Received 28 July 2022, accepted 8 August 2022, date of publication 16 August 2022, date of current version 25 August 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3198981

RESEARCH ARTICLE

CuMARL: Curiosity-Based Learning in Multiagent Reinforcement Learning

DEVARANI DEVI NINGOMBAM[1], **BYUNGHYUN YOO**[2], **HYUN WOO KIM**[2], **HWA JEON SONG**[2], **AND SUNGWON YI**[2]

[1]Department of Informatics, School of Computer Science, University of Petroleum and Energy Studies (UPES), Dehradun, Uttarakhand 248007, India
[2]Electronics and Telecommunications Research Institute (ETRI), Daejeon 34129, South Korea

Corresponding authors: Devarani Devi Ningombam (devaraninin@gmail.com) and Sungwon Yi (sungyi@etri.re.kr)

This work was supported in part by the Electronics and Telecommunications Research Institute (ETRI) funded by the Korean Government, in part by the Core Technology Research for Self-Improving Integrated Artificial Intelligence System under Grant 22ZS1100, in part by the Exploratory Research for Human-Agent Collaboration in Multi-Agent Games under Grant 19YE1410, and in part by AFOSR under Grant FA2386-19-1-4020.

ABSTRACT In this paper, we propose a novel curiosity-based learning algorithm for Multi-agent Reinforcement Learning (MARL) to attain efficient and effective decision-making. We employ the centralized training with decentralized execution framework (CTDE) and consider that each agent has knowledge of the prior action distribution of others. To quantify the difference in agents' knowledge, curiosity, we introduce conditional mutual information (CMI) regularization and use the amount of information for updating decision-making policy. Then, to deploy these learning frameworks in a large-scale MARL setting while acquiring high sample efficiency, we consider a Kullback-Leibler (KL) divergence-based prioritization of experiences. We evaluate the effectiveness of the proposed algorithm in three different levels of StarCraft Multi Agent Challenge (SMAC) scenarios using the PyMARL framework. The simulation-based performance analysis shows that the proposed technique significantly improves the test win rate compared to various state-of-the-art MARL benchmarks, such as the Optimistically Weighted Monotonic Value Function Factorization (OW_QMIX) and Learning Individual Intrinsic Reward (LIIR).

INDEX TERMS Multi-agent reinforcement learning, curiosity, conditional mutual information, prioritized experience replay.

I. INTRODUCTION

In spite of unprecedented success in many applications, uncertain and non-stationary environments pose a serious challenge in finding an efficient solution in multi-agent systems. The nature of the complexity comes from constant interactions among agents, which are viewed as a part of environments in Multi-agent Reinforcement Learning (MARL).

A common approach adopted widely across various modern MARL techniques is the greedy algorithm, where agents are trained to maximize their individual rewards [1], [11], [17], [18]. Although several works proposed to train the agents to learn high-level strategic behaviors such as individual agent's sacrifice for the greater good [2], [3], [4], [5], [6], an agent's action selection is still based on a technique called epsilon greedy. For example, an agent mostly chooses actions that are expected to give maximum rewards or selects random actions that have not been explored before in search for a possible better reward. Thus, an agent's exploration solely relies on small randomness, and the inefficient exploration often contributes to a longer training time or results in suboptimal solutions. In addition, each agent explores the problem spaces independently, and the exploration can be significantly overlapped.

To address these problems, various techniques have been proposed based on intrinsic motivation [7], counters [8], and curiosity [9] concepts. However, these approaches require additional structure or complicated parameter optimizations. [10], [11] were proposed in a coordinated exploration concepts based on a shared reply buffer. Under these approaches,

The associate editor coordinating the review of this manuscript and approving it for publication was Qiang Li.

the agents simultaneously explore a wide range of problem spaces using a set of policies, whereas learning a good policy requires extensive sharing of information through communication and training time increases with the growing action spaces.

Recently a hybrid learning framework for distributed transmission (HDT) in MARL was presented in [12]. The authors introduced cloud-edge computing to attain network-wide low-delay performance. The novelty of this paper is in the reward function design, which considers the instantaneous QoS constraint utilizing the Lagrange technique. The proposed strategy produces a local maximum rather than a global maximum when building reward functions using the Lagrange multiplier technique.

In social science, observation is known as a primary source of the intellectual process of acquiring knowledge. Young children often observe and imitate their older siblings, and such activities are known to contribute to their cognitive development [13], [14]. In this paper, we propose a novel curiosity-based learning technique inspired by such a concept. The idea of curiosity-based learning is to exploit peer agents' acquired knowledge. Using the knowledge of other agents can affect an agent's learning process in positive or negative ways. However, the agent can quickly investigate such cases via their various experiences, and thus the training time can be significantly reduced.

As children learn from their siblings' knowledge, often more mature than them, an agent exploits others' knowledge if it is different from the agent itself. In real time strategy applications, an agent exploits others' knowledge if it is different from their own. To quantify the difference between agents' knowledge, in this paper, we introduce the curiosity concept defined by conditional mutual information (CMI). The main significance of this method is that it reduces the uncertainty in the knowledge of the target agent's decision-making policy, given the knowledge of other agents. By doing so, each agent manages the long-term visitation counts, which are in the form of the probability distribution of state-actions.

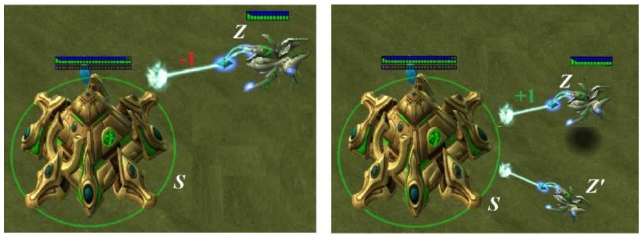


FIGURE 1. **An example of how an agent exploits other's knowledge to take a wise decision. Game Scenario 1 (left): Inexperienced agent Z shoots agent S but missed the target and get penalty as -1. Game Scenario 2 (right): Inexperienced agent Z shoots agent S by observing behaviors of experienced agent Z' and get a reward as $+1$.**

Moreover, to quantify the interdependence between probability distributions, we compute the CMI and use this quantity to update an agent's decision-making policy. An example of curiosity-based learning is presented in Figure 1. We apply this curiosity concept in experience sampling as well. Reply buffer or prioritized reply buffer are widely applied techniques to address the sample efficiency problem in MARL. However, the prioritized technique only selects the samples based on an agent's maximum expected reward, so the agent's experience can be biased. To address this problem, we considered a Kullback-Leibler (KL) divergence-based prioritization of experiences that helps an agent acquire more diverse experiences and, in turn, achieve a better learning curve.

To the best of our knowledge, the most relevant approach to ours is [15], where curiosity-based exploration for episodic MARL is presented. Though both approaches share the similar name of "curiosity", the definitions are quite different. Authors in [15] used prediction errors of individual Q-values as intrinsic reward (called curiosity) for coordinated exploration and employed episodic memory to exploit explored informative experience to enhance policy training. On the other hand, curiosity is defined by CMI to measure the difference in policy learned in this paper. The performance of [15] can be limited due to the lack of adaptable exploration, which can significantly affect the resilience of the system.

Our proposed technique was evaluated in the StarCraft Multi-Agent Challenge (SMAC) [16], which is one of the most widely used simulation testbeds for multi-agent systems, and compared against previous techniques, including [3] and [17]. In the simulation-based experiments, the proposed scheme showed improved performance overall. In particular, our simulation results demonstrate the efficiency of curiosity-based learning in MARL by significantly accelerating the convergence speed. On the super-hard game scenarios, the test win rate of our proposed scheme is 1.7 and 1.2 times better than the benchmarks [3] and [17], respectively. The technical contributions of this paper are summarized as follows:

- We proposed a novel curiosity-based reward individualization for fully cooperative agents in a partially observable environment MARL. To quantify the difference in agents' knowledge, curiosity, we introduce conditional mutual information regularization (CMIR) and use the amount of information for updating decision-making policy.
- We proposed the Kullback-Leibler (KL) divergence-based prioritization of experiences to deploy our proposed learning frameworks in a large-scale MARL, which can acquire a high sample efficiency.

