Multi-Agent Trust Region Policy Optimisation: A Joint Constraint Approach

AAAI Track

Chak Lam Shek
University of Maryland, College Park
College Park, United States of America
cshek1@umd.edu

Guangyao Shi
University of Southern California
LA, United States of America
shig@usc.edu

Pratap Tokekar
University of Maryland, College Park
College Park, United States of America
tokekar@umd.edu

ABSTRACT

Multi-agent reinforcement learning (MARL) requires coordinated and stable policy updates among interacting agents. Heterogeneous-Agent Trust Region Policy Optimization (HATRPO) enforces per-agent trust region constraints using Kullback–Leibler (KL) divergence to stabilize training. However, assigning each agent the same KL threshold can lead to slow and locally optimal updates, especially in heterogeneous settings. To address this limitation, we propose two approaches for allocating the KL divergence threshold across agents: HATRPO-W, a Karush–Kuhn–Tucker-based (KKT-based) method that optimizes threshold assignment under global KL constraints, and HATRPO-G, a greedy algorithm that prioritizes agents based on improvement-to-divergence ratio. By connecting sequential policy optimization with constrained threshold scheduling, our approach enables more flexible and effective learning in heterogeneous-agent settings. Experimental results demonstrate that our methods significantly boost the performance of HATRPO, achieving faster convergence and higher final rewards across diverse MARL benchmarks. Specifically, HATRPO-W and HATRPO-G achieve comparable improvements in final performance, each exceeding 22.5%. Notably, HATRPO-W also demonstrates more stable learning dynamics, as reflected by its lower variance.

KEYWORDS

Multi-agent Learning; Reinforcement learning; Multi-agent Reinforcement Learning

ACM Reference Format:
Chak Lam Shek, Guangyao Shi, and Pratap Tokekar. 2026. Multi-Agent Trust Region Policy Optimisation: A Joint Constraint Approach: AAAI Track. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/ZOYC2112

1 INTRODUCTION

Multi-Agent Reinforcement Learning (MARL) is a critical area of artificial intelligence whose goal is to enable multiple agents to make decisions and coordinate within shared environments [2, 21]. Its versatility has driven impactful applications in domains such as robotics [6, 28], autonomous driving [1, 25], and smart grid management [14, 17], where effective coordination and adaptability are crucial. More recently, MARL techniques have been leveraged to enhance Large Language Models, exemplified by methods such as Group Relative Policy Optimization (GRPO) for coordinating multiple reasoning agents during text generation [20]. These advancements underscore the importance of developing efficient and scalable MARL algorithms to propel progress in both traditional control systems and emerging foundation models.

In single-agent reinforcement learning, foundational methods such as Deep Q-Learning [13] have laid the groundwork for effective policy learning in high-dimensional environments. Trust-region-based methods such as Trust Region Policy Optimization (TRPO) [18] and Proximal Policy Optimization (PPO) [19] have demonstrated strong empirical performance, particularly in continuous control tasks. These methods have become the canonical approaches for RL. Inspired by these successes, researchers have extended these algorithms to the multi-agent setting, leading to variants such as Independent PPO (IPPO) [3] and Multi-Agent PPO (MAPPO) [26]. Although effective, these methods often suffer from gradient interference when multiple agents update their policies simultaneously, which can result in convergence to suboptimal local minima.

To address this issue of gradient interference between agents, Kuba et al. [8] proposed Heterogeneous-Agent Trust Region Policy Optimization (HATRPO), which introduces a sequential policy update strategy, allowing agents to update one at a time. This reduces gradient conflicts and improves learning stability in heterogeneous multi-agent environments. Despite the advancements brought by HATRPO, it assumes a uniform KL divergence threshold for all agents, which may not be optimal in heterogeneous settings where agents have varying capacities for improvement. This uniform allocation can limit overall system performance, as some agents might benefit from larger policy updates. To address this limitation, we propose two approaches for allocating the KL divergence threshold among agents: HATRPO-G, a greedy algorithm that prioritizes agents based on their improvement-to-divergence ratio, and HATRPO-W, a method based on Karush-Kuhn-Tucker (KKT) conditions to optimize threshold assignment under KL constraints. These methods aim to enhance the flexibility and effectiveness of policy updates in heterogeneous-agent environments.

Our study investigates whether such adaptive KL divergence threshold allocation can lead to improved performance, more structured learning dynamics, and faster convergence, all while maintaining the same total KL divergence budget. Through this lens, we

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/ZOYC2112

examine whether the learned threshold allocation reflects meaningful distinctions in agent advantage and whether it can overcome the limitations of fixed-threshold designs in coordinated multi-agent learning.

This paper makes the following key contributions:

- We show that uniform KL thresholding across agents is suboptimal in sequential MARL, and propose a Joint KL constraint optimization problem to enable threshold allocation that improves overall coordination.
- We propose two methods for adaptive threshold allocation: a greedy method based on the improvement-to-divergence ratio, and a KKT-based optimization method that allocates the KL threshold under global constraints.
- Through extensive experiments on MARL benchmarks, we show that our adaptive KL allocation methods improve HATRPO's performance over baselines. Our results highlight that prioritizing agents based on their improvement potential leads to faster convergence, more effective policy updates, and better utilization of the KL budget. Specifically, HATRPO-W and HATRPO-G achieve comparable improvements in final performance, each exceeding 22.5%. Notably, HATRPO-W also demonstrates more stable learning dynamics, as reflected by its lower variance.

2 RELATED WORK

Early MARL methods predominantly assumed *homogeneous* agents sharing policy parameters. Weighted QMIX (WQMIX) [15], Independent Q-Learning (IQL) [3], and MADDPG [12] achieve strong performance in discrete cooperative tasks (e.g., SMAC) [16]. While shared-policy approaches simplify cooperation, they inhibit expressive behavior when agents play *heterogeneous* roles. The Heterogeneous Agent Reinforcement Learning (HARL) framework addresses this by assigning each agent its own policy network, combined with centralized critics, sequential updates, and trust-region guarantees [8, 32]. In HARL, HATRPO (Heterogeneous-Agent TRPO) and HAPPO (Heterogeneous-Agent PPO) are two representative algorithms that are tractable and ensure monotonic improvement and joint-return guarantees [8, 32]. The theoretical foundation is strengthened by Heterogeneous-Agent Mirror Learning (HAML) [9], which guarantees both monotonic performance improvement and convergence to Nash equilibria; it also spawns additional agents like HAA2C, HADDPG, and HATD3 [11, 32]. Zhong et al. [32] propose the Multi-Agent Transformer (MAT), framing MARL as a sequential decision process in which a Transformer generates agent actions one-by-one to enhance coordination. Building on trust-region methods, A2PO [23] introduces an agent-by-agent policy optimization scheme that highlights the role of update ordering in achieving superior performance. To promote exploration in heterogeneous environments, MADPO [4] maximizes mutual policy divergence by explicitly distinguishing between intra- and inter-agent differences. FP3O [5] extends PPO to support versatile parameter-sharing structures while preserving monotonic joint policy improvement guarantees. Complementary to these approaches, an optimism-driven gradient method [30] has been developed to improve sample efficiency in cooperative tasks. In parallel, TSPPO [22] leverages Transformer-based sequential modeling within a PPO framework to improve agent coordination. Further enhancing Transformer integration, the Sequence Value Decomposition Transformer [31] aligns agent-specific returns with cooperative objectives. Additionally, Liu et al. [10] explore the role of joint optimization in PPO-based multi-agent learning. Finally, Zhang et al. [29] propose a Stackelberg Decision Transformer to address asynchronous action dependencies through a sequential coordination framework.

