AdaptAUG: Adaptive Data Augmentation Framework for Multi-Agent Reinforcement Learning

Xin Yu[1], Yongkai Tian[1], Li Wang[2], Pu Feng[1], Wenjun Wu[2], and Rongye Shi[2,*]

***Abstract*— Multi-agent reinforcement learning has emerged as a promising approach for the control of multi-robot systems. Nevertheless, the low sample efficiency of MARL poses a significant obstacle to its broader application in robotics. While data augmentation appears to be a straightforward solution for improving sample efficiency, it usually incurs training instability, making the sample efficiency worse. Moreover, manually choosing suitable augmentations for a variety of tasks is a tedious and time-consuming process. To mitigate these challenges, our research theoretically analyzes the implications of data augmentation on MARL algorithms. Guided by these insights, we present AdaptAUG, an adaptive framework designed to selectively identify beneficial data augmentations, thereby achieving superior sample efficiency and overall performance in multi-robot tasks. Extensive experiments in both simulated and real-world multi-robot scenarios validate the effectiveness of our proposed framework.**

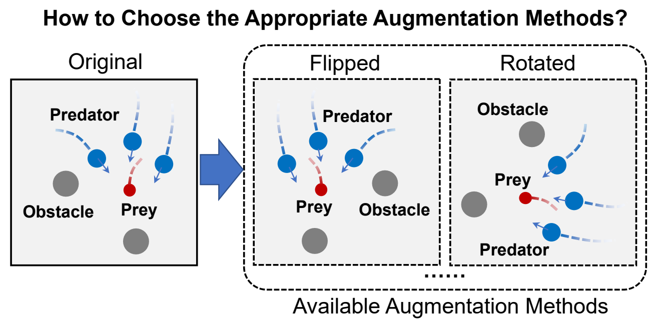


Fig. 1: Data Augmentation Techniques in MARL: This figure displays the original data along with its y-axis flipped and $\frac{\pi}{2}$ rotated versions. This paper aims to automate the selection of suitable augmentation techniques for a specific task.

I. INTRODUCTION

Multi-agent reinforcement learning (MARL) has gained considerable attention as an effective approach for controlling multi-robot systems [1]. The technique holds substantial promise in various applications such as autonomous navigation and cooperative exploration [2]. However, one significant drawback of applying MARL in these scenarios is its low sample efficiency, a limitation that hampers its broader implementation in real-world [3], [4].

Improving the sample efficiency of MARL algorithms through data augmentation appears to be an intuitively straightforward approach [5]. However, the stability of RL algorithms can be compromised when data augmentation is applied in a naive manner [6]. This issue is further exacerbated in MARL, as the number of agents increases, training instability becomes more pronounced [7]. Moreover, different reinforcement learning tasks may demand specialized types of data augmentation for optimal performance [6]. Current methods for selecting suitable augmentations are far from ideal: they either rely on expert knowledge, which may not be universally available or applicable [8], or they involve evaluating a broad spectrum of transformations to identify the most effective ones [9]. Both approaches come with their own limitations, such as inefficiency or the risk of suboptimal selection.

Given the low data efficiency in MARL, enhancing the feasibility of data augmentation becomes imperative. A desirable attribute for applying data augmentation would be to ensure a stable training process. Motivated by this, we introduce the concept of **Augmentation Sensitivity** to evaluate the impact of different augmentation methods on current policy. Theoretical analysis shows that minimizing the Augmentation Sensitivity leads to reduced algorithmic error.

Inspired by these findings, we propose AdaptAUG, an adaptive data augmentation framework specifically designed for MARL. AdaptAUG selects suitable data augmentation methods during the training of MARL by balancing both reward metrics and a new metric, the Augmentation Sensitivity. To tackle this trade-off, we employ the Multi-Objective Multi-Armed Bandit algorithm[10]. In this setup, each arm corresponds to a different data augmentation technique. As training progresses, AdaptAUG continuously pulls these arms to learn the trade-offs between competing objectives, eventually identifying an appropriate set of data augmentations. This enhances the training process, resulting in both improved data efficiency and final performance metrics. Our approach enables the intelligent selection of the most beneficial augmentations for any given task, thereby increasing the stability of training and the quality of task performance.

We validate our approach on various multi-agent tasks, such as pursuit-evasion, cooperative navigation, and formation change. Our results demonstrate that AdaptAUG is highly effective in identifying beneficial augmentations, often performing comparably or better than the most effective manual augmentations. Moreover, AdaptAUG outperforms baseline methods both in training and in real-world testing scenarios.

The contributions of this paper are summarized as follows:

- We introduce the concept of the **Augmentation Sen-**

[1]School of Computer Science and Engineering, Beihang University, Beijing, China.
[2]Institute of Artificial Intelligence, Beihang University, Beijing, China.
*Corresponding author: Rongye Shi (shirongye@buaa.edu.cn).

sitivity, a metric for assessing the impact of data augmentation methods on the current policy. Through theoretical analysis, we show that minimizing this leads to reduced algorithmic error.

