Robust Multi-Agent Reinforcement Learning Under Adversarial State Perturbations

Songyang Han, *Student Member, IEEE*, Sanbao Su, *Student Member, IEEE*, Sihong He, *Student Member, IEEE*, Shuo Han, *Member, IEEE,* Haizhao Yang, *Member, IEEE,* Fei Miao, *Member, IEEE*

***Abstract*—Various types of Multi-Agent Reinforcement Learning (MARL) methods have been developed, assuming that the agents' policies are based on the true states. The recent works improve the robustness of MARL under uncertainties arising from reward, transition probability, or opponents' policies. However, in real-world multi-agent systems, state estimations may be perturbed by sensor measurement noise, poor weather conditions, or even adversaries. Agents' policies trained with only accurate state information will deviate from the optimal or suboptimal reward policies when facing adversarial state perturbations during execution. The MARL under adversarial state perturbations has limited study. Hence, in this work, we propose a robust MARL framework under adversarial state perturbations to study the fundamental properties of MARL under state uncertainties. We define the solution concept, the robust perfect Nash equilibrium, of the proposed MARL problem and prove that this equilibrium exists with finite state and finite action space. We design a gradient descent ascent-based robust MARL algorithm to learn the robust policies for the MARL agents. Our experiments show that adversarial state perturbations decrease the total reward of agents for several baselines from the existing literature, while our algorithm gains larger rewards and significantly improves the robustness of the MARL policies under state uncertainties.**

***Index Terms*—Multi-agent reinforcement learning, robust reinforcement learning, Markov game, Nash equilibrium.**

I. Introduction

Multi-Agent Reinforcement Learning (MARL) has been successfully applied to many scenarios, such as multi-robot coordination [1], multi-agent communication [2], sequential social dilemmas [3], resource management [4], etc. The non-stationary training environment issue is addressed by using an actor-critic framework in [5], [6]. However, the learned policies can be brittle and sensitive to measurement noise and converge to a poor local optimum [7]. When agents' observations are extremely noisy, the performance of the learned policies can be drastically worse. Therefore, it is important to have a robust policy: a well-trained agent should behave well under an uncertainty set of state information. Noisy observation issues have been investigated in single-agent scenarios to narrow the gap between simulation and real world [8], [9] or to defend against adversarial attacks [10], [11]. However, how to deal with state uncertainties in a multi-agent setting remains challenging.

Additionally, the Deep Reinforcement Learning (DRL) agent is vulnerable under many attack scenarios [10]–[12]. Imperceptible noises are usually added by these adversaries to perturb the state observation slightly different from the true environment. In real-world applications, agents may not have the perfect state information due to sensor measurement noise, poor weather or illumination conditions [13], [14]. For agents trained in simulation with true state information, the policy may have poor performance in practice, which is usually known as sim-to-real gaps [15]. This is a significant issue, especially for safety-critical applications like autonomous vehicles [16], [17]. Therefore, considering state uncertainty in MARL is a significant challenge and in demand.

In this work, we study the challenges and fundamental properties of robust MARL under adversarial state perturbations. We propose a problem formulation of MARL under state uncertainties and define the concept of robust perfect Nash equilibrium among all the agents and the adversaries. We prove that the equilibrium exists under finite state and finite action space. We design an algorithm, called Robust Multi-Agent Reinforcement Learning (RMARL), to train robust policies of all agents with adversarial state perturbations. The algorithm uses Gradient Descent Ascent (GDA) algorithm [18] to update each agent's policy network and the adversarial state perturbation network. Our experiment results show that the proposed RMARL algorithm improves the robustness of the agents' policy compared with the existing MARL literature. To the best of our knowledge, we propose the first mathematical formulation and algorithm that considers state uncertainty in MARL with both theoretical and empirical justifications. In summary, the main contributions of this work are:

- We formulate a robust multi-agent reinforcement learning problem to study the fundamental properties of MARL under adversarial state perturbations. We define the solution concept, robust perfect Nash equilibrium, and prove that the equilibrium exists with finite state and finite action space.
- We propose an algorithm, called Robust Multi-Agent Reinforcement Learning (RMARL), to solve the challenge of training robust policies under adversarial state perturbations.
- We empirically evaluate our proposed RMARL algorithm

This work was supported by NSF 1849246, NSF 1952096 and NSF 2047354 grants.

Songyang Han, S. Su, S. He, and F. Miao are with the Department of Computer Science and Engineering, University of Connecticut, Storrs Mansfield, CT, USA 06268.

Shuo Han is with the Department of Electrical and Computer Engineering, University of Illinois at Chicago, Chicago, IL, USA 60607.