In contrast, our work introduces a KL-constrained scheduling mechanism for HATRPO that adapts each agent's policy update magnitude based on its improvement potential. By formulating the update process as a joint KL optimization problem and introducing both greedy and KKT-based threshold allocation strategies, our approach addresses the limitations of uniform KL assignments and enhances both convergence speed and final performance in sequential MARL.

3 PRELIMINARIES

We consider a **Markov game**, defined by the tuple $\mathcal{M} = \langle \mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma \rangle$, where $\mathcal{N} = \{1, \ldots, n\}$ is the set of n agents, $\mathcal{S}$ is the finite state space, and $\mathcal{A} = \prod_{i=1}^{n} \mathcal{A}_i$ is the joint action space with $\mathcal{A}_i$ denoting the action space of agent i. The transition dynamics are governed by the probability function $\mathcal{P}(s' \mid s, \mathbf{a})$, which specifies the likelihood of transitioning to state s' from state s under a joint action $\mathbf{a} \in \mathcal{A}$. The reward function $r(s, \mathbf{a})$ returns a scalar reward for a given state and joint action, and $\gamma \in [0, 1)$ is the discount factor. At each time step t, the agents occupy a state $s_t \in \mathcal{S}$ and each agent $i \in \mathcal{N}$ selects an action $a_t^i \in \mathcal{A}_i$ according to its policy $\pi_i(\cdot \mid s_t)$. These actions form the joint action $\mathbf{a}_t = (a_t^1, \ldots, a_t^n) \in \mathcal{A}$, drawn from the joint policy $\pi(\cdot \mid s_t) = \prod_{i=1}^{n} \pi_i(\cdot \mid s_t)$. The agents then receive a joint reward $r_t = r(s_t, \mathbf{a}_t)$ and transition to the next state s_{t+1}, sampled according to $\mathcal{P}(s_{t+1} \mid s_t, \mathbf{a}_t)$.

The joint policy π, transition function $\mathcal{P}$, and initial state distribution ρ_0 together induce a sequence of marginal state distributions ρ_t^π at each timestep t. The overall discounted state visitation distribution is defined as: $\rho^\pi = \sum_{t=0}^{\infty} \gamma^t \rho_t^\pi$.

The value of a state under policy π is given by the state value function: $V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \,\middle|\, s_0 = s\right]$, and the expected return for executing a joint action $\mathbf{a}$ from state s is captured by $Q^\pi(s, \mathbf{a}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \,\middle|\, s_0 = s, \mathbf{a}_0 = \mathbf{a}\right]$.

The corresponding advantage function, which quantifies the benefit of taking joint action $\mathbf{a}$ over following the policy π, is defined as: $A^\pi(s, \mathbf{a}) = Q^\pi(s, \mathbf{a}) - V^\pi(s)$.

In this setting, all agents share the same reward function. The goal is to maximize the expected total reward: $J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$, where $s_0, s_1, \cdots \sim \rho^\pi$ and $\mathbf{a}_0, \mathbf{a}_1, \cdots \sim \pi$.

We also investigate the contribution to performance from different subsets of agents. Let $i_{1:m}$ denote an ordered subset $\{i_1, \ldots, i_m\}$ of $\mathcal{N}$, and $-i_{1:m}$ refer to its complement. The k-th agent in the subset is written as i_k.

The **multi-agent state-action value function** is defined as: $Q_{i_{1:m}}^\pi(s, \mathbf{a}) = \mathbb{E}_{\mathbf{a}_{-i_{1:m}} \sim \pi_{-i_{1:m}}}\left[Q^\pi(s, \mathbf{a}_{i_{1:m}}, \mathbf{a}_{-i_{1:m}})\right]$.

For disjoint sets $j_{1:k}$ and $i_{1:m}$, the **multi-agent advantage function** is defined as:

$$A_{i_{1:m}}^\pi(s, \mathbf{a}_{j_{1:k}}, \mathbf{a}_{i_{1:m}}) = Q_{j_{1:k}, i_{1:m}}^\pi(s, \mathbf{a}_{j_{1:k}}, \mathbf{a}_{i_{1:m}}) - Q_{j_{1:k}}^\pi(s, \mathbf{a}_{j_{1:k}}).$$

Here, the joint policies $\pi = (\pi_1, \ldots, \pi_n)$ and $\bar{\pi} = (\bar{\pi}_1, \ldots, \bar{\pi}_n)$ represent the "current" and "new" joint policies, respectively, as agents update their strategies.

3.1 Heterogeneous-Agent TRPO Algorithm

The *Heterogeneous Agents Trust Region Proximal Optimization* (HATRPO) algorithm [32] extends Trust Region Policy Optimization (TRPO) to heterogeneous multi-agent systems. Unlike homogeneous agent methods that assume shared observation and action spaces or identical policy architectures, HATRPO accommodates agents with distinct modalities and parameterizations. A key feature of HATRPO is that it preserves the monotonic improvement guarantee at the *joint policy level*, meaning that sequential updates of individual agents still lead to consistent improvement of the overall system performance. Each agent solves a constrained optimization problem to improve its local policy while ensuring bounded divergence from the previous policy.

The policy update for agent $i_m \in i_{1:m}$ at iteration $k+1$ is given by the following constrained optimization:

$$\pi^{i_m}_{k+1} = \arg\max_{\pi^{i_m}} \mathbb{E}_{s\sim\rho_\pi,\, a^{i_{1:m-1}}\sim\pi^{i_{1:m-1}},\, a^{i_m}\sim\pi^{i_m}} \left[A^{i_m}_\pi(s, \mathbf{a}^{i_{1:m-1}}, a^{i_m})\right]$$
$$\text{s.t. } \mathbb{E}_{s\sim\rho_\pi}\left[D_{\mathrm{KL}}\left(\pi^{i_m}_k(\cdot \mid s) \,\|\, \pi^{i_m}(\cdot \mid s)\right)\right] \le \delta \tag{1}$$

The surrogate objective used in this optimization is defined as:

$$L^{i_{1:m}}_\pi(\bar{\pi}^{i_{1:m-1}}, \hat{\pi}^{i_m}) \triangleq \mathbb{E}_{s\sim\rho_\pi,\, a^{i_{1:m-1}}\sim\bar{\pi}^{i_{1:m-1}},\, a^{i_m}\sim\hat{\pi}^{i_m}} \left[A^{i_m}_\pi(s, \mathbf{a}^{i_{1:m-1}}, a^{i_m})\right] \tag{2}$$

By leveraging centralized training with decentralized execution, HATRPO facilitates efficient, stable learning in multi-agent systems composed of heterogeneous agents while maintaining strong performance guarantees.