The rest of the paper is organized as follows. Section II describes the related work in MARL. Thereafter, the proposed technique is presented in Section III. Section IV explains the performance results and discussion. Finally, we conclude the paper in Section V.

II. RELATED WORK

In this section, we briefly summarize and discuss previous techniques proposed to address the issues in MARL. In reinforcement learning environments, curiosity has been extensively studied as a means of accelerating learning. A mixing neural network is proposed in [1] to connect the joint action-value function and decompose the action-value functions. Each action-value function is called an individual utility function under a monotonic constraint. QMIX is learned by minimizing the squared temporal difference error of the joint action-value function. Coordination exploration problems are discussed in [3]. Existing work exploited curiosity-based learning techniques for single-agent learning environments [2], [9], [19], which focus on the individual agent's curiosity-based reward for learning tasks independently. However, due to the stochastic nature of the environment, a wide range of learning is restricted. Some similar methods are the intrinsically motivated approach as successful learning bonuses are presented in [4], [5], [20], which alleviate hard learning problems. Using curiosity as a form of intrinsic reward, finding an agent's action that has the highest impact on other agents' actions is the main aim of this paper. In a similar direction, another learning technique [21] focuses on agents' coordination to attain better learning of the action spaces. The problem is that the agents' action-spaces are predicted directly without taking environmental state characteristics into account. Thus, the above learning techniques cannot extend their applicability in non-stationary MARL settings, where the policies of agents are always changing.

Many researchers focus on the concurrent learning framework to accelerate efficient learning. A variable filtration method is applied to measure the reward loss due to learning [22]. This paper propagates multiple agents' parametric distributions to attain fast learning of the state-action space. Scalable coordinated exploration [23] considers unrestricted communication as a means to coordinate the agent's behaviors. This method fails to provide significant performance improvement in large action space real-world applications due to the high cost of communication. The main difference between all these approaches and our proposed approach is that our proposed approach addresses the multi-agent learning problem by coordinating agents' control policies using state transition characteristics.

Lately, the inequality aversion method has been used in MARL to encourage fast and successful learning through cooperation [24]. The intertemporal social dilemma concept is analyzed by illustrating the trade-off between global reward and an intrinsically motivated individual agent's reward. Nonetheless, this approach successfully benefits the self-interested agents but fails to have negative contributions to the global actions as the training timestep increases. [25] proposes to avoid some unwanted actions that can affect the performance of multi-agent systems and thus reduce training time. However, it cannot overcome the overestimation biases in multi-agent systems due to instability and high variance.

[26], [27], [28], [29] propose sampling optimization techniques to overcome the overestimation bias. Rather than selecting experiences from the replay buffer randomly, sampling through prioritizing a small section of experiences in the replay buffer is presented. In contrast to these approaches, our approach proposes a more practical learning framework with less uncertainty in MARL action selection in partially observable decentralized Markov Decision Processes. To encourage successful and fast learning, a novel

TABLE 1. List of acronyms used in the paper.

Acronyms	Abbreviations
MARL	Multi-agent Reinforcement Learning
CTDE	Centralized Training and Decentralized Execution
CMI	Conditional Mutual Information
KL	Kullback-Leibler
SMAC	StarCraft Multi Agent Challenge
OW_QMIX	Optimistically Weighted Monotonic Value Function Factorization
CuE	Episodic Multi-agent Reinforcement Learning with Curiosity-Driven Exploration
LIIR	Learning Individual Intrinsic Reward
HDT	Hybrid Learning Framework for Distributed Transmission
CMIR	Conditional Mutual Information Regularization
MDP	Markov Decision Process
IMM	Intrinsically Motivated Model
MAAC	Multi-agent Actor-critic
PER	Prioritized Experience Replay

TABLE 2. Comparison between proposed technique and highly relevant existing techniques.

Algorithm	Type of Learning	Proposed Method	Simulation Environment
QMIX [1]	Value-based	Monotonic value function factorization	SMAC [16]
LIIR [3]	Policy-based	Learning individual Intrinsic reward	SMAC [16]
IMM [5]	Policy-based	Intrinsically motivated model-based learning	Aldebaran Nao robot [5]
MAAC [11]	Actor-critic	Hierarchical graph attention-based actor-critic	Predator-Prey [12]
HDT [12]	Extended Actor-critic	Distributed transmission using cloud-edge computing	Cloud-edge system [12]
CuE [15]	Actor-critic	Curiosity-based Episodic Exploration	SMAC [16]
OW_QMIX [17]	Value-based	Weighted monotonic value function factorization	SMAC [16]
CuMARL	**Policy-based**	**Curiosity-based exploration using KL-divergence**	**SMAC [16]**

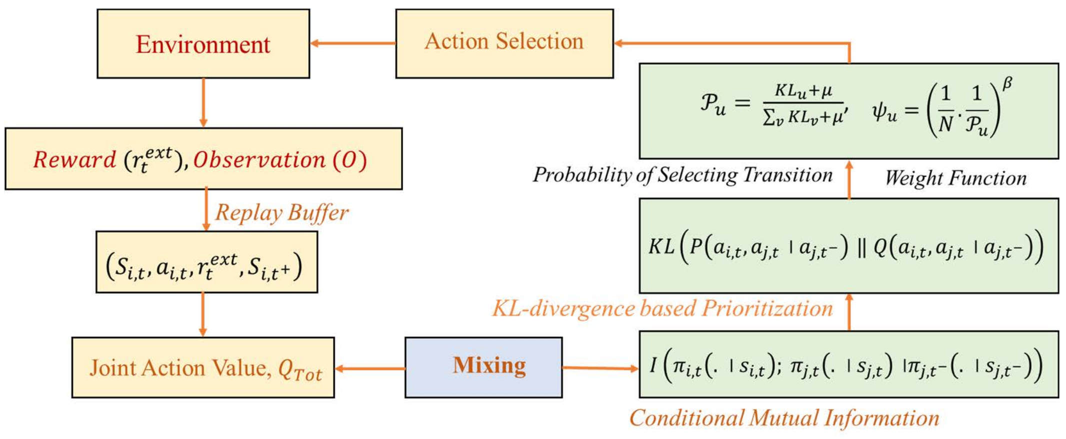

Environment
Action Selection
$Reward\ (r_t^{ext}), Observation\ (O)$
Replay Buffer
$(S_{i,t}, a_{i,t}, r_t^{ext}, S_{i,t^+})$
Joint Action Value, Q_{Tot}
Mixing
$\mathcal{P}_u = \frac{KL_u+\mu}{\sum_v KL_v+\mu}, \quad \psi_u = \left(\frac{1}{N}\cdot\frac{1}{\mathcal{P}_u}\right)^\beta$
Probability of Selecting Transition
Weight Function
$KL\left(P(a_{i,t}, a_{j,t} \mid a_{j,t^-}) \parallel Q(a_{i,t}, a_{j,t} \mid a_{j,t^-})\right)$
KL-divergence based Prioritization
$I\left(\pi_{i,t}(. \mid s_{i,t}); \pi_{j,t}(. \mid s_{j,t}) \mid \pi_{j,t^-}(. \mid s_{j,t^-})\right)$
Conditional Mutual Information

FIGURE 2. Architecture of the proposed technique.

learning strategy is analyzed based on the self-curiosity and others' long-term visitation values, and to accelerate learning of diverged and important samples from a large buffer size, prioritization using the moving average with time window technique is introduced in the replay buffer. The acronyms used in the paper are shown in Table 1. Finally, as a summary of the related works, we presented a comparison report in Table 2.

III. PROPOSED TECHNIQUE

Based on the discussions in the previous section, we propose a novel learning mechanism with high sample efficiency. The proposed method contains two main modules. One is the curiosity-based learning with intrinsic reward via manipulating agents' policies relying on others' behaviors. The other is the KL divergence-based prioritized experience replay to attain a better learning curve. The architecture of the proposed method is presented in Figure 2.

