385–4400, 2022.