H. Yang is with the Department of Mathematics, Purdue University, West Lafayette, IN, USA 47907.

Email: {songyang.han, sanbao.su, sihong.he, fei.miao}@uconn.edu, hanshuo@uic.edu, haizhao@purdue.edu.

in cooperative and competitive environments. The agents' policies trained by the RMARL outperform baselines. We also show that our algorithm achieves higher mean episode reward under state uncertainties while the baselines downgrade their rewards under state uncertainties.

The rest of this paper is organized as follows. We introduce related work in Section II and preliminary concepts in Section III. In Section IV, we give the problem formulation and the solution concept of the MARL with adversarial state perturbations. In Section V, we propose an algorithm for finding a robust policy for the MARL agents. The experiments are shown in Section VI via simulations in cooperative and competitive environments. In Appendix, we give detailed proof of the existence of the solution concept.

II. Related Work

Multi-Agent Reinforcement Learning has a long history in the AI field [19], [20]. The most popular branch is the cooperative MARL assuming all agents are cooperating. Existing works have been investigated to encourage every agent to work collaboratively by assigning rewards appropriately, such as a value decomposition network [21], [22], subtracting a counterfactual baseline [6], or an implicit method [23]. The recent advances in MARL include Multi-Agent Deep Deterministic Policy Gradient (MADDPG) and League training. MADDPG uses a centralized Q-function to alleviate the problem caused by the non-stationary environment [5]. The scalability issue of the MADDPG can be alleviated by adding attention to the critic [24] or using neighbor information [25]. League training is used in AlphaStar to avoid the cycles that appear in self-play and increase the diversity of the strategies [26]. However, during training, all agents are assumed to get the true state value. None of the recent advances specifies how to deal with perturbed state values by malicious adversaries.

Robust reinforcement learning can be traced back to [27] in the single-agent setting. Combined with deep learning techniques, the robust MARL is recently studied considering uncertainties in reward [28], transition dynamics [28]–[30], training partner's type [31] and policies [32], [33]. The work in [28] considers the robust equilibrium of multi-agents with reward and transition dynamics uncertainties when the perfect knowledge of the environment model is unknown. The work in [31] considers uncertain training partner's type (e.g. adversary, neutral, or teammate) to the protagonist in two-player scenarios. The M3DDPG algorithm extends the MADDPG to the robust adversarial reinforcement learning for the worst situation assuming all training partners are adversaries [32]. However, none of the above works considers the state uncertainty. Recent works in [9], [34] first consider the adversarial state perturbations for single-agent reinforcement learning. In this work, we focus on the multi-agent cases with adversarial state perturbations, where the interaction among different agents and equilibrium policies is much more challenging.

Nash equilibrium is a solution concept proposed by Nash in [35] for general-sum finite one-shot games. In Nash equilibrium, each player selects the best response strategy to the others' strategies. No player would like to deviate from the Nash equilibrium otherwise its utility becomes worse. This concept is extended to infinite games by Debreu [36], Glicksberg [37], and Fan [38]. Markov game is first defined by Shapley in [39] in a two-player zero-sum setting with a sequential decision process. Fink extends the Nash equilibrium concept to Markov games in [40] and proves that an equilibrium point exists in n-player general-sum discounted Markov games. The uncertainty in transition dynamics of a Markov game is considered in [41], [42] using a robust optimization approach with independent proofs for the existence of the equilibrium point. Besides uncertainty in transition dynamics, uncertainty in utility (it is called "reward" in reinforcement learning) is also considered in [43] for n-player finite state/action discounted Markov games with the proof for the existence of the equilibrium point. However, the uncertainty in the state has not been studied yet for Markov games. To the best of our knowledge, we are the first to formulate the problem for n-player finite state/action discounted Markov games with state uncertainty and prove the existence of the equilibrium point. We also propose an empirical algorithm for solving this problem utilizing the recent MARL techniques.

III. Preliminary

We introduce preliminary concepts in Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [5] and Gradient Descent Ascent (GDA) [18] in this section.