- Motivated by the theoretical analysis, we propose **AdaptAUG**, a novel adaptive data augmentation framework designed specifically for MARL. AdaptAUG not only improves the data efficiency of MARL algorithms but also ensures stable training.
- Through empirical validation on multiple multi-agent tasks, AdaptAUG demonstrates competitive or superior sample efficiency and convergence reward. Additionally, we validate the effectiveness of our learned policies in real-world robotic environments, showing improved sim-to-real transfer capabilities.

II. RELATED WORK

A. Data Augmentation in Single-Agent RL

Data augmentation has found broad application in computer vision, notably in supervised and self-supervised learning, and has recently shown potential in contrastive learning across various tasks [8], [11], [12], [13], [14]. In robotics, domain randomization serves as a specialized form of data augmentation, facilitating the transition of reinforcement learning policies from simulations to real-world applications, although it often requires a physics simulator [15], [16]. While recent work has delved into data augmentation in reinforcement learning, most strategies are fixed and lack dynamic adaptability, often being borrowed from supervised learning settings [17]. Some work has explored the effects of various data augmentations in offline RL but did not dynamically choose the augmentation methods [18]. In contrast, our work uniquely focuses on dynamically selecting the optimal subset of augmentations from a predefined set, ensuring algorithmic stability throughout the process.

B. Data Augmentation in Multi-Agent RL

Data augmentation has been less extensively studied in the context of MARL, with the primary focus being on the exchange invariance of homogenous groups to facilitate parameter sharing [7]. Another category of data augmentation in MARL utilizes rotational symmetries and employs symmetry loss functions to assist [19]. Similar to single-agent settings, domain randomization also serves as a unique form of data augmentation when applying reinforcement learning to multi-robot systems [20]. However, existing approaches generally exploit single properties without examining the cumulative effects of multiple augmentation techniques on MARL algorithms. Moreover, there's a lack of research exploring adaptive data augmentation in MARL contexts. As depicted in Figure 1, facing different scenarios, it is difficult to know which augmentation method can achieve the best results. Besides, existing literature has primarily focused on empirical studies, leaving the theoretical impact of data augmentation on MARL algorithms less explored.

III. PRELIMINARIES

A. Cooperative Markov Game

We model the multi-agent cooperative problem as a cooperative Markov game [21]. An n-agent cooperative Markov game is formally represented by a tuple $M = (N, S, \{A_i\}_{i=1}^{n}, R, T)$. Here, N signifies the set of agents, S represents the state space, and A_i denotes the action space for agent i, where i ranges from 1 to n. The joint action space is denoted as $A = A_1 \times A_2 \times \cdots \times A_n$. $T: S \times A \times S \to [0,1]$ is the transition function. At time step t, each agent then takes an independent action $(a_1, \ldots, a_N)$ based on its individual policy. Then, the environment emits the bounded joint reward R and moves to the next state s_{t+1}. The agents aim to maximize the expected joint return, defined as $E_\pi\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$, where γ is the discount factor, by selecting actions according to the policy $\pi_i : S \times A_i \to [0,1]$. The initial states are determined by a distribution $\eta : S \to [0,1]$.

IV. THEORETICAL ANALYSIS FOR DATA AUGMENTATION

To investigate which types of data augmentation are suitable, this section provides an analysis of data augmentation in the context of MRAL.

***Definition** 1 (Optimality Augmentation):* Given a cooperative Markov game M, we define the state transformation and action transformation as $L_f : S \to S$ and $K_f : A \to A$ respectively, where f is a certain transformation. Transformation L_f and K_f are the Optimality Augmentation if $\forall s \in S, a \in A$, satisfies:

$$Q(s,a) = Q(L_f[s], K_f[a])$$
$$\pi(a \mid s) = \pi(K_f[a] \mid L_f[s]).$$

It has been proven that training RL using Optimality Augmentation does not introduce additional errors.[5]. For state-action pairs (s,a), we denote the augmented state-action pairs as (fs, fa) where $fs = L_f(s)$ and $fa = K_f(a)$ for short.

***Definition** 2 (Augmentation Sensitivity):* Given an augmentation function f, the augmentation sensitivity of a policy π is defined as:

$$\varepsilon_\pi^f = \sup_{s \in S} \Delta(\pi(\cdot \mid s) \| \pi(\cdot \mid fs)) \tag{1}$$

where $\Delta(\mu, \nu) = \frac{1}{2} \sum_{a \in A} |\mu(a) - \nu(a)|$ measures the distance between two probability distributions.