4 JOINTLY CONSTRAINED HATRPO

While HATRPO offers theoretical guarantees by enforcing trust region constraints for stable policy updates, it applies a uniform KL divergence threshold to each agent individually:

$$D^{\max}_{\mathrm{KL}}(\pi_i \,\|\, \bar{\pi}_i) \le \delta, \quad \forall i = 1, \ldots, m \tag{3}$$

A small KL divergence threshold is often preferred in trust-region methods because it provides a tighter lower bound on policy performance improvement. By limiting how far a new policy can deviate from the old one, small δ values ensure stable and theoretically grounded learning steps.

However, in HATRPO, assigning the same small δ to every agent can inadvertently slow down the overall learning process. Specifically, agents with low advantage values or those operating in flat regions of the optimization landscape may receive updates that are minimal or unproductive. This restricts their ability to escape poor local optima and contributes to inefficient use of the policy update threshold.

This limitation is clearly illustrated in Figure 1, which depicts a two-player differential matrix game where the reward function consists of a combination of two Gaussian modes: one corresponding to a local optimum and the other to a global optimum with broader spread in the a_2 (vertical) direction. The policies are initialized as Gaussians with means at $(a_1, a_2) = (1, 1)$, placing them near the local optimum. Under HATRPO, both players are constrained by the same KL threshold, which prevents sufficient exploration, especially by player 1, and leads the joint policy to remain trapped near the local optimum.

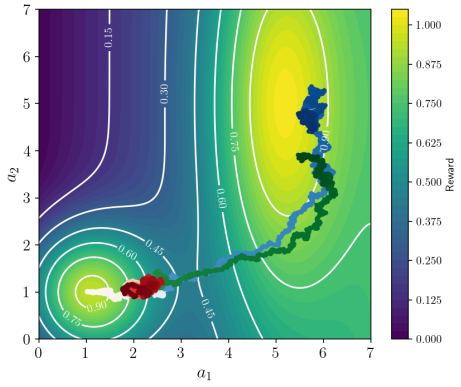


Figure 1: Heatmap of the reward function $R(a_1, a_2) = 10\,\mathcal{N}(5, 5;\, 1, 3) + 0.1a_1 + 5.3\,\mathcal{N}(1, 1;\, 1, 1)$, where $\mathcal{N}(\mu_1, \mu_2;\, \sigma_1, \sigma_2)$ denotes a 2D Gaussian with mean (μ_1, μ_2) and standard deviation (σ_1, σ_2). The plot shows the joint reward surface and the action trajectories of two players optimizing their actions a_1 and a_2. Color gradients indicate optimization progress: red for HATRPO, blue for HATRPO-G, and green for HATRPO-W.

In contrast, HATRPO-G and HATRPO-W dynamically adjust the KL threshold allocation, allowing player 1 to receive a larger update allowance. Since the global optimum has low curvature in the a_2 direction, increasing the KL threshold for player 1 allows the joint policy to escape the local optimum and converge to the global maximum.

Moreover, since each agent is constrained independently, the uniform allocation does not account for varying contributions to global performance. As a result, the overall system improvement may be suboptimal, especially in heterogeneous settings where agents differ in learning potential or strategic significance.

This constraint also directly affects the speed of convergence. To illustrate, consider a multi-agent matrix game where each of the N agents simultaneously selects an action $a_i \in \{0, 1\}$. The joint reward is defined as follows:

$$R(\mathbf{a}) = \begin{cases} 1.5 & \text{if } a_i = 1 \text{ for all } i, \\ 1 & \text{if } \exists j \text{ s.t. } a_j = 0 \text{ and } a_k = 0\ \forall k > j, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

In this setting, the optimal reward (1.5) is only achieved when all agents simultaneously select 1. However, the reward function is inherently non-symmetric. Low-index agents are more likely to receive positive advantage estimates because a later agent can only obtain reward if all earlier agents choose action 1. As a result, during the early learning phase, low-index agents tend to accumulate positive gradients and request substantial policy updates, while high-index agents experience little or no improvement signal and remain relatively static.

This imbalance leads to a mismatch between policy improvement demand and update capacity when a fixed KL threshold is distributed uniformly. Consequently, the overall learning progress is bottlenecked by high-index agents that adapt slowly, which hinders the system's ability to coordinate toward the optimal joint policy. This phenomenon is reflected in the learning curve behavior shown in Figure 2.

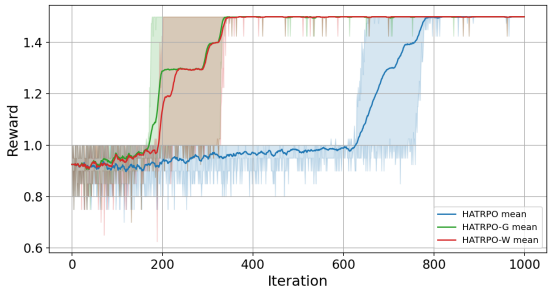


Figure 2: Average reward as a function of training iterations in the 4-agent matrix game.

Algorithm 1: Modified HATRPO with Hard KL Constraint and KL Divergence Threshold Allocation

Require: Initial joint policy $\pi_0 = (\pi_0^1, \ldots, \pi_0^m)$

for $k = 0, 1, 2, \ldots$ **do**
  Compute the joint advantage function $A^{\pi_k}(s, a)$ for all state-action pairs
  Compute $Order(i_{1:m})$, $\{\delta_1, ..., \delta_m\}$ using KL allocation algorithm 2 or 3
  // Solve the following constrained sequential optimization problem 1 with δ_i
end for

These observations motivate the need for a more flexible allocation of policy update thresholds, which can accelerate convergence and enable coordinated breakthroughs in complex multi-agent interactions.