A. NAÏVE CURIOSITY-BASED LEARNING IN MARL

We consider a multi-agent environment with decentralized partially observable Markov Decision Processes (Dec-PoMDP) allowing centralized training and decentralized execution framework, which is given by a tuple ($\mathcal{G}$, S, A, P, r, γ, O, P_0), where $\mathcal{G} = (1, 2, .., n)$ represents the set of n agents, $S = (S_1, S_2, \ldots, S_n)$ represents the set of state, $A = (A_1, A_2, \ldots, A_n)$ represents the action space of the agent, $\mathbf{r}$ is the global reward, and $\gamma\epsilon[0, 1]$ is the discount factor, $O = (O_1, O_2, \ldots, O_n)$ represents the set of finite observations, and P_0 is the initial state distribution. We assume n independent parametrized policies π_{θ_i} where $\theta = (\theta_1, \theta_2, \ldots, \theta_n)$. At each timesteps, each agent. Then, the stochastic process at state A_n selects an action $a_i \in A$ at state s, forming a joint $\boldsymbol{a} \in \boldsymbol{A} \equiv S_t$ is defined as $\chi(o_t|s_t)$ and its transition function $P(s'|\text{s, a}) : \text{S} \times \text{A} \times \text{S} \rightarrow [0, 1]$. In the ccc coo MARL environment, each agent i aims to maximize its own total expected return $J^{ext}(\theta_i) = \text{E}_{s,a}[R^{ext}]$ where $R^{ext} = \sum_{t=0}^{T} \gamma^t.r^{ext}$. Therefore, the policy gradient with respect to the parametrized policy parameter θ_i is defined as

$$\nabla_{\theta_i} J^{ext}(\theta_i) = \text{E}_\pi[\nabla_{\theta_i} log_\pi(s, a, \theta_i) A^{ext}(s, a)] \tag{1}$$

where $A^{ext}(s, a)$ is the advantage function. In the Dec-PoMDP environment, the state information available to the learning agents is restricted, and training of multiple agents at a time is more difficult. In this scenario, a greedy action selection algorithm helps agents to discover promising behaviors quickly. However, in MARL, multiple agents not only interact with the environment, but also among themselves. The greedy action selection algorithm fails to discover promising action space for MARL and thereby results in insufficient exploration. We believe that coordinating agents' policies in a MARL setting can accelerate the learning of good policies and improve performance across several real-time strategy applications.

In this work, we focus on handling the policy coordination technique through agents' curiosities to perform predefined tasks. Intuitively, coordinating multiple agents' control policies requires knowledge of each agent's state transition behavior, and to promote efficient coordination, we need to observe the behaviors of the agents. In order to foster coordination, we analyze long-term visitation counts of agents as probability distributions of actions and measure the degree of interdependence between two random distributions using CMI regularization. In our proposed algorithm, the value of the CMI acts as a strategy to manipulate agents' policies and thereby update the decision-making. Thus, the agent can make smart decisions at each timestep and explore the useful environment that results in high reward without overlapping with each other.

Considering the above-mentioned learning framework, in this method we consider the long-term visitation counts explicitly in exploration to increase the possibility of finding good moves. Thus, each agent can look up the long-term visitation counts of others and will try to anticipate other agents' hidden motivations ("curiosity") to perform the given tasks. Moreover, to understand the value iteration propagation details of the past visitations, we use the moving average with time window technique.

Suppose that, at time t, a target agent i with a decision-making policy $\pi_{i,t}(. \mid s_{i,t})$ tries to estimate and decide an action-value of a state by inferring other agents' j_s current decision-making policy $\pi_{j,t}(. \mid s_{j,t})$ and the long-term visitation value $\pi_{j,t^-}(. \mid s_{j,t^-})$. Each agent under this learning framework has control over its actions and can change their strategy to foster efficient exploration. In this work, we consider a neural network that has three inputs, namely, the target agent's decision-making policy (self-curiosity to perform given tasks), other agents' current policies and past decision-making policies (long-term visitation values), and outputs an action-value estimate of the target agent. Upon introducing this concept, a new reward function can be defined as

$$r_{i,t}^{new} = r_t^{ext} + \delta . \sum\nolimits_{i,j\in[1,2,\ldots,n],i\neq j} \times I\left(\pi_{i,t}\left(.\mid s_{i,t}\right) ; \pi_{j,t}\left(.\mid s_{j,t}\right) \mid \pi_{j,t^-}\left(.\mid s_{j,t^-}\right)\right) \tag{2}$$

where $I\left(\pi_{i,t}\left(.\mid s_{i,t}\right) ;\pi_{j,t}\left(.\mid s_{j,t}\right) \mid \pi_{j,t^-}\left(.\mid s_{j,t^-}\right)\right)$ is the CMI between policies of agents i and j at state s_t conditioned to policy of agent j at state s_{i,t^-}. $\pi_{i,t}\left(.\mid s_{i,t}\right)$ and $\pi_{j,t}\left(.\mid s_{j,t}\right)$ are the policies of agents i and j at time t, respectively, and $\pi_{j,t^-}\left(.\mid s_{j,t^-}\right)$ is the policy of agent j at time t$^-$, and δ is the weighting term which is always positive. Corresponding discounted surrogate reward for agent i is defined as

$$J^{new}\left(\theta_i\right) = \mathrm{E}\left[\sum\nolimits_{t=0}^{T} \gamma_t . r_{i,t}^{new}\right] \tag{3}$$

$$J^{new}\left(\theta_i\right) = \mathrm{E}\left[\sum\nolimits_{t=0}^{T} \gamma_t . \left(r_t^{ext} + \delta . \sum\nolimits_{i,j\in[1,2,\ldots,n],i\neq j} \times I\left(\pi_{i,t}\left(.\mid s_{i,t}\right) ; \pi_{j,t}\left(.\mid s_{j,t}\right) \mid \pi_{j,t^-}\left(.\mid s_{j,t^-}\right)\right)\right)\right] \tag{4}$$

The system dynamics at time t is defined as

$$P_t = P\left(s_{t^+} \mid s_j, a_j\right) \tag{5}$$

Thus, we compute the CMI between two agents i and j from the perspective of agent i as

$$I\left(\pi_{i,t}\left(.\mid s_{i,t}\right) ; \pi_{j,t}\left(.\mid s_{j,t}\right) \mid \pi_{j,t^-}\left(.\mid s_{j,t^-}\right)\right) = P\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right) + P(a_{i,t}) \tag{6}$$

where

$$P\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right) = \mathrm{E}_{P\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right)}\left[log \frac{Q\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right)}{P\left(a_{i,t}\right)}\right] \tag{7}$$

$$P\left(a_{i,t}\right) = \mathrm{E}_{P\left(a_{i,t}\right)}\left[KL\left(P\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right) \parallel Q\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right)\right)\right] \tag{8}$$

Eq. (6) signifies that the CMI between the target agent i's decision making policy and the influenced agent j's decision making policy w.r.t the influenced agent j's previous timestep policy is derived as the sum of the joint probability function of actions of agents i and j and marginal probability function of the target agent i. The joint probability function, eq. (7), is derived as the expected value of the logarithmic value of the variational distribution of target agent i's action and the influenced agent j's action w.r.t the influenced agent j's previous timestep action to the marginal probability function of the agent i. The marginal probability distribution of a target agent i is expressed as the expectation of the KL-divergence between the joint probability distribution function and the variational distribution function of actions of agents i and j conditioned to the agent j's previous timestep action.

Substituting eqs. (7) and (8) to (6), we get,

$$I\left(\pi_{i,t}\left(.\mid s_{i,t}\right) ; \pi_{j,t}\left(.\mid s_{j,t}\right) \mid \pi_{j,t^-}\left(.\mid s_{j,t^-}\right)\right) = \mathcal{H}\left(\pi_{i,t}\left(.\mid s_{i,t}\right)\right) + logQ\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right) \tag{9}$$

where $\mathcal{H}\left(\pi_{i,t}\left(.\mid s_{i,t}\right)\right)$ is the policy entropy term of agent i, and $Q\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right)$ is the variational distribution to approximate the conditional distribution $P\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right)$.