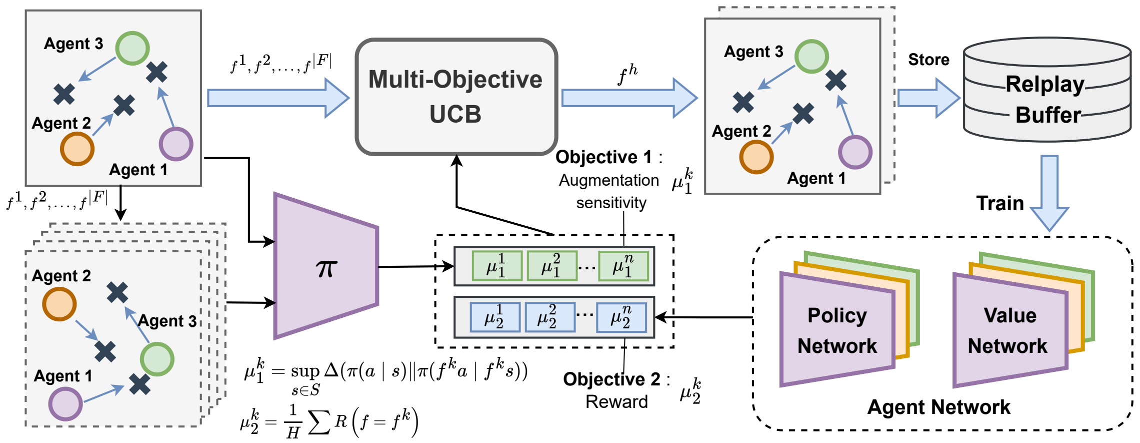
Based on the definition provided, we propose that a lower augmentation sensitivity for a given augmentation results in a more precise bound on Q-value estimation during data augmentation.

***Proposition** 1:* Consider a cooperative Markov game M, a policy π and transformation f. For $\forall a \in A, \forall s \in S$, the following inequality holds:

$$|Q^\pi(s,a) - Q^\pi(fs, fa)| \leq 2R_{\max} \frac{\varepsilon_\pi^f + 1}{1-\gamma}. \tag{2}$$

We denote the bound of reward as $R_{\max}$. The proof for Proposition 1 is provided in detail within the Appendix[1].

[1]Video demonstrations and appendix are available at the project website https://xinyu-site.github.io/AdaptAUG/.



Agent 3
Agent 2
Agent 1
$f^1, f^2, \ldots, f^{|F|}$
Multi-Objective UCB
f^h
Store
Relplay Buffer
Train
Objective 1 : Augmentation sensitivity μ_1^k
π
μ_1^1 μ_1^2 ⋯ μ_1^n
μ_2^1 μ_2^2 ⋯ μ_2^n
$\mu_1^k = \sup_{s\in S} \Delta(\pi(a \mid s) \| \pi(f^k a \mid f^k s))$
$\mu_2^k = \frac{1}{H}\sum R\left(f = f^k\right)$
Objective 2 : Reward μ_2^k
Policy Network
Value Network
Agent Network

Fig. 2: Illustration of the proposed AdaptAUG framework. The framework calculates two objective vectors for each augmentation method f^k. By balancing these objectives, AdaptAUG selects an augmentation to apply to the original data. Both the augmented and original data are then stored in the replay buffer for training the agent network.

V. Methods

This paper focuses on solving the following problem: *How to choose appropriate augmentation methods to enhance the performance of MARL in a specific task?* We introduce a general framework called Adaptive Data Augmentation (AdaptAUG) for selecting augmentation methods. The AdaptAUG framework is illustrated in Figure 2.

A. Data Augmentation Policy

In our research, we consider two criteria for selecting data augmentation techniques. Firstly, based on Proposition 1, we prefer techniques with lower sensitivity to ensure more precise Q-value estimation. Such techniques also enhance algorithmic stability, as outlined in [automatic]. Secondly, it is crucial that these augmentations are designed to optimize the reinforcement learning (RL) reward. To address this dual-objective selection challenge, we adopt the framework of Multi-Objective Multi-Armed Bandit [10], [22]. Within this framework, each 'arm' symbolizes a distinct data augmentation strategy. Our AdaptAUG framework systematically explores various strategies to balance the two objectives, continuously evaluating the trade-offs between enhancing Q-value accuracy and maximizing RL reward.

Multi-Objective UCB. The Multi-Objective UCB framework utilizes a set of 'arms' $F = \{f^1, \ldots, f^{|F|}\}$, with $|F|$ denoting the number of data augmentations. Each arm k is associated with an expectation vector $\mu^k = [\mu_1^k, \ldots, \mu_N^k]$, representing its expected performance across N objectives. This vector reflects the anticipated outcomes for each augmentation strategy. The UCB's action space encompasses the suite of available transformations, enabling informed strategic decisions based on these expectations.