4.1 Problem Formulation

To address this, we reformulate the constraint structure while retaining the same objective. We continue to maximize the total advantage across all agents:

$$\max_{\pi_1,\ldots,\pi_m} \sum_{i=1}^{m} \mathcal{L}_i(\bar{\pi}_{1:i-1}, \pi_i) \tag{5}$$

But instead of enforcing per-agent KL constraints, we propose a global trust region:

$$\sum_{i=1}^{m} D_{\mathrm{KL}}^{\max}(\pi_i \,\|\, \bar{\pi}_i) \le \delta_{\mathrm{total}} \tag{6}$$

This formulation allows agents to share a common KL threshold and redistribute improvement potential more effectively—allocating more threshold to high-impact agents and reducing waste. Agents with larger advantage functions and smaller policy changes are more desirable for updates. Additionally, since KL divergence is not symmetric and policy order affects downstream value changes, update order matters [24]. This prioritization enables better group-level coordination and maximizes collective reward.

Algorithm 2: Greedy Agent Ordering via Score-Based Ranking

Require: Fixed priors $\bar{\pi}$, total KL threshold δ_{total}, loss functions $\mathcal{L}_i$, KL bounds $D_{\mathrm{KL}}^{\max}(\pi_i \| \bar{\pi}_i)$, constant $\epsilon > 0$

Initialize ordered list $O \leftarrow []$, remaining agents $\mathcal{R} \leftarrow \mathcal{A}$
while $\mathcal{R} \neq \emptyset$ and $\delta_{\mathrm{total}} > 0$ **do**
  for all $i \in \mathcal{R}$ **do**
    Solve
    $$\max_{\pi_i} \mathcal{L}_i(\bar{\pi}_{1:i-1}, \pi_i)$$
    $$\text{s.t.} \quad \mathbb{E}_{s\sim d^{\pi_k}}\left[D_{\mathrm{KL}}\left(\pi_i(\cdot|s) \,\|\, \pi_k^i(\cdot|s)\right)\right] \le \delta_{total}$$
    Compute score: $\mathrm{Score}_i \leftarrow \frac{\mathcal{L}_i(\bar{\pi}_{1:i-1}, \pi_i)}{D_{\mathrm{KL}}^{\max}(\pi_i \| \bar{\pi}_i) + \epsilon}$
  end for
  Select agent $i^* \leftarrow \arg\max_{i \in \mathcal{R}} \mathrm{Score}_i$
  Append i^* to O, fix π_{i^*}, remove i^* from $\mathcal{R}$
  Update $\delta_{total} \leftarrow \delta_{total} - D_{\mathrm{KL}}^{\max}(\pi_i \| \bar{\pi}_i)$
end while

To improve coordination efficiency in multi-agent policy optimization, we modify the original HATRPO algorithm by introducing adaptive, agent-specific KL divergence thresholds. Instead of applying a uniform trust region constraint across all agents, our approach dynamically allocates the KL budget based on each agent's estimated contribution to overall performance. As shown in **Algorithm 1**, we incorporate a KL allocation strategy—either a greedy score-based ranking or a KKT-based optimization—that determines both the per-agent KL bounds and their update order. This enables the algorithm to sequentially optimize each agent's policy within a tailored trust region.

4.2 HATRPO-Greedy

We first propose an effective greedy algorithm, HATRPO-G to determine the order of policy updates. The idea is to prioritize agents whose updates provide the highest benefit-to-cost ratio, measured as the advantage gain divided by the KL divergence. At each step, we select the agent with the best ratio and fix its policy for subsequent evaluations. As described in **Algorithm 2**, the method constructs an ordered list of agents by selecting, at each iteration, the agent that maximizes the expected improvement per unit of KL cost. Specifically, each agent i is assigned a score given by

$$\mathrm{Score}_i = \frac{\mathcal{L}_i(\bar{\pi}_{1:i-1}, \pi_i)}{D_{\mathrm{KL}}^{\max}(\pi_i \| \bar{\pi}_i) + \epsilon},$$

where $\mathcal{L}_i$ represents the estimated local policy improvement and $\epsilon > 0$ is a small constant for numerical stability. The agent with the highest score is selected, added to the update order, and removed from the candidate pool. The process repeats until all agents are ranked. This greedy procedure introduces a soft prioritization mechanism that favors agents expected to yield higher utility per KL cost, resulting in a scalable approach for determining the agent update sequence without requiring global coordination or joint optimization.

Lemma 4.1 (Monotonic Bond). *Let π' be the old policy and π the new policy produced by Algorithm 1 and Algorithm 2. Then,*

$$J(\pi) \geq J(\pi') + (1-\tfrac{1}{e})\, L^* - C \sum_i D_{\max}(\pi'_i, \pi_i), \tag{7}$$

Proof. By submodularity, the surrogate objective corresponding to the greedy policy obtained from Algorithms 1 and 2 satisfies

$$L(\pi) \geq (1-\tfrac{1}{e})\, L^*. \tag{8}$$

The algorithm is based on HATRPO and follows its monotonic bond

$$J(\pi) \geq J(\pi') + L(\pi) - C \sum_i D_{\max}(\pi'_i, \pi_i). \tag{9}$$

Substituting the bound from Lemma 1 into the inequality above yields

$$J(\pi) \geq J(\pi') + (1-\tfrac{1}{e})\, L^* - C \sum_i D_{\max}(\pi'_i, \pi_i). \tag{10}$$

□

4.3 HATRPO-Weighted

In the HATRPO-W, we adapt the classical water-filling strategy [7, 27] from communications to formulate KL divergence allocation as a constrained optimization problem, solved using KKT conditions. Just as the water-filling method distributes power across noisy channels to maximize capacity, we allocate a total KL divergence budget across agents to maximize overall policy improvement. Agents with higher expected gain per unit KL—analogous to high-SNR channels—are assigned larger updates, while those with lower impact may receive little or none. This induces a soft prioritization mechanism that allocates the trust region more efficiently among agents. The resulting KKT-based allocation can be solved numerically (e.g., via bisection over the Lagrange multiplier) and yields a principled, globally coordinated update scheme under a shared KL constraint.

Algorithm 3: KL Allocation via KKT

Require: Fixed priors $\bar{\pi}$, advantage estimates A_i, total KL threshold δ_{total}, tolerance ε

Initialize $\delta_i \leftarrow 0$ for all agents i
Compute effective utility: $U_i \leftarrow \mathbb{E}_{a\sim\pi_i}[A_i(a)]$
$Order(i_{1:m})$ = Sort agents in decreasing order of U_i
Initialize $\lambda \leftarrow$ large positive value
repeat
  Compute tentative allocation: $\delta_i \leftarrow \max\left(0, \frac{U_i}{\lambda} - 1\right)$
  Evaluate total KL: $\delta = \sum_i \delta_i$
  Update $\lambda \leftarrow \lambda \cdot (\delta/\delta_{\text{total}})$
until $|\delta - \delta_{\text{total}}| < \varepsilon$
return $Order(i_{1:m}), \{\delta_i, ..., \delta_m\}$

We relax the total KL constraint using a Lagrange multiplier $\lambda \geq 0$, yielding the following Lagrangian objective:

$$\min_{\lambda} \max_{C_1,\ldots,C_m} \sum_{i=1}^{m} [\mathcal{L}_i(\bar{\pi}_{1:i-1}, \pi_i) - \lambda \cdot C_i] + \lambda, \delta_{\text{total}}$$

where C_i denotes the KL divergence between the updated policy π_i and its prior $\bar{\pi}_i$.