Eq. (9) signifies that CMI between the between the target agent i's decision making policy and influenced agent j's decision making policy conditioned to influenced agent j's previous timestep policy is derived as the sum of marginal entropy of target agent i's decision making policy and logarithmic of variational distribution of target agent i's action and influenced agent j's action conditioned to influenced agent j's previous timestep action. We finally have the resulting policy gradient update of agent i is calculated as

$$J^{new}\left(\theta_i\right) = \mathrm{E}\left[R_i^{new}\right] \tag{10}$$

$$J^{new}\left(\theta_i\right) = \mathrm{E}\left[\sum\nolimits_{t=0}^{T} \gamma_t . \left(r_t^{ext} + \delta . \sum\nolimits_{i,j\in[1,2,\ldots,n],i\neq j} \times I\left(\pi_{i,t}\left(.\mid s_{i,t}\right) ; \pi_{j,t}\left(.\mid s_{j,t}\right) \mid \pi_{j,t^-}\left(.\mid s_{j,t^-}\right)\right)\right)\right] \tag{11}$$

$$J^{new}\left(\theta_i\right) = \mathrm{E}\left[\sum\nolimits_{t=0}^{T} \gamma_t . \left(r_t^{ext} + \delta . \sum\nolimits_{i,j\in[1,2,\ldots,n],i\neq j} \times \mathcal{H}\left(\pi_{i,t}\left(.\mid s_{i,t}\right)\right) + logQ\left(a_{i,t}, a_{j,t} \mid a_{j,t^-}\right)\right)\right] \tag{12}$$

For optimization, a Bi-level optimization technique is considered, which is one of the promising techniques that allows for efficient search of a large number of policy parameters. A Bi-level optimization problem consists of two levels: upper-level and lower-level optimization problems. Each optimization level has its own target functions and constraints. The goal of our strategy is to find the extrinsic policy parameters that maximize the surrogate reward function, whereas the upper-level optimization problem's goal is to find the curiosity-based intrinsic reward parameters that maximize the predicted extrinsic reward function. The upper-level optimization has full access to the lower-level optimization. Then, we formulate a bi-level optimization problem as illustrated below.

$$\max_{\theta,\zeta} \; J^{ext}(\zeta) \tag{13}$$

$$\text{such that } \theta_i = \text{argmax } J_i^{new}(\theta, \zeta), \quad \forall i \in [1, 2, \ldots, n] \tag{14}$$

where ζ is the curiosity-based intrinsic reward parameter and θ is the policy parameter. It should be noted that the benefit of introducing bi-level optimization problem in our approach is that, by learning an individual distinct curiosity-based intrinsic reward and also anticipating the curiosity-based intrinsic rewards of others per timestep, we can diversely stimulate the decision-making and improve overall system performance. The pseudo code of module one is shown in Algorithm 1.

B. KL DIVERGENCE-BASED PRIORITIZED EXPERIENCE REPLAY

In general, a system is said to be sample efficient if it can select the most significant experience from many samples of experiences in the replay buffer and improve its control policy. Prioritized experience replays (PER) is a technique that boosts the training efficiency by giving weight to the samples so that significant ones are selected more frequently for training. In this paper, we consider a KL-divergence based prioritization technique to characterize which transitions are more beneficial to the learning system and their effectiveness on MARL applications. The KL- divergence is a technique to define the divergence of one probability distribution from another and, thereby, measures how similar the samples that the target policy may have been. The reason for combining the curiosity-based algorithm with the prioritization is that we want to combine information theory and sample distribution algorithms in Multi-Agent Reinforcement Learning to address the learning speed problem in a large-action space scenario.

Specifically, at each time step, experience, in the form of a tuple consisting of state input, action taken, received reward, and next state, is stored in the replay. Then, assume a replay buffer memory, $B = (E_1, E_2, E_3, \ldots, E_N)$ having the most recent sample transitions accumulated by the control policy. To compute the priority, we consider a time-window and calculate the KL-divergence between transitions. Then, the moving average is applied to make it tractable for time series analysis. We observe a diverse range of transitions after calculating the divergence values in the replay buffer. It should be noted that in the traditional TD-error-based prioritization, the authors choose the samples with the highest TD-error. On the contrary, we select a range of samples through the moving average with the time window.

Finally, we sample the transitions according to the values of the KL-divergence such that the high-value transition acquires higher priority than others and we sample more frequently from the replay buffer at a fixed interval of time for use in learning. For example, we define a window size of 10. With this window size, we moved the series of transitions with

Algorithm 1 Curiosity-Based Learning in MARL

```
Step 1:  Initialize $\boldsymbol{\theta}$, $\boldsymbol{\zeta}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$
Step 2:  for $\boldsymbol{t} = \mathbf{0}$ to $\boldsymbol{T}$ do
Step 3:  Observe long-term visitation count of other agents
Step 4:  Compute conditional mutual information ($I$)
         between distributions $i$ and j as follows:
             $\boldsymbol{I}\left(\boldsymbol{\pi}_{i,t}(.|\boldsymbol{s}_{i,t});\boldsymbol{\pi}_{j,t}(.|\boldsymbol{s}_{j,t})\,|\,\boldsymbol{\pi}_{j,t^-}(.|\boldsymbol{s}_{j,t^-})\right)$
Step 5:  Compute proposed reward, such that
             $\boldsymbol{r}_{i,t}^{new} = \boldsymbol{r}_t^{ext} + \boldsymbol{\delta}.\sum_{i,j\in[1,2,\ldots,n],i\neq j} \times \boldsymbol{I}\left(\boldsymbol{\pi}_{i,t}(.|\boldsymbol{s}_{i,t});\boldsymbol{\pi}_{j,t}(.|\boldsymbol{s}_{j,t})\,|\,\boldsymbol{\pi}_{j,t^-}(.|\boldsymbol{s}_{j,t^-})\right)$
         where $\boldsymbol{\delta}$ is the scaling factor used to weight the
         influence, such that $\mathbf{0} \leq \boldsymbol{\delta} \leq \mathbf{1}$.
Step 6:  Compute policy gradient as:
             $\boldsymbol{J}^{new}(\boldsymbol{\theta}_i) = \mathbf{E}\left[\sum_{t=0}^{T} \boldsymbol{\gamma}_t.\boldsymbol{r}_{i,t}^{new}\right]$
Step 7:  Initialize replay buffer B with the most recent
         transitions
Step 8:  Considering the system dynamic, compute the new
         policy gradient function as:
             $\boldsymbol{J}^{new}(\boldsymbol{\theta}_i) = \mathbf{E}\Big[\sum_{t=0}^{T} \boldsymbol{\gamma}_t.\,(\boldsymbol{r}_t^{ext} + \boldsymbol{\delta} \cdot \sum_{i,j\in[1,2,\ldots,n],i\neq j} \mathcal{H}\left(\boldsymbol{\pi}_{i,t}(.|\boldsymbol{s}_{i,t})\right) + \boldsymbol{logQ}\left(\boldsymbol{a}_{i,t}, \boldsymbol{a}_{j,t}|\boldsymbol{a}_{j,t^-}\right))\Big]$
         where $\mathcal{H}\left(\boldsymbol{\pi}_{i,t}(.|\boldsymbol{s}_{i,t})\right)$ is the policy entropy and
         $\boldsymbol{Q}\left(\boldsymbol{a}_{i,t}, \boldsymbol{a}_{j,t}|\boldsymbol{a}_{j,t^-}\right)$ is the variational distribution
         to approximate the conditional distribution
         $\boldsymbol{P}\left(\boldsymbol{a}_{i,t}, \boldsymbol{a}_{j,t}|\boldsymbol{a}_{j,t^-}\right)$.
Step 9:  Update policy parameter ($\boldsymbol{\theta}$) and curiosity param-
         eter ($\boldsymbol{\zeta}$) with learning rate ($\boldsymbol{\alpha}$).
Step 10: Update the target networks
Step 11: end for
```

some values of probability backward by one timestep. This is because we desire to calculate the divergence between our observations as well as the others in a moving average to predict the next transition probability. Thus, the KL-divergence between probability distributions of agents i and j is defined as

$$KL\left(\mathcal{P}_{\pi_{i,t}}|\mathcal{P}_{\pi_{j,t}}\right) = \sum_{s_0,a_0,\ldots,s_T,a_T} \mathcal{P}_{\pi_{i,t}}(s_0, a_0, \ldots, s_T, a_T) \cdot \log\left(\frac{\mathcal{P}_{\pi_{i,t}}(s_0, a_0, \ldots, s_T, a_T)}{\mathcal{P}_{\pi_{j,t}}(s_0, a_0, \ldots, s_T, a_T)}\right) \tag{15}$$