Objective Vector. Our framework integrates Multi-Objective UCB into the RL training process to optimize data augmentation selections. The first objective aims to select augmentations characterized by lower sensitivity. The estimated expectation for objective 1, specific to arm k, is then calculated as follows:

$$\mu_1^k = \sup_{s\in S} \Delta(\pi(a \mid s) \| \pi(f^k a \mid f^k s)). \tag{3}$$

The second objective aims to select augmentation methods that yield higher RL rewards. For each f^k, We maintain a queue to store the rewards obtained from the most recent L selections. These rewards are then used to estimate μ_2^k by:

$$\mu_2^k = \frac{1}{L}\sum_{i=0}^{L} R_i^k \tag{4}$$

where R_i^k represents the reward obtained at the i-th selection of the augmentation f^k. This estimate is calculated as a sliding window average of the rewards.

Augmentation Selection. The AdaptAUG framework begins by playing each augmentation method once to collect initial data. For each iteration t and each arm k, we calculate a term that combines the estimated expectation vector $\hat{\mu}^k(t)$ with a confidence interval. The term is expressed as $\hat{\mu}^k(t) + \sqrt{\frac{2\ln t \sqrt[4]{N|F|}}{C_t(f^k)}}$, where $C_t(f^k)$ is the number of times transformation f^k has been selected before time step t. After these calculations, we establish a Pareto optimal set F_t^* to select an augmentation method randomly. The Pareto optimal set F_t^* includes $h \in F$ if and only if there does not exist any $f^k \in F$ such that:

$$\hat{\mu}^k(t) + \sqrt{\frac{2\ln t \sqrt[4]{N|F|}}{C_t(f^k)}} > \hat{\mu}^h(t) + \sqrt{\frac{2\ln t \sqrt[4]{N|F|}}{C_t(f^h)}}. \tag{5}$$

This means that no arm outperforms f^h across all objectives. Once the Pareto optimal set F_t^* is obtained, we proceed with the following steps. First, we randomly select a data augmentation method from F_t^*. Next, we update the agent's policy and value network using the MARL algorithm. Subsequently, we collect rollouts using the newly updated policy to gather data. Then we update the estimated $\hat{\mu}^h(t)$. Finally, the counter $C_t(f^h)$ is incremented to $C_t(f^h) = C_{t-1}(f^h) + 1$.

B. Data Augmentation for MARL

In contrast to data augmentation techniques commonly employed in the field of computer vision, those in reinforcement learning often consider both states and actions. To our knowledge, there is limited research on data augmentation methods in the context of MARL [8]. This section introduces multiple data augmentation approaches specifically designed for MARL. we use $s_t^{(i)}$ to represent the i^{th} dimension of the state vector at time t. Additionally, $\hat{s}_t$ is used to denote the augmented state. The hyperparameters for the augmentation methods used in this paper are provided below. Other parameters are also viable.

1) **Rotation Augmentation:** In multi-agent scenarios where spatial symmetry is prevalent, simultaneous rotation of both state and actions can yield additional training samples [5]. The augmented state is given by $\hat{s}_t \leftarrow \text{Rotate}(s_t, \theta)$, where θ specifies the angle of rotation. We choose $\theta = \frac{\pi}{2}$ for the remainder of this paper, though other angles can be considered based on specific circumstances.
2) **Flip Augmentation:** In scenarios where certain dimensions of the state can be meaningfully flipped, this augmentation applies $\hat{s}_t \leftarrow \text{Flip}(s_t, d)$, where d specifies the axis to be flipped. We choose the y axis to make the augmentation.
3) **Zero-mean Gaussian Noise:** The augmented state $\hat{s}_t$ is given by $\hat{s}_t \leftarrow s_t + \varepsilon$, where ε is sampled from a zero-mean Gaussian distribution with variance σ. We set $\sigma = 0.2$.
4) **Zero-mean Uniform Noise:** This augmentation adds zero-mean uniform noise to the state, formulated as $\hat{s}_t \leftarrow s_t + \varepsilon$, where ε is sampled from a uniform distribution $U(-\alpha, \alpha)$. Here, we set $\alpha = 0.2$ to define the range of the noise.
5) **Amplitude Scaling:** This augmentation scales a random dimension d of the state $s_t^{(d)}$ by a random factor ε_d, with $\varepsilon_d \in U(0.5, 1.5)$. The augmented state in this dimension is $\hat{s}_t^{(d)} = s_t^{(d)} \times \varepsilon_d$.
6) **Dimension Dropout:** To simulate potential sensor malfunctions, this augmentation sets one randomly selected dimension i to zero. Specifically, $\hat{s}_t^{(i)} = 0$, where i is sampled uniformly from the available dimensions. Extending this approach to zero multiple dimensions is also feasible.
7) **State-switch:** This method swaps two randomly selected dimensions i and j within the state vector s_t to create $\hat{s}_t$. The augmented state is computed as $\hat{s}_t^{(i,j)} = s_t^{(j,i)}$. The aim is to simulate the uncertainty of complex environments and to enhance data diversity.
8) **Mix-up:** The augmented state is computed as $\hat{s}_t = \lambda s_t + (1-\lambda)s_{t+1}$, where λ is a weight that controls the degree of mixing between the two states. We set λ = 0.5 in the rest of this paper.