To obtain the optimal allocation, we formulate the problem as a min–max optimization based on the loss function. The loss function defines as follows:

$$\mathcal{L}_i(\bar{\pi}_{1:i-1}, \pi_i) \approx \sum_a A_i(a) \cdot \pi_i(a) \tag{11}$$

where $A_i(a)$ is the advantage function under the prior $\bar{\pi}_i$. We define the KL divergence:

$$C_i = \sum_a \pi(a) \log \frac{\pi(a)}{\pi'(a)}. \tag{12}$$

Optimizing the Lagrangian with respect to π_i, we obtain the condition:

$$\frac{\partial \mathcal{L}_i}{\partial C_i} = \lambda. \tag{13}$$

Assuming a linear relationship between utility and KL, we find a closed-form for the KL budget:

$$C_i \approx \frac{\mathbb{E}_{a\sim\pi_i}[A_i(a)]}{\lambda} - 1. \tag{14}$$

To ensure non-negativity of KL, we apply a projection:

$$C_i = \max\left(0, \frac{\mathbb{E}_{a\sim\pi_i}[A_i(a)]}{\lambda} - 1\right). \tag{15}$$

To satisfy the global trust region constraint, the total allocated KL divergence must equal δ_{total}:

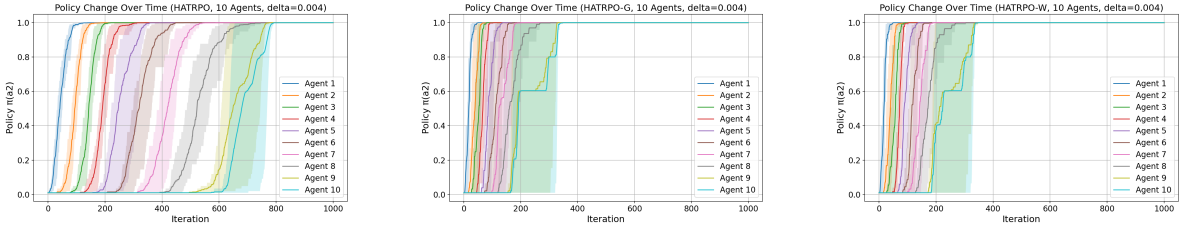
$$\sum_{i=1}^{m} C_i = \delta_{\text{total}}. \tag{16}$$

Substituting the expression for C_i, we obtain:

$$\sum_{i=1}^{m} \max\left(0, \frac{\mathbb{E}_{a\sim\pi_i}[A_i(a)]}{\lambda} - 1\right) = \delta_{\text{total}}. \tag{17}$$

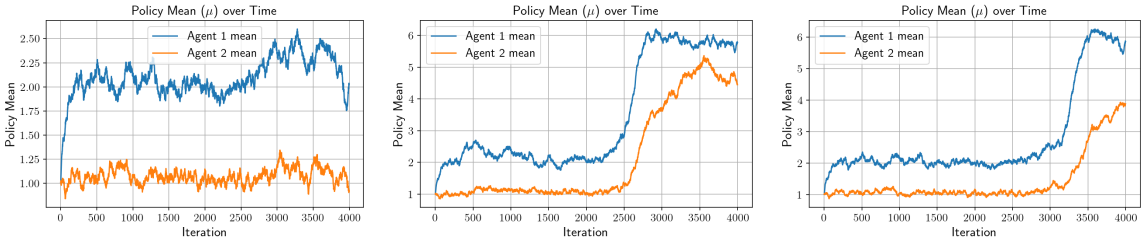
This nonlinear equation implicitly defines the optimal λ and can be efficiently solved using numerical methods such as bisection. Once λ is determined, each agent's KL allocation C_i can be computed accordingly, resulting in an adaptive and coordinated trust region allocation strategy.

In practice, we compute the optimal KL allocation by solving for the Lagrange multiplier λ using an iterative bisection-style update. As shown in **Algorithm 3**, we first compute each agent's utility score $U_i = \mathbb{E}_{a\sim\bar{\pi}_i}[A_i(a)]$, which measures the expected improvement under the prior policy. We initialize λ to a small positive value and repeatedly compute the tentative KL allocation for each agent using the closed-form: $D_{\text{KL}}^{\max}(i) = \max\left(0, \frac{U_i}{\lambda} - 1\right)$. This expression reflects the intuition that agents with higher utility receive more KL budget, while those with lower utility may receive none. We then evaluate the total KL allocation $\delta = \sum_i D_{\text{KL}}^{\max}(i)$ and adjust λ multiplicatively to match the target total KL threshold δ_{total}. This process is repeated until convergence, defined by a small absolute difference $|\delta - \delta_{\text{total}}| < \varepsilon$. This method provides a efficient solution to the KL allocation problem and scales easily to many agents.



(a) HATRPO (b) HATRPO-G (c) HATRPO-W

Figure 3: Policy trajectory evolution for all agents under different algorithms in the 10-agent matrix game.



(a) HATRPO (b) HATRPO-G (c) HATRPO-W

Figure 4: Policy trajectory evolution for all agents under different algorithms in the 2-player differential game.

5 RESULTS

We evaluate our proposed methods on three MARL settings of increasing complexity. 1) **Matrix Game: Coordination Under Sparse Reward** to study how KL allocation affects convergence speed in the presence of non-uniform advantage distributions. 2) **Differential Game: Escaping Local Optima** to test the ability of adaptive KL strategies to escape local optima. 3) **Multi-Agent MuJoCo: Realistic Heterogeneous Agents** to assess the scalability and adaptability of our methods in realistic settings.

Baselines. We compare our methods against several representative MARL algorithms: 1) **MADDPG** [12]; 2) **MAPPO** [26]; 3) **HAPPO** [8]; 4) **HATRPO** [8].

Implementation details, MARL benchmark descriptions, and hyperparameter settings are provided in Appendix A. We aim to answer the following research questions.

5.0.1 *(a) How does our algorithm compare to strong baselines under the same total KL divergence threshold?*

Our algorithm consistently outperforms or matches strong baselines across diverse environments under a fixed total KL divergence threshold shown in Figure 5 . Specifically, HATRPO-G improves final performance by 25.2%, while HATRPO-W achieves a 22.5% gain over the original HATRPO. However, HATRPO-G exhibits higher variability, with a standard deviation 39% greater than that of HATRPO-W. This reflects the broader generalization capabilities of our methods HATRPO-W. In both the matrix game and the differential game, our approach converges significantly faster and successfully escapes suboptimal local minima with reward plateaus, where HATRPO remains trapped. On Multi-Agent MuJoCo tasks, as shown in Fig. 5, particularly Ant and HalfCheetah, our method achieves notably higher returns. These results underscore the effectiveness and robustness of our KL-adaptive strategy across both discrete and continuous domains.