Therefore, the state-action distributions for the agents i and j are

$$\mathcal{P}_{\pi_{i,t}}\left(s_t, a_t\right) = \mathcal{P}_{\pi_{i,0}}\left(s_t\right).\pi_{i,t}\left(a_t | s_t\right) \tag{16}$$

and

$$\mathcal{P}_{\pi_{j,t}}\left(s_t, a_t\right) = \mathcal{P}_{\pi_{j,0}}\left(s_t\right).\pi_{j,t}\left(a_t | s_t\right) \tag{17}$$

respectively, where $\mathcal{P}_{\pi_{i,0}}\left(s_t\right)$ and $\mathcal{P}_{\pi_{j,0}}\left(s_t\right)$ are the initial state distribution of agents i and j, respectively. Therefore, the final expression of the KL-divergence is expressed as:

$$KL\left(\mathcal{P}_{\pi_{i,t}} | \mathcal{P}_{\pi_{j,t}}\right) = \sum\nolimits_{j=0}^{J} \mathbb{E}_{\mathcal{P}_{\pi_{i,0}}}\left[KL\left(\pi_{i,t} | \pi_{j,t}\right)\right] \tag{18}$$

Now, we can estimate the value of the KL-divergence to determine the priorities. That means our experiences in the replay buffer are updated to $\left(s_{i,t}, a_{i,t}, r_{i,t}^{new}, s_{i,t^+}, KL(\mathcal{P}_{\pi_{i,t}} | \mathcal{P}_{\pi_{j,t}}) + \mu\right)$. Thus, we sample the experiences with high value of the KL-divergence more frequently than others. However, the KL-divergence based prioritization still have the bias issue. To overcome this issue, we consider importance sampling weights of each transition as illustrated below:

$$\psi_u = \left(\frac{1}{N} \cdot \frac{1}{\mathcal{P}_u}\right)^{\beta} \tag{19}$$

such that

$$\mathcal{P}_u = \frac{KL_u + \mu}{\sum_v KL_v + \mu}, \quad \mu > 0, \ (u, v) \in \mathbb{Z} \cdot u \neq v \tag{20}$$

where $\mathbb{Z}$ is the number of sample transitions, μ denotes probability of selection, and β is the parameter that controls how much prioritization to apply during sampling. The pseudocode for the proposed prioritization technique is presented in Algorithm 2. The reason for combining our curiosity-based algorithm with prioritization is that we want to combine information theory and the sample distribution algorithm in Multi-Agent Reinforcement Learning to address the learning speed problem in a large-action space scenario.

IV. PERFORMANCE RESULTS AND DIISCUSSION

In this section, we present the simulation environment and experimental results to analyze the performance of the proposed technique in the SC2 micromanagement simulation environment.

A. SIMULATION ENVIRONMENT

For the following experiments, we implemented the proposed scheme by considering a fully connected layer with 2-hidden layers, each of 128 hidden units. We used the ADAM optimizer with a learning rate of 0.0005 and the ReLU nonlinearity activation function for hidden layers.

The training method starts with the gathering of episodic data, followed by training the critic for each timestep and training the actor. We employed a target network that updates parameters every 100 training steps for the feed-forward centralized critics. All maps are considered for batch mode training with a batch size of 32 and a gamma of 0.9. We monitored 2 million timesteps for the simple scenarios, 5 million timesteps for the hard scenarios, and 10 million timesteps for the super-hard scenarios during training and testing.

Algorithm 2 KL-Divergence Based PER

```
Step 1:  Initialize replay buffer $B$ with the most recent transitions
Step 2:  Inside the replay buffer, consider a time-window and compute the KL-divergence value between transition tuples. The, apply moving average technique to tune the batch size
Step 3:  Observe $\boldsymbol{S_{i,t^+}}, \boldsymbol{a_{i,t^+}}$
Step 4:  Store transition $\left(\boldsymbol{S_{i,t}}, \boldsymbol{a_{i,t}}, \boldsymbol{r_{i,t}^{new}}, \boldsymbol{S_{i,t^+}}\right)$ in the replay buffer
Step 5:  Calculate the priority of transition $u$
             $\boldsymbol{p_{max} = \max_{u \in \mathbb{Z}} \left(KL\left(\mathcal{P}_{\pi_{i,t}} | \mathcal{P}_{\pi_{j,t}}\right) + \mu\right), \quad \mu > 0}$
         where $\boldsymbol{u} \in \mathbb{Z}$, $\mathbb{Z}$ is the number of transition tuples, $\boldsymbol{\mu}$ is the small positive constant guaranteeing that the transition has some non-zero probability of being selected.
Step 6:  Calculate the probability of selecting transition $\boldsymbol{u}$ from the replay buffer as
             $\boldsymbol{\mathcal{P}_u = \frac{KL_u + \mu}{\sum_v KL_v + \mu}, \quad \mu > 0, \ (u, v) \in \mathbb{Z} \cdot u \neq v}$
Step 7:  Calculate the weights function using importance sampling technique such that,
             $\boldsymbol{\psi_u = \left(\frac{1}{N} \cdot \frac{1}{\mathcal{P}_u}\right)^{\beta}}$
         where $\boldsymbol{\beta}$ is the parameter that controls how much prioritization to apply and $\boldsymbol{0 < \beta < 1}$
Step 8:  Update the replay buffer with priority-based transition as presented below:
             $\boldsymbol{\left(s_{i,t}, a_{i,t}, r_{i,t}^{new}, S_{i,t^+}, KL\left(\mathcal{P}_{\pi_{i,t}} | \mathcal{P}_{\pi_{j,t}}\right) + \mu\right)}$
Step 9:  Sample a trajectory that has highest value of the KL-divergence value with the parametrized policies $\left\{\boldsymbol{\pi_{\theta_1}, \pi_{\theta_2}, .., \pi_{\theta_n}}\right\}$
Step 10: Update weight factor $\boldsymbol{\psi_u}$
Step 11: Update policy parameter $(\boldsymbol{\theta})$ and curiosity parameter $(\boldsymbol{\zeta})$ with learning rate $(\boldsymbol{\alpha})$.
Step 12: Update the target networks
```

TABLE 3. SMAC Scenarios [16].

Scenario	Level	Limit
3m	easy	60
2s3z	easy	120
3s5z	easy	150
3s_vs_5z	hard	250
bane_vs_bane	hard	200
MMM	hard	150
27m_vs_30m	super-hard	180
MMM2	Super-hard	180
3s5z_vs_3s6z	Super-hard	170

TABLE 4. Parameters and Values used in the experiment.

Notation	Name	Values
α	learning rate	0.0005
γ	discount factor	0.9
ζ	intrinsic reward parameter	0.01
β	prioritization control parameter	0.3-0.8
-	target function update	0.05
-	batch size	32
-	test interval	20K
-	number of test episode	24
B	maximum size of replay buffer	1e6
-	target update interval	100

TABLE 5. Agents properties in StarCraft II.

Name of Agent	Hit Points	Shield Points
Marines(m)	45	10
Stalkers(s)	80	80
Zealots(z)	100	50
Colossi(c)	200	150

For performance analysis, we consider PyMARL learning framework build on *PyTorch* and compare the results with LIIR, OW_QMIX, and CuPG_TD_PCr (Curiosity-based policy gradient with a temporal difference (TD)-error based prioritization). The considered MARL benchmark algorithms are listed as:

1. *CuPG*, Naïve curiosity-based Learning in Multi-Agent Reinforcement Learning: In this algorithm, each agent can predict the action space of others by observing previous behaviors.
2. *LIIR*, Learning Individual Intrinsic Reward in Multi-Agent Reinforcement Learning [3]: In this algorithm, each agent has its own observation and allows sharing of the same policy among the agents.
3. *OW_QMIX*, Optimistically Weighted Monotonic Value Function Factorization for Deep Multi-Agent Reinforcement Learning [17]: This algorithm used exact weighting of the joint actions of the behavior policy without any approximations.