The data augmentation techniques introduced in this section demonstrate various methods to enhance algorithm performance and robustness in MARL.

Algorithm 1 AdaptAUG

```
Input: Set of transformations $F = \{f^1, \ldots, f^{|F|}\}$, window length $L$, number of updates $T$.
1: Initialize the agent networks.
2: For all $f^k \in F$, initialize $C_t(f^k) = 1, \hat{\mu}^k(t) = [0, \ldots, 0]$.
3: Initialize an empty queue with length $L$ for each augmentation to store the RL returns.
4: Play each arm once to gather initial data.
5: for $t = 1, \ldots, T$ do
6:     For all $f^k \in F$, compute $\hat{\mu}^k(t) + \sqrt{\frac{2\ln t \sqrt[4]{N|F|}}{C_t(f^k)}}$
7:     Find set $F_t^*$ and select an augmentation $f^h \in F_t^*$
8:     Update the agent network according to MARL.
9:     Update the estimated $\hat{\mu}^h(t)$ using equation (3) and (4)
10:    $C_t(f^h) \leftarrow C_{t-1}(f^h) + 1$
11: end for
```

C. Algorithm Description

The AdaptAUG algorithm is designed to optimize data augmentation strategies within a reinforcement learning framework as shown in algorithm 1. It begins by initializing agent networks and setting up a confidence score and an estimated performance vector for each data transformation. An essential feature of AdaptAUG is its iterative process, where it systematically evaluates each augmentation technique over T updates. At each iteration, the algorithm calculates an exploration term for each transformation, balancing the need to explore new strategies with the exploitation of known ones. This is achieved by adjusting the estimated performance vector $\hat{\mu}^k(t)$ with a dynamic exploration bonus, encouraging the selection of less-explored yet potentially rewarding augmentations. The selected augmentation, f^h, is then applied, and the agent network is updated according to MARL algorithm. Performance estimations are refined based on the outcomes of these calculations, ensuring the algorithm progressively focuses on the most effective augmentation strategies. This cycle of selection, application, and update continues until all T updates are completed, culminating in an optimized augmentation framework designed to boost learning efficiency and effectiveness.

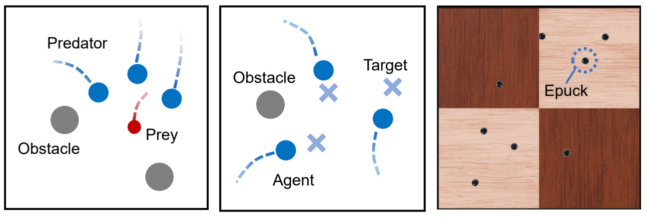
VI. Experiments

A. Experimental Settings

The proposed framework was applied to several mainstream algorithms, including MAPPO, MADDPG, QMIX [23], [24], [25]. All experiments have been implemented using Python 3 on a computer with an i7-13700KF CPU, 64GB RAM, and an Nvidia RTX 4080 GPU. To account for variability and ensure statistical significance, each experiment was repeated across 10 distinct random seeds. We carried out experiments across various tasks, namely Predator-Prey, Cooperative Navigation, and Formation Change, as depicted in Figure 3.

TABLE I: Episode rewards for AdaptAUG and the baseline algorithms are presented across three tasks. In the table, 'AUG' represents the algorithm implemented within the AdaptAUG. The terms **500k** and **3000k** indicate the number of training steps taken by the algorithms. Error bars represent the standard error of the mean, calculated over 10 random seeds.

Task	Steps	Algorithms					
		MADDPG	**MADDPG-AUG**	QMIX	**QMIX-AUG**	MAPPO	**MAPPO-AUG**
Predator Prey	500k	60.57±10.84	**83.33±6.98**	41.91±7.17	**60.42±5.36**	32.22±5.7	**47.09±6.0**
	3000k	87.83±8.08	**90.45±6.51**	53.02±6.91	**57.68±5.56**	33.56±7.3	**58.24±6.61**
Cooperative Navigation	500k	-142.28±3.75	**-129.48±1.81**	-152.34±3.14	**-138.8±1.5**	-130.51±2.01	**-119.79±1.88**
	3000k	-120.78±4.92	**-111.97±2.15**	-122.26±4.23	**-113.09±1.44**	-123.52±1.96	**-111.28±1.13**
Formation Change	200k	-42.87±13.62	**-40.45±10.53**	50.97±18.7	**53.36±14.44**	126.05±12.69	**164.01±14.51**
	1000k	**-40.01±10.47**	-43.51±11.12	51.77±18.56	**59.37±16.58**	142.43±18.56	**170.97±11.56**



(a) Predator-Prey (b) Navigation (c) Formation change

Fig. 3: Schematic diagrams of the three simulated tasks employed in our experiments to evaluate the framework.