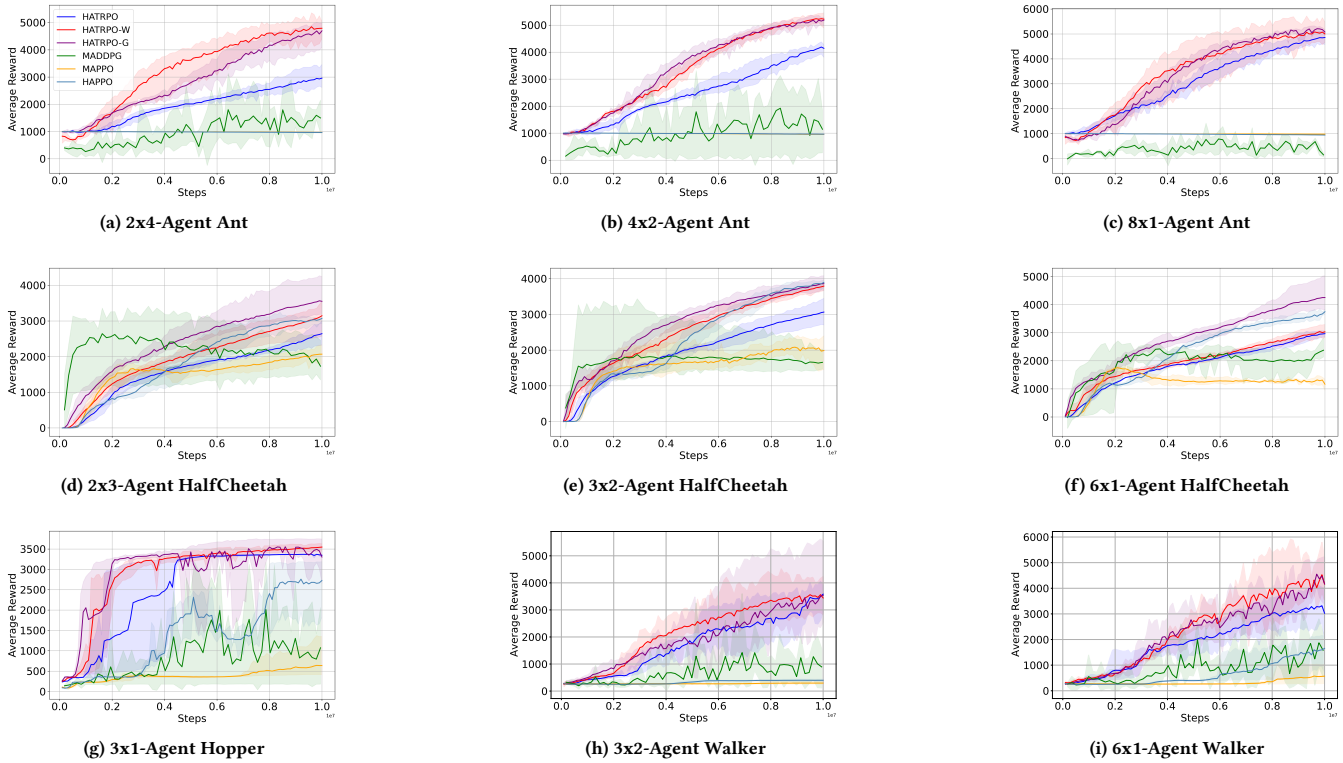
5.0.2 *(b) Can adaptive KL allocation lead to more effective and structured policy updates, especially in heterogeneous or imbalanced agent settings?*

We provide further analysis of policy behavior in the matrix game with 10 agents and differential game (see Figure 3 and Figure 4). In the matrix game, the reward structure depends on the actions of earlier-indexed agents, leading to asymmetries in advantage estimation. As a result, agents earlier in the update sequence tend to exhibit greater policy changes, while later agents often show minimal adaptation. Allocating more KL threshold to agents with higher advantage values—typically those in earlier positions—facilitates faster group convergence.

In contrast, the differential Gaussian game illustrates a case where standard HATRPO becomes stuck at a local optimum. Here, equal KL allocation prevents exploration beyond the local peak at (2, 1), especially for Agent 1. Our adaptive variants (HATRPO-G/W), which dynamically reassign KL constraints, enable Agent 1 to pursue larger updates and ultimately escape to the global optimum near (5, 5), demonstrating the benefits of asymmetric KL allocation for structured exploration and escape from local minima.

5.0.3 *(c) Does our method accelerate convergence by prioritizing policy updates for high-advantage agents?*

We further evaluate the convergence efficiency of each method in a 4-agent matrix game by measuring the number of steps required to reach 99% of the maximum achievable reward. As shown in Figure 7, HATRPO consistently requires more iterations to reach optimality, especially under tighter KL constraints (smaller δ values). This



(a) 2x4-Agent Ant (b) 4x2-Agent Ant (c) 8x1-Agent Ant

(d) 2x3-Agent HalfCheetah (e) 3x2-Agent HalfCheetah (f) 6x1-Agent HalfCheetah

(g) 3x1-Agent Hopper (h) 3x2-Agent Walker (i) 6x1-Agent Walker

Figure 5: Performance comparison on multiple Multi-Agent MuJoCo tasks. Our KL-adaptive methods, HATRPO-G and HATRPO-W, consistently outperform or match strong baselines under a fixed total KL divergence threshold.

performance gap underscores the inefficiencies of uniform KL distribution in environments with asymmetric advantage distributions. In contrast, both HATRPO-G and HATRPO-W exhibit significantly faster convergence.

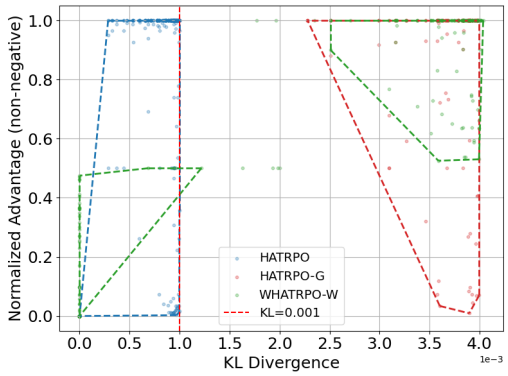


Figure 6: Normalized advantage vs. KL divergence for all methods with convex boundaries. Each point represents an agent's KL-advantage pair. HATRPO(blue) uses a uniform KL threshold, resulting in tightly bounded updates near the global KL limit (red dashed line at 0.0004). HATRPO-G(orange) and HATRPO-W(green) dynamically allocate KL based on agent-specific advantage signals.