Our proposed algorithms are summarized as:

I. *CuPG_KL_PCr*, Curiosity-based Learning in Multi-Agent Reinforcement Learning with the KL Divergence based Prioritization: This algorithm samples the prioritized experiences from the replay buffer.

II. *CuPG_TD_PCr*, Curiosity-based Learning in Multi-Agent Reinforcement Learning with TD-error based Prioritization.

The simulation scenarios, hyperparameters with their values, and the agents' properties used in the experiments are listed in Table 3, 4, and 5 respectively.

B. SIMULATION RESULTS

In the experiments, the techniques described in the previous sub-section and the proposed scheme are evaluated five times under each scenario. To clearly capture the advantages of the proposed algorithm and the usefulness of the probability distribution calculation using the moving average time-window concept, we compared two different methods of prioritization of experiences in the reply buffer. We then conducted extensive simulations under scenarios with three difficulty levels to demonstrate the effectiveness of the proposed scheme. The results are presented in the following section.

1) EASY SCENARIO

In TABLE 2, 3m, 2s3z, and 3s5z are considered as the easy level category, where the number and the type of ally units and enemy units are the same. This feature provides an easier challenge for learning agents. For training and testing, we restricted episode lengths for 3m, 2s3z and 3s5z maps to 60, 120 and 150 timesteps, respectively. The results of all algorithms on the three easy maps are shown in Figure 3 and their corresponding summarized results of win rates at different timesteps are provided in Table 6.

2) HARD SCENARIO

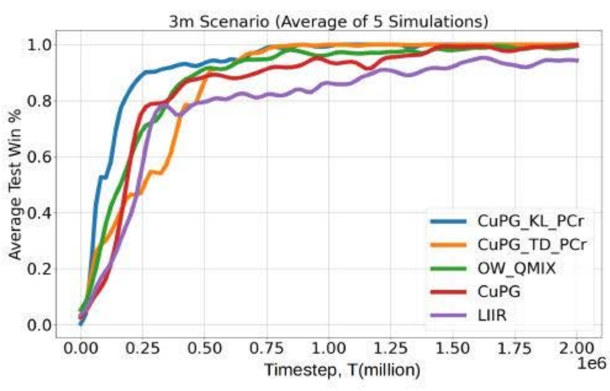
In hard scenarios, the types and number of units involved in the experiment can be asymmetric. Under these scenarios, learning focus fire strategy is essential in winning the game. Focus fire is a strategy often used by programmers where the ally units crossfire one selected enemy by one to quickly eliminate enemy units sequentially. These scenarios require a promising level of coordination between learning agents. Initially, due to hard game scenario, the proposed approaches show a little delay in convergence to optimal solution. As the timestep increases, test win rates increased. Overall, the test win rate results show that our proposed approaches are superior to that of LIIR and OW_QMIX. Specifically, at 2e6 timestep, our proposed algorithm attains 48%- and 100%-win rates as compare with 30% and 88% for OW QMIX and 12% and 33% for LIIR, respectively.

The test win performance for MMM is presented in Figure 4 and corresponding summarized results of win rates at different timesteps are provided in Table 7.

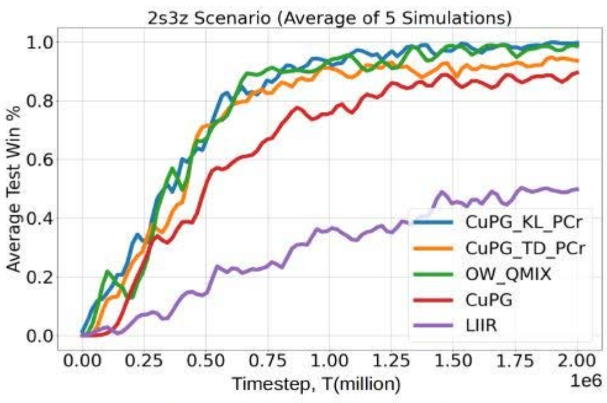
3) SUPER-HARD SCENARIO

In Figures 5, we present the individual test win rates for 27m_vs_30m, MMM2 and 3s5z_vs_3s6z scenarios. In 27m vs 30m, the number of the enemy units are more than that of the ally units. Thus, the ally units need a considerable policy coordination to win the game against the outnumbered enemy units. For MMM2, at less timestep, it fails to learn the good policy and results in insufficient exploration. As the timestep increases, MMM2 catch up the learning and converges to optimal solution. Overall, these scenarios are very hard to solve and longer training periods are required.

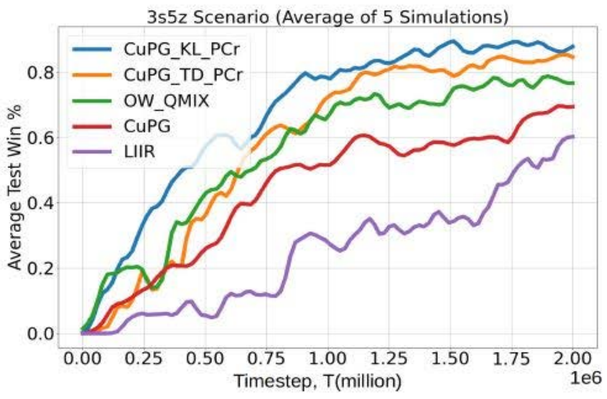
In above experiments, we observed that the proposed approaches solved the SMAC scenarios much faster than benchmark through efficient learning in MARL. Different values of win rate for all algorithms are listed in Table 8. Overall, the proposed algorithm achieves highest test-win



(a) Test win rate for 3m scenario.



(b) Test win rate for 2s3z scenario.



(c) Test win rate for 3s5z scenario.

FIGURE 3. Test win rates for easy scenarios.

TABLE 6. Test Win rates for easy scenarios at different timesteps.

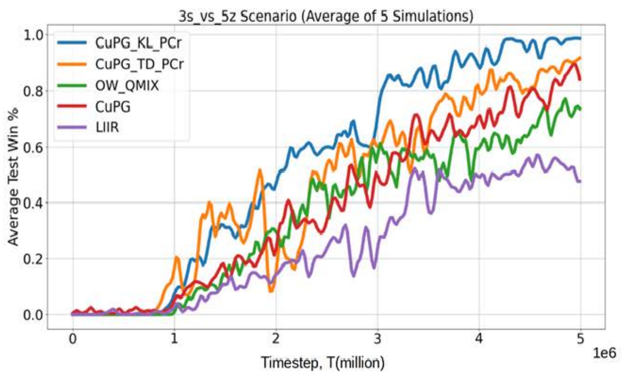
		Average Test Win Rate (%)				
Time Step (M)	Scenario	CuPG_KL_PCr	CuPG_TD_PCr	OW_QMIX	CuPG	LIIR
1.0	3m	**99.2**	99.9	97.3	94.3	86.6
	2s3z	**92.6**	91.1	94.2	75.6	36.2
	3s5z	**78.3**	72.2	66.8	51.5	27.9
2.0	3m	**100**	100	99.8	99	94.5
	2s3z	**99.8**	93.5	98.5	89.6	49.7
	3s5z	**99.1**	84.4	76.8	69.2	60.4

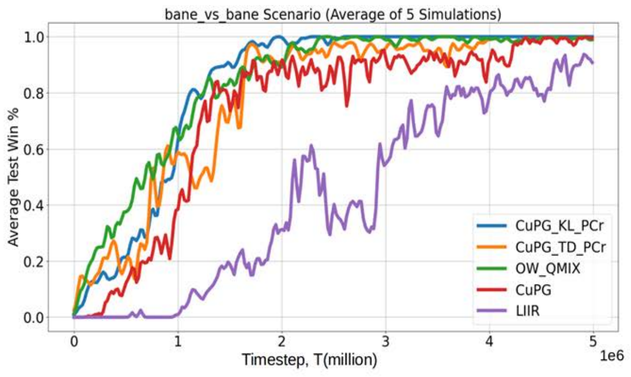
TABLE 7. Test win rates for hard scenarios at different timesteps.