Predator-Prey. This is a well-known scenario, implemented in [26], where collaborating agents aim to capture prey. A reward is allocated to predators at each time step if any of them occupies the same space as the prey. Figure 3a showcases the Predator-Prey task. In this scenario, we configure three predators and one prey.

Cooperative Navigation. This scenario implemented by [26], requires agents to navigate towards target landmarks while avoiding inter-agent collisions. At the beginning of each episode, agents and landmarks are randomly positioned. A standard setup involving three agents for this task is illustrated in Figure 3b. In this task, we deploy three agents with the objective of reaching three landmarks while minimizing collisions.

Formation Change. As depicted in Figure 3c, all robots initiate in a square formation, aiming to reach their designated positions on the opposite side. The task employs the E-puck in the open-source Webots simulator [27]. E-pucks are trained to circumvent each other while coordinating to reach their destinations. We set up 8 E-pucks in the task.

B. Main Results

We primarily report the reward at two key stages during training: the early stage of training (500k) and the convergence reward (3000k). The early-stage reward serves as an indicator of sample efficiency to some extent, while the 3000k reward reflects the convergence performance.

Predator-Prey Task: As demonstrated in Table I, our proposed AdaptAUG framework significantly outperforms the baseline methods, including a substantial improvement over MAPPO. This indicates that AdaptAUG can improve both data efficiency and convergence reward.

Cooperative Navigation Task: The reward during training for this task is presented in Table I. Our framework demonstrates a distinct advantage in enhancing data efficiency, resulting in faster convergence to higher reward values compared to baseline methods.

Formation Change Task: To assess the efficacy of our method in more complicated tasks, we conducted experiments on a multi-robot formation change task in the Webots simulator, as illustrated in Figure 3c. Experimental results demonstrate that MADDPG and QMIX fail to learn useful policies in this task, whereas agents trained with MAPPO-AUG and MAPPO manage to reach the goals while avoiding collisions and obstacles. As shown in Table I, MAPPO augmented by our proposed framework yields higher rewards compared to its original version, indicating that our approach effectively enhances the performance of data augmentation in challenging environments.

C. Data Augmentations for Each Task

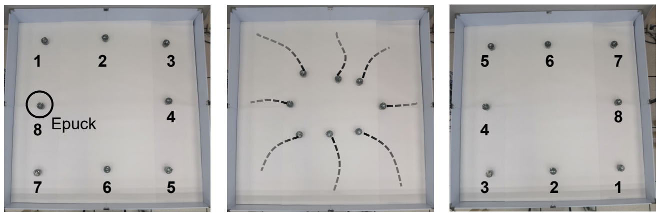
The experimental outcomes are summarized in Table II, where rotation involved a $\frac{\pi}{2}$ clockwise adjustment, and flipping was executed along the y-axis. Further details on hyperparameters can be found in Section V.B. This table presents the average of the final converged rewards, with "Normal" indicating the baseline performance without any data augmentation. The experimental results indicate that in the Cooperative Navigation and Predator-Prey tasks, Rotation, Flip, and MixUp positively impact the algorithm's performance. This is because Rotation and Flip satisfy the condition of Optimality Augmentation, thereby introducing no additional error. MixUp creates new data points by blending existing ones, thereby enhancing the algorithm's robustness. On the other hand, other augmentation methods disrupt the sample distribution to some extent and thus can have negative effects. For the Formation Change task, we found that our Rotation, MixUp, and Gaussian Noise methods were effective. This phenomenon may be attributed to the Webots simulation environment's realistic nature. Its inherent dynamics already include a level of noise, so adding noise to the samples can indeed improve robustness.

TABLE II: Investigating the importance of different augmentation schemes in MARL. We train policy using different augmentation schemes and report the mean normalized scores over 10 random seeds. To make the table easier to read we color the best algorithm on a task in **deep blue**, the second best in **deep brown**, and the third best in **deep green**.