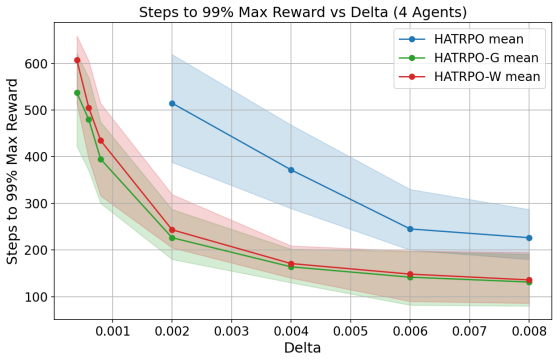


Figure 7: Number of steps required to reach 99% of the maximum reward as a function of the KL divergence constraint δ in the 4-agent matrix game.

5.0.4 (d) How well does the allocated KL threshold reflect each agent's contribution?

The allocated KL threshold closely reflects each agent's contribution as indicated by its advantage function. As shown in Figure 6, both HATRPO-G and HATRPO-W exhibit a strong correlation between KL divergence and normalized advantage. HATRPO-W allocates more KL to agents with the highest individual advantage. In contrast, HATRPO-G distributes KL more collectively, favoring

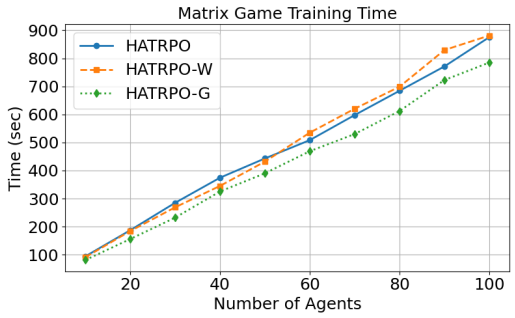


Figure 8: Training time in the Matrix Game environments.

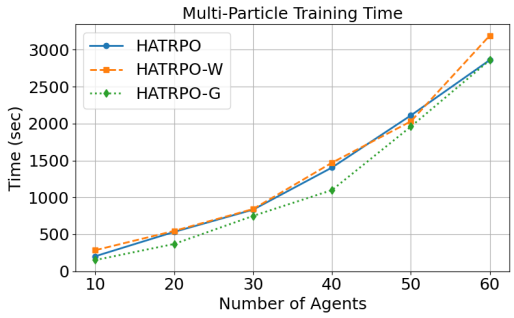


Figure 9: Training time in the Multi-Particle environments.

groups of agents with consistently high advantage. This alignment between KL allocation and advantage demonstrates that our adaptive methods effectively prioritize impactful updates, in contrast to HATRPO's uniform strategy which neglects agent-specific learning potential.

5.0.5 (e) Can our method overcome the bottlenecks of uniform KL constraints and improve joint policy optimization?

Yes, our method overcomes the bottlenecks of uniform KL constraints and leads to improved joint policy optimization. As shown in Figure 10, HATRPO allocates low and uniformly distributed KL values across agents and time, limiting its ability to prioritize critical updates. In contrast, both HATRPO-W and HATRPO-G exhibit structured and adaptive KL allocation. Notably, agent updates follow a sequential order (from agent 1 to agent 4), and the adaptive methods leverage this structure to focus learning more effectively. HATRPO-W updates agents in an alternating pattern—e.g., agent 3 → agent 4 → agent 3—allowing for iterative refinement, while HATRPO-G concentrates KL budget on a subset of high-advantage agents, enabling parallel but selective updates. This dynamic allocation enables more expressive and efficient joint policy optimization, particularly in heterogeneous-agent settings.

5.0.6 (f) What is the training time comparison with the baseline?

In this section, we compare the training time of our method with the baseline, HATRPO. All methods exhibit similar training durations. The results are presented in Fig. 8 and Fig. 9. We conducted experiments in both the Multi-Particle and Matrix Game environments, with the number of agents ranging from 10 to 100. The results demonstrate that our method maintains computational efficiency comparable to the original approach, even when scaling to different numbers of agents.

6 CONCLUSION

We introduced HATRPO-G and HATRPO-W, two adaptive extensions of HATRPO that address the inefficiencies of uniform KL divergence constraints in multi-agent reinforcement learning. By reallocating the total KL threshold based on agents' advantage signals, our methods enable more targeted policy updates that improve both learning speed and final performance. Experiments across matrix games, differential games, and multi-agent MuJoCo environments demonstrate that our adaptive algorithms converge

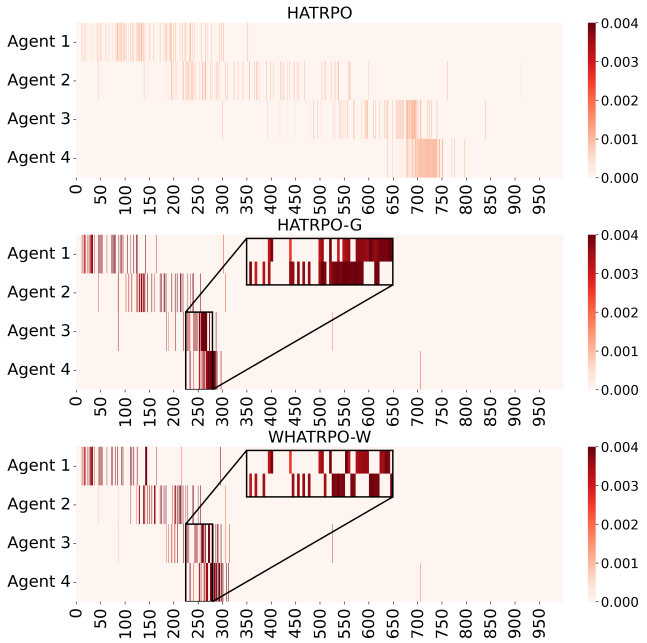


Figure 10: KL divergence over time for each agent across methods. Each heatmap shows the per-agent KL divergence at each policy update iteration. HATRPO (top) applies a uniform threshold to all agents, resulting in consistently low and evenly spread KL values over time. In contrast, HATRPO-G (middle) and HATRPO-W (bottom) adaptively allocate KL thresholds based on agent-specific learning signals.

faster and better utilize the available policy update threshold, particularly in settings with heterogeneous roles or imbalanced agent importance. These results suggest that our methods improve overall performance, promote more structured learning updates, and accelerate convergence, all while operating under a fixed total KL divergence budget. The learned KL threshold allocation effectively captures variations in agent advantage and overcomes the limitations of fixed-threshold designs, leading to more coordinated and efficient multi-agent learning.

REFERENCES

[1] Sushrut Bhalla, Sriram Ganapathi Subramanian, and Mark Crowley. 2020. Deep multi agent reinforcement learning for autonomous driving. In *Canadian Conference on Artificial Intelligence*. Springer, Springer International Publishing, Cham, Switzerland, 67–78.