		Average Test Win Rate (%)				
Time Step (M)	Scenario	CuPG_KL_PCr	CuPG_TD_PCr	OW_QMIX	CuPG	LIIR
1.0	3s_vs_5z	**8.9**	18.9	0.00	6.5	2.2
	bane_vs_bane	**60**	59.8	63.1	40	1.03
	MMM	**35.2**	50	48.9	11.8	93.7
3.0	3s_vs_5z	**75.7**	61.1	50.1	70	27.8
	bane_vs_bane	**100**	94	91.3	89.9	53
	MMM	**100**	95.5	80	81.9	71.2
5.0	3s_vs_5z	**98.9**	92.2	73.9	83	48
	bane_vs_bane	**100**	100	99.7	99.3	91
	MMM	**100**	99.8	98.3	89.13	73.5

rates for all the game scenarios as compared with the existing method and faster training converges on the desirable decision-making policy.

Note that the proposed exploration reward does not require extra time through the experiments except a small time at the start of the experiments. After each individual agent





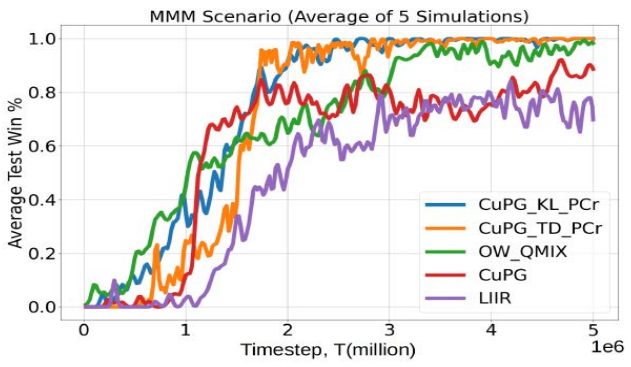
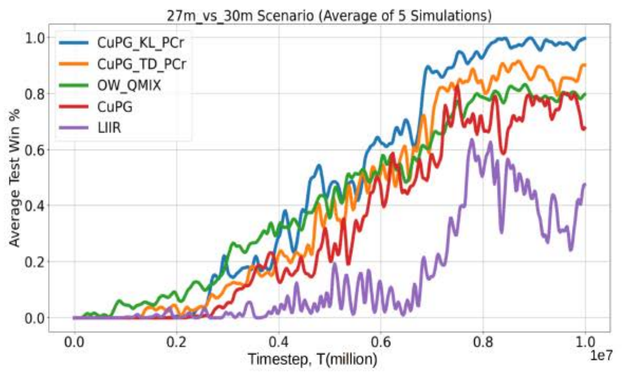
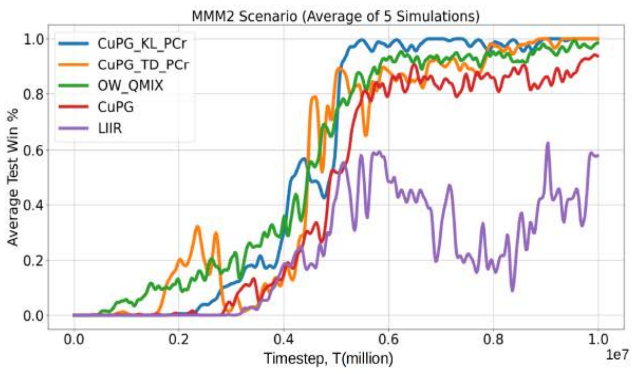


FIGURE 4. **Test win rates for hard scenarios.**





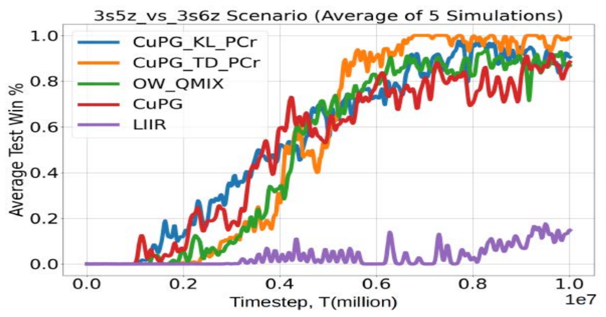


FIGURE 5. **Test win rates for super-hard scenarios.**

computes the conditional mutual information between the target agent's decision-making policy and influenced agents' decision-making policy conditioning the influenced agents' previous timestep decision making policy, the agent will make faster action selection and training converges faster as shown in simulation results (Figs. 4 and 5). We analyzed

TABLE 8. Test win rates for super-hard scenarios at different timesteps.

		Average Test Win Rate (%)				
Time Step (M)	Scenario	CuPG_KL_PCr	CuPG_TD_PCr	OW_QMIX	CuPG	LIIR
1.0	27m_vs_30m	**0.0**	0.0	8.72	0.0	0.0
	MMM2	**0.0**	0.0	6.2	0.0	0.0
	3s5z_vs_3s6z	**0.2**	0.0	0.0	0.0	12.3
4.0	27m_vs_30m	**33.5**	22.7	34.1	14.1	3.76
	MMM2	**32.7**	18.6	37.2	17.9	19.1
	3s5z_vs_3s6z	**52.5**	28.4	34.4	59.8	4.5
8.0	27m_vs_30m	**92.6**	83.7	79.7	70	53.2
	MMM2	**97.9**	97.2	93.5	84	18.8
	3s5z_vs_3s6z	**96.8**	50.9	48.9	98	3.5
10.0	27m_vs_30m	**99.8**	99.6	82.4	71	48.6
	MMM2	**100**	100	98.9	92	59
	3s5z_vs_3s6z	**90.8**	99.1	86.9	88.4	15

the time consumption of each of the existing methods and verified that our proposed algorithm requires reasonable time consumption compared with the existing state-of-the-art methods. From the simulation results, it is concluded that the proposed algorithm effectively improves the test-win rates and converges faster.

V. CONCLUSION

In this paper, we proposed a novel learning technique called ‘‘curiosity-based learning’’ which manipulates the agent's policies by observing others' past behaviors. We formulate the prioritization model to maximize the sample efficiency through frequent sampling of transitions that have a high value of the KL-divergence and compare the performance with several multi-agent benchmark algorithms. Overall, experimental results show that our proposed technique achieved a viable optimal solution with high test win rates for all SMAC game scenarios. The limitation of the proposed work is that it assists execution of only a particular task domain at a time.

Motivated by these criteria, a promising future direction might be to execute multiple task domains simultaneously.

REFERENCES

[1] T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. N. Foerster, and S. Whiteson, ‘‘QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning,’’ in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 4295–4304.

[2] S. Singh, R. L. Lewis, A. G. Barto, and J. Sorg, ‘‘Intrinsically motivated reinforcement learning: An evolutionary perspective,’’ *IEEE Trans. Auto. Mental Develop.*, vol. 2, no. 2, pp. 70–82, Jun. 2010.

[3] Y. Du, L. Han, M. Fang, J. Liu, T. Dai, and D. Tao, ‘‘LIIR: Learning individual intrinsic reward in multi-agent reinforcement learning,’’ in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2019, pp. 8–14.

[4] N. Jaques, A. Lazaridou, E. Hughes, C. Gulcehre, P. A. Ortega, D. Strouse, J. Z. Leibo, and N. de Freitas, ‘‘Social influence as intrinsic motivation for multi-agent deep reinforcement learning,’’ in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 3040–3049.

[5] T. Hester and P. Stone, ‘‘Intrinsically motivated model learning for developing curious robots,’’ *Artif. Intell.*, vol. 247, pp. 170–186, Jun. 2017.

[6] H. M. R. U. Rehman, B.-W. On, D. D. Ningombam, S. Yi, and G. S. Choi, ‘‘QSOD: Hybrid policy gradient for deep multi-agent reinforcement learning,’’ *IEEE Access*, vol. 9, pp. 129728–129741, 2021.