Task	Algorithm	Augmentations									
		Normal	Rotation	Flip	Gaussian	Uniform	Amp	Dropout	Switch	MixUp	AdaptAUG
Predator Prey	MADDPG	87.83	**89.90**	89.83	86.71	87.63	82.53	84.13	83.73	**90.33**	**90.45**
	QMIX	53.02	54.01	**55.07**	52.42	51.71	48.12	48.82	49.02	**54.72**	**57.68**
	MAPPO	33.56	42.76	**45.56**	36.16	32.56	29.16	29.36	31.56	**43.76**	**58.24**
Cooperative Navigation	MADDPG	-120.78	**-115.78**	-117.78	-121.89	-125.84	-125.78	-123.78	-122.78	**-115.88**	**-111.97**
	QMIX	-122.26	**-117.16**	**-113.56**	-123.81	-125.30	-126.27	-125.35	-126.29	-118.16	**-113.09**
	MAPPO	-123.52	-118.31	**-117.02**	**-116.62**	-121.72	-128.62	-129.52	-127.52	-119.52	**-111.28**
Formation Change	MADDPG	**-40.01**	**-35.07**	-41.05	-41.15	**-40.71**	-44.11	-43.31	-41.71	-42.94	-43.51
	QMIX	51.77	**56.17**	54.27	**56.67**	48.67	49.71	49.78	47.87	55.77	**59.37**
	MAPPO	142.17	**149.43**	147.43	143.53	141.33	142. 73	140.21	138.48	**151.41**	**170.97**

TABLE III: Output augmentations type for each task.

TASK	Algorithm	Augmentations
Predator Prey	MADDPG	Rotation+Flip+MixUp
	QMIX	Rotation+Flip+MixUp
	MAPPO	Rotation+Flip+MixUp
Cooperative Navigation	MADDPG	Rotation+Flip+MixUp
	QMIX	Rotation+Flip+MixUp
	MAPPO	Rotation+Flip+MixUp
Formation Change	MADDPG	Rotation+MixUp+Gaussian noise
	QMIX	Rotation+MixUp+Gaussian noise
	MAPPO	Rotation+MixUp+Gaussian noise



(a) Start points (b) Trajectories (c) End points

Fig. 4: Real-world formation change on a swarm of E-puck robots. The E-puck robots successfully switched their positions to the antipodal points by achieving collision avoidance.

The augmentation methods ultimately chosen by our framework are listed in Table III. From the experimental results, it can be observed that the augmentation methods which individually yield positive impacts on the algorithm, as listed in Table II, are also frequently selected by our framework. This further validates the effectiveness of our approach. Additionally, it's noteworthy that different tasks converge to different augmentation combinations.

D. Real-World Evaluation

To empirically validate the efficacy of our framework, we employed a straightforward sim-to-real approach [28], deploying policies directly trained in Webots simulation onto actual E-puck robots. The experiments were conducted in a 2m x 2m indoor area and utilized motion capture systems for accurate localization. Our results demonstrate a substantial improvement in MARL performance, specifically reducing high-risk states (defined as inter-agent distances less than 5 cm) from 8.6% to 5.1%. These findings indicate that our approach effectively enhances the sim-to-real capabilities of MARL algorithms. Although our training eventually converged on specific augmentation techniques, the UCB algorithm in the early stages of training attempted to select noise-based data augmentation methods, which likely contributed to the improvement in sim-to-real capabilities.

VII. Conclusions

In this paper, we presented AdaptAUG, a theoretically grounded adaptive data augmentation framework specifically designed to address the challenges of low sample efficiency and unstable training in multi-agent reinforcement learning (MARL). We theoretically show that a lower augmentation sensitivity for a given augmentation results in a more precise bound on Q-value estimation during data augmentation for MARL. The empirical evidence indicates that AdaptAUG significantly improves performance across a variety of settings, emphasizing its wide applicability and potential in advancing MARL technology. Moreover, the validation of our framework on real robotic platforms serves as evidence of its practical effectiveness, showcasing its adaptability to real-world scenarios and its promise for deployment in practical applications. Moving forward, we plan to investigate the potential applications of our framework in other areas of reinforcement learning[29], [30].

Acknowledgments

This work was supported by the National Key Research and Development Program of China (Grant No. 2022ZD0116401), the National Natural Science Foundation of China (Grant No. 62306023), and the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2023ZX-20).

REFERENCES

[1] C. Yu, X. Wang, and Z. Feng, “Coordinated multiagent reinforcement learning for teams of mobile sensing robots,” in *Proceedings of the 18th international conference on autonomous agents and multiagent systems*, 2019, pp. 2297–2299.

[2] B. Singh, R. Kumar, and V. P. Singh, “Reinforcement learning in robotic applications: a comprehensive survey,” *Artificial Intelligence Review*, pp. 1–46, 2022.

[3] F. L. Da Silva and A. H. R. Costa, “A survey on transfer learning for multiagent reinforcement learning systems,” *Journal of Artificial Intelligence Research*, vol. 64, pp. 645–703, 2019.

[4] X. Yu, W. Wu, P. Feng, and Y. Tian, “Swarm inverse reinforcement learning for biological systems,” in *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2021, pp. 274–279.

[5] X. Yu, R. Shi, P. Feng, Y. Tian, J. Luo, and W. Wu, “Esp: Exploiting symmetry prior for multi-agent reinforcement learning,” in *ECAI 2023*, 2023, pp. 2946–2953.

[6] R. Raileanu, M. Goldstein, D. Yarats, I. Kostrikov, and R. Fergus, “Automatic data augmentation for generalization in reinforcement learning,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 5402–5415, 2021.