[2] Lorenzo Canese, Gian Carlo Cardarilli, Luca Di Nunzio, Rocco Fazzolari, Daniele Giardino, Marco Re, and Sergio Spanò. 2021. Multi-agent reinforcement learning: A review of challenges and applications. *Applied Sciences* 11, 11 (2021), 4948.

[3] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip H. S. Torr, Mingfei Sun, and Shimon Whiteson. 2020. Is Independent Learning All You Need in the StarCraft Multi-Agent Challenge? *arXiv preprint arXiv:2011.09533* (2020). https://arxiv.org/abs/2011.09533

[4] Haowen Dou, Lujuan Dang, Zhirong Luan, and Badong Chen. 2024. Measuring mutual policy divergence for multi-agent sequential exploration. *Advances in Neural Information Processing Systems* 37 (2024), 76265–76288.

[5] Lang Feng, Dong Xing, Junru Zhang, and Gang Pan. 2023. FP3O: Enabling proximal policy optimization in multi-agent cooperation with parameter-sharing versatility. *arXiv preprint arXiv:2310.05053* (2023).

[6] Shangding Gu, Jakub Grudzien Kuba, Yuanpei Chen, Yali Du, Long Yang, Alois Knoll, and Yaodong Yang. 2023. Safe multi-agent reinforcement learning for multi-robot control. *Artificial Intelligence* 319 (2023), 103905.

[7] Peter He, Lian Zhao, Sheng Zhou, and Zhisheng Niu. 2013. Water-Filling: A Geometric Approach and its Application to Solve Generalized Radio Resource Allocation Problems. *IEEE Transactions on Wireless Communications* 12, 7 (Jan. 2013), 3637–3647. https://doi.org/10.1109/TWC.2013.061713.130278

[8] Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. 2021. Trust region policy optimisation in multi-agent reinforcement learning. *arXiv preprint arXiv:2109.11251* (2021).

[9] Jakub Grudzien Kuba, Xidong Feng, Shiyao Ding, Hao Dong, Jun Wang, and Yaodong Yang. 2022. Heterogeneous-agent mirror learning: A continuum of solutions to cooperative marl. *arXiv preprint arXiv:2208.01682* (2022).

[10] Chenxing Liu and Guizhong Liu. 2024. JointPPO: diving deeper into the effectiveness of PPO in multi-agent reinforcement learning. *arXiv preprint arXiv:2404.11831* (2024).

[11] Jiarong Liu, Yifan Zhong, Siyi Hu, Haobo Fu, Qiang Fu, Xiaojun Chang, and Yaodong Yang. 2023. Maximum entropy heterogeneous-agent reinforcement learning. *arXiv preprint arXiv:2306.10715* (2023).

[12] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* 30 (2017).

[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[14] Keonwoo Park and Ilkyeong Moon. 2022. Multi-agent deep reinforcement learning approach for EV charging scheduling in a smart grid. *Applied energy* 328 (2022), 120111.

[15] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. 2020. Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. *Advances in neural information processing systems* 33 (2020), 10199–10210.

[16] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2020. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research* 21, 178 (2020), 1–51.

[17] Martin Roesch, Christian Linder, Roland Zimmermann, Andreas Rudolf, Andrea Hohmann, and Gunther Reinhart. 2020. Smart grid for industry using multi-agent reinforcement learning. *Applied Sciences* 10, 19 (2020), 6900.

[18] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*. PMLR, PMLR, Lille, France, 1889–1897.

[19] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[20] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300* (2024).

[21] Chak Lam Shek, Amrit Singh Bedi, Anjon Basak, Ellen Novoseller, Nick Waytowich, Priya Narayanan, Dinesh Manocha, and Pratap Tokekar. 2025. Learning Multi-Robot Coordination through Locality-Based Factorized Multi-Agent Actor-Critic Algorithm. *arXiv e-prints* (2025), arXiv–2503.

[22] YANG Tao, SHI Xinhao, XU Cheng, YANG Yulin, ZENG Qinghan, and LIU Hongzhe. 2025. TSPPO: Transformer-Based Sequential Proximal Policy Optimization for Multi-Agent Systems. *Preprint at Research Square* (2025).

[23] Xihuai Wang, Zheng Tian, Ziyu Wan, Ying Wen, Jun Wang, and Weinan Zhang. 2023. Order matters: Agent-by-agent policy optimization. *arXiv preprint arXiv:2302.06205* (2023).

[24] Ying Wen, Hui Chen, Yaodong Yang, Minne Li, Zheng Tian, Xu Chen, and Jun Wang. 2022. A game-theoretic approach to multi-agent trust region optimization. In *International conference on distributed artificial intelligence*. Springer, 74–87.

[25] Xiyang Wu, Rohan Chandra, Tianrui Guan, Amrit Bedi, and Dinesh Manocha. 2023. Intent-aware planning in heterogeneous traffic via distributed multi-agent reinforcement learning. In *Conference on Robot Learning*. PMLR, 446–477.

[26] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. 2022. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information processing systems* 35 (2022), 24611–24624.

[27] David D. Yu and John M. Cioffi. 2006. SPC10-2: Iterative Water-filling for Optimal Resource Allocation in OFDM Multiple-Access and Broadcast Channels. In *IEEE Globecom 2006*. IEEE, San Francisco, California, USA, 1–5. https://doi.org/10.1109/GLOCOM.2006.587

[28] Tian Yu, Jing Huang, and Qing Chang. 2021. Optimizing task scheduling in human-robot collaboration with deep multi-agent reinforcement learning. *Journal of Manufacturing Systems* 60 (2021), 487–499.

[29] Bin Zhang, Hangyu Mao, Lijuan Li, Zhiwei Xu, Dapeng Li, Rui Zhao, and Guoliang Fan. 2024. Sequential asynchronous action coordination in multi-agent systems: A stackelberg decision transformer approach. In *Forty-first International Conference on Machine Learning*.

[30] Wenshuai Zhao, Yi Zhao, Zhiyuan Li, Juho Kannala, and Joni Pajarinen. 2023. Optimistic multi-agent policy gradient. *arXiv preprint arXiv:2311.01953* (2023).

[31] Zhitong Zhao, Ya Zhang, Wenyu Chen, Fan Zhang, Siying Wang, and Yang Zhou. 2025. Sequence Value Decomposition Transformer for Cooperative Multi-Agent Reinforcement Learning. *Information Sciences* (2025), 122514.

[32] Yifan Zhong, Jakub Grudzien Kuba, Xidong Feng, Siyi Hu, Jiaming Ji, and Yaodong Yang. 2024. Heterogeneous-agent reinforcement learning. *Journal of Machine Learning Research* 25, 32 (2024), 1–67.