[7] M. G. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, ‘‘Unifying count-based exploration and intrinsic motivation,’’ in *Proc. 30th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 1479–1487.

[8] H. Tang, R. Houthooft, D. Foote, A. Stooke, X. Chen, Y. Duan, J. Schulman, F. D. Turck, and P. Abbeel, ‘‘Exploration: A study of count-based exploration for deep reinforcement learning,’’ in *Proc. 31st Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, p. 280.

[9] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, ‘‘Curiosity-driven exploration by self-supervised prediction,’’ in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2778–2787.

[10] S. Khadka, S. Majumdar, T. Nassar, Z. Dwiel, E. Tumer, S. Miret, Y. Liu, and K. Tumer, ‘‘Collaborative evolutionary reinforcement learning,’’ in *Proc. 36th Int. Conf. Mach. Learn. (PMLR)*, 2019, 3341–3350.

[11] H. Ryu, H. Shin, and J. Park, ‘‘Multi-agent actor-critic with hierarchical graph attention network,’’ in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2020, pp. 7236–7243.

[12] C. Xu, S. Liu, C. Zhang, Y. Huang, Z. Lu, and L. Yang, ‘‘Multi-agent reinforcement learning based distributed transmission in collaborative cloud-edge systems,’’ *IEEE Trans. Veh. Technol.*, vol. 70, no. 2, pp. 1658–1672, Feb. 2021, doi: 10.1109/TVT.2021.3055511.

[13] M. Azmitia and J. Hesser, ‘‘Why siblings are important agents of cognitive development: A comparison of siblings and peers,’’ *Child Develop.*, vol. 64, no. 2, pp. 430–444, Apr. 1993.

[14] G. H. Brody, ‘‘Sibling relationship quality: Its causes and consequences,’’ *Annu. Rev. Psychol.*, vol. 49, no. 1, pp. 1–24, Feb. 1998.

[15] L. Zheng, J. Chen, J. Wang, H. Jiamin, Y. Hu, Y. Chen, C. Fan, Y. Gao, and C. Zhang, ‘‘Episodic multi-agent reinforcement learning with curiosity-driven exploration,’’ in *Proc. 35th Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2021, pp. 1–13.

[16] M. Samvelyan, T. Rashid, C. S. de Witt, G. Farquhar, N. Nardelli, T. G. Rudner, C. M. Hung, P. H. Torr, J. Foerster, and S. Whiteson, ‘‘The starcraft multi-agent challenge,’’ in *Proc. 18th Int. Conf. Auton. Agents Multiagent Syst.*, 2019, pp. 2186–2188.

[17] T. Rashid, G. Farquhar, B. Peng, and S. Whiteson, ‘‘Weighted QMIX: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning,’’ in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1–12.

[18] J. Wang, Z. Ren, T. Liu, Y. Yu, and C. Zhang, ‘‘QPLEX: Duplex dueling multi-agent *Q*-learning,’’ in *Proc. ICLR*, 2021, pp. 1–27.

[19] M. Szemenyei and P. Reizinger, ‘‘Attention-based curiosity in multi-agent reinforcement learning environments,’’ in *Proc. Int. Conf. Control, Artif. Intell., Robot. Optim. (ICCAIRO)*, May 2019, pp. 176–181.

[20] N. Bougie and R. Ichise, ‘‘Skill-based curiosity for intrinsically motivated reinforcement learning,’’ *J. Mach. Learn.*, vol. 109, pp. 493–512, Oct. 2020.

[21] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, ‘‘QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning,’’ in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5887–5896.

[22] M. Dimakopoulou and B. V. Roy, ‘‘Coordinated exploration in concurrent reinforcement learning,’’ in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, p. 80.

[23] M. Dimakopoulou, I. Osband, and B. V. Roy, ‘‘Scalable coordinated exploration in concurrent reinforcement learning,’’ in *Proc. 35th Int. Conf. Mach. Learn.*, 2018.

[24] E. Hughes, J. Z. Leibo, M. G. Phillips, K. Tuyls, E. A. Duez-Guzmn, A. G. Castaeda, I. Dunning, T. Zhu, K. R. McKee, R. Koster, H. Roff, and T. Graepel, ''Inequity aversion improves cooperation in intertemporal social dilemmas,'' in *Proc. 32nd Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 1–11.

[25] M. Mutti and M. Restelli, ''An intrinsically-motivated approach for learning highly exploring and fast mixing policies,'' in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 5232–5239.

[26] D. Zha, K.-H. Lai, K. Zhou, and X. Hu, ''Experience replay optimization,'' in *Proc. 28th Int. Joint Conf. Artif. Intell. (IJCAI)*, Aug. 2019, pp. 4243–4249.

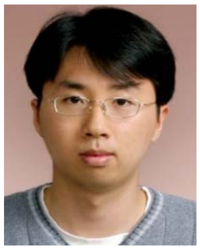
[27] L. F. Vecchietti, T. Kim, K. Choi, J. Hong, and D. Har, ''Batch prioritization in multigoal reinforcement learning,'' *IEEE Access*, vol. 8, pp. 137449–137461, 2020.

[28] H. Zhang, C. Qu, J. Zhang, and J. Li, ''Self-adaptive priority correction for prioritized experience replay,'' *Appl. Sci.*, vol. 10, no. 19, p. 6925, Oct. 2020.

[29] J. Foerster, N. Nardelli, G. Farquhar, T. Afouras, P. H. S. Torr, P. Kohli, and S. Whiteson, ''Stabilising experience replay for deep multi-agent reinforcement learning,'' in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2018, pp. 1146–1155.

DEVARANI DEVI NINGOMBAM received the M.Tech. degree in communication engineering from the National Institute of Technology Karnataka (NITK), Surathkal, India, in 2015, and the Ph.D. degree in computer engineering from Chosun University, Gwangju, South Korea, in 2019. From 2019 to 2021, she was worked as a Postdoctoral Research Fellow at the Artificial Intelligence Laboratory, Smart Data Research Section, Electronics and Telecommunications Research Institute (ETRI), South Korea. From 2021 to 2022, she was an Assistant Professor at the Computer Science and Engineering Department, GITAM University, Visakhapatnam. Currently, she is an Assistant Professor-Senior Scale at the School of Computer Science, UPES, Dehradun. Her current research interests include multiagent deep reinforcement learning (MARL), the Internet of Things (IoT), device-to-device (D2D) communications, and wireless networks.

BYUNGHYUN YOO received the Ph.D. degree in mechanical engineering from the Korea Advanced Institute of Science and Technology (KAIST), in 2019. Since 2019, he has been with the Electronics and Telecommunications Research Institute (ETRI), South Korea, where he is currently a Senior Researcher with the Integrated Intelligence Research Section and the Intelligence Information Research Division. His research interests include the areas of multi-agent reinforcement learning (MARL) and artificial general intelligence (AGI).



HYUN WOO KIM received the B.S. and M.S. degrees in electrical engineering from Seoul National University (SNU), Seoul, South Korea, in 2001 and 2003, respectively. Since 2003, he has been with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, South Korea, where he is currently a Principal Member of Engineering Staff. His research interests include speech signal processing, meta-learning, and machine learning.

HWA JEON SONG received the B.S., M.S., and Ph.D. degrees in electronics engineering from Pusan National University, Republic of Korea, in 1993, 1995, and 2005, respectively. From 1995 to 2001, he was a Researcher at Hyundai Motor Company. Since 2010, he has been a Principal Researcher with the Electronics and Telecommunications Research Institute (ETRI). His research interests include speech recognition, multi-modal representation, and artificial general intelligence (AGI).

SUNGWON YI received the M.S. and Ph.D. degrees in computer science and engineering from Pennsylvania State University, in 2004 and 2005, respectively. Since 2005, he has been with ETRI, South Korea, where he is currently a Principal Researcher with the Planning Division. His research interests include the areas of network security, storage systems, mobile computing, and machine learning. He has been served on the Technical Program Committee for the IEEE GLOBECOM and ICC, since 2005.