[7] Z. Ye, Y. Chen, X. Jiang, G. Song, B. Yang, and S. Fan, “Improving sample efficiency in multi-agent actor-critic methods,” *Applied Intelligence*, pp. 1–14, 2021.

[8] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, “Reinforcement learning with augmented data,” *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[9] K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman, “Quantifying generalization in reinforcement learning,” in *International Conference on Machine Learning*, 2019, pp. 1282–1289.

[10] M. M. Drugan and A. Nowe, “Designing multi-objective multi-armed bandits algorithms: A study,” in *The 2013 international joint conference on neural networks (IJCNN)*. IEEE, 2013, pp. 1–8.

[11] D. Yarats, I. Kostrikov, and R. Fergus, “Image augmentation is all you need: Regularizing deep reinforcement learning from pixels,” in *International Conference on Learning Representations*, 2020.

[12] Y. Lin, J. Huang, M. Zimmer, Y. Guan, J. Rojas, and P. Weng, “Invariant transform experience replay: Data augmentation for deep reinforcement learning,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6615–6622, 2020.

[13] F. Amadio, A. Colomé, and C. Torras, “Exploiting symmetries in reinforcement learning of bimanual robotic tasks,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1838–1845, 2019.

[14] M. Schwarzer, A. Anand, R. Goel, R. D. Hjelm, A. Courville, and P. Bachman, “Data-efficient reinforcement learning with self-predictive representations,” *arXiv preprint arXiv:2007.05929*, 2020.

[15] W. Zhao, J. P. Queralta, and T. Westerlund, “Sim-to-real transfer in deep reinforcement learning for robotics: a survey,” in *2020 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2020, pp. 737–744.

[16] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, “Sim-to-real transfer of robotic control with dynamics randomization,” in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[17] D. Yarats, I. Kostrikov, and R. Fergus, “Image augmentation is all you need: Regularizing deep reinforcement learning from pixels,” in *International conference on learning representations*, 2020.

[18] C. Lu, B. Huang, K. Wang, J. M. Hernández-Lobato, K. Zhang, and B. Schölkopf, “Sample-efficient reinforcement learning via counterfactual-based data augmentation,” *arXiv preprint arXiv:2012.09092*, 2020.

[19] X. Yu, R. Shi, P. Feng, Y. Tian, S. Li, S. Liao, and W. Wu, “Leveraging partial symmetry for multi-agent reinforcement learning,” *arXiv preprint arXiv:2401.00167*, 2023.

[20] S. Chen, G. Liu, Z. Zhou, K. Zhang, and J. Wang, “Robust multi-agent reinforcement learning method based on adversarial domain randomization for real-world dual-uav cooperation,” *IEEE Transactions on Intelligent Vehicles*, 2023.

[21] M. L. Littman, “Markov games as a framework for multi-agent reinforcement learning,” in *Machine learning proceedings 1994*, 1994, pp. 157–163.

[22] A. Durand, C. Bordet, and C. Gagné, “Improving the pareto ucb1 algorithm on the multi-objective multi-armed bandit,” in *Workshop of the 27th Neural Information Processing (NIPS) on Bayesian Optimization*, 2014.

[23] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, “Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning,” in *International Conference on Machine Learning*, 2018, Conference Proceedings, pp. 4295–4304.

[24] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, “Multi-agent actor-critic for mixed cooperative-competitive environments,” *arXiv preprint arXiv:1706.02275*, 2017.

[25] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, “The surprising effectiveness of ppo in cooperative, multi-agent games,” *arXiv preprint arXiv:2103.01955*, 2021.

[26] I. Mordatch and P. Abbeel, “Emergence of grounded compositional language in multi-agent populations,” *arXiv preprint arXiv:1703.04908*, 2017.

[27] P. Feng, X. Yu, J. Liang, W. Wu, and Y. Tian, “Mact: Multi-agent collision avoidance with continuous transition reinforcement learning via mixup,” in *International Conference on Swarm Intelligence*. Springer, 2023, pp. 74–85.

[28] C. De Souza, R. Newbury, A. Cosgun, P. Castillo, B. Vidolov, and D. Kulić, “Decentralized multi-agent pursuit using deep reinforcement learning,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4552–4559, 2021.

[29] T. Peng, W. Wu, H. Yuan, Z. Bao, Z. Pengrui, X. Yu, X. Lin, Y. Liang, and Y. Pu, “Graphrare: Reinforcement learning enhanced graph neural network with relative entropy,” *arXiv preprint arXiv:2312.09708*, 2023.

[30] S. Li, J. Guo, J. Xiu, X. Yu, J. Wang, A. Liu, Y. Yang, and X. Liu, “Byzantine robust cooperative multi-agent reinforcement learning as a bayesian game,” *arXiv preprint arXiv:2305.12872*, 2023.