A. Multi-Agent Deep Deterministic Policy Gradient (MADDPG)

It is difficult to apply single-agent RL algorithms directly to the multi-agent case, because the environment's state transition is also influenced by the policy of other agents and it is non-stationary from a single agent's view. To alleviate this problem and stabilize training, the MADDPG algorithm is proposed using a centralized Q function that has global state and global action information [5]. It assumes all agents are self-interested and every agent's objective is to maximize its own expected return. The objective for agent i is $J(\theta^i) = \mathbb{E}[R^i]$ and its gradient is

$$\begin{aligned}&\nabla_{\theta^i} J(\theta^i) = \\ &\mathbb{E}_{\mathbf{x},a\sim\mathcal{D}}\left[\nabla_{\theta^i}\mu^i(o^i)\nabla_{a^i}Q^i(\mathbf{x},a^1,...,a^n)|_{a^i=\mu^i(o^i)}\right],\end{aligned} \tag{1}$$

where $Q^i(\mathbf{x}, a^1, ..., a^n)$ is a centralized action-value function, $\mathbf{x} = (o^1, \ldots, o^n)$, and o^i represents agent i's observation. The experience replay buffer $\mathcal{D}$ contains transition experience $\mathbf{x}, a^1, ..., a^n, \mathbf{x}', r^1, ..., r^n$ to decorrelate data. The centralized Q^i can be trained using the Bellman loss:

$$\begin{aligned}&\mathcal{L}(\theta^i) = \mathbb{E}_{\mathbf{x},a,\mathbf{r},\mathbf{x}'\sim\mathcal{D}}[y - Q^i(\mathbf{x}, a^1, ..., a^n)]^2, \\ &y = r^i + \gamma Q^{i\prime}(\mathbf{x}', a^{1\prime}, ..., a^{n\prime})|_{a^{j\prime}=\mu^{j\prime}(o^j)},\end{aligned} \tag{2}$$

where $Q^{i\prime}$ is the target network whose parameters are copied from Q with a delay to stabilize the moving target. Note that this algorithm adopts centralized training and decentralized execution paradigm. When testing, each agent can only access its local observation to select actions.

In M3DDPG [32], the uncertainty from training partner's policies is considered: all other partners are considered as adversaries that select actions to minimize the expected return of the training agent. In other words, when updating both actor and critic, they select training partner's actions by $a^{j\neq i} = \arg\min_{a^{j\neq i}} Q^i(\mathbf{x}, a^1, ..., a^n)$.

B. Gradient Descent Ascent (GDA)

Gradient Descent Ascent (GDA) [18] is currently one widely-used algorithm for solving the following minimax optimization problem defined in (3):

$$\min_{x\in\mathbb{R}^m} \max_{y\in\mathbb{R}^n} f(x, y). \tag{3}$$

GDA simultaneously performs gradient descent update on the variable x and gradient ascent update on the variable y according to (4).

$$\begin{aligned} x_{t+1} &= x_t - \eta_x \nabla_x f(x_t, y_t), \\ y_{t+1} &= y_t + \eta_y \nabla_y f(x_t, y_t). \end{aligned} \tag{4}$$

It has a variety of variants to accommodate different types of geometries of the minimax problem, such as convex-concave geometry, nonconvex-concave geometry, nonconvex-nonconcave geometry, etc.

IV. Problem Formulation

We consider n agents in the agent set $\mathcal{N} = \{1, ..., n\}$. Each agent i is associated with an action $a^i \in \mathcal{A}^i$ and a state $s^i \in \mathcal{S}^i$. The global joint state is $s = (s^1, ..., s^n) \in \mathcal{S}$, $\mathcal{S} := \mathcal{S}^1 \times \cdots \times \mathcal{S}^n$. The global joint action is $a = (a^1, ..., a^n) \in \mathcal{A}$, $\mathcal{A} := \mathcal{A}^1 \times \cdots \times \mathcal{A}^n$. Each agent has a stage-wise reward function $r^i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. We consider that each agent is associated with a state perturbation function $\rho^i : \mathcal{S} \to \mathcal{S}$ to perturb the knowledge or observation about the global state. The corresponding perturbed state is $\rho^i(s)$. We denote the joint state perturbation as $\rho := [\rho^i]_{i\in\mathcal{N}}$. We consider the admissible perturbed state as a task-specific "neighboring" state of s, e.g. the bounded sensor measurement errors, to model the real-world challenges of getting accurate states for multi-agent systems like connected and autonomous vehicles and multi-robots systems [13], [14]. The state perturbation reflects the state uncertainty from the perspective of each agent, and it does not change the true state of multi-agent systems. In particular, we have the following constraint for the state perturbation:

Definition 1 (Admissible State Perturbation Set). *We consider the set of admissible state perturbation for agent i as $\mathcal{P}^i := \{\rho^i : \|\rho^i(s) - s\| \leq d \text{ for all } s \in \mathcal{S}\}$ where d is a radius denoting how large the set is. Denote the joint admissible state perturbation set as $\mathcal{P} := \mathcal{P}^1 \times \cdots \times \mathcal{P}^n$.*

The admissible state perturbation of state s must stay within a ball centered at s with radius d. A d-ball is a popular way to define the uncertainty set in adversarial robust reinforcement learning [9], [44]. The state transition function is $p : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ is a probability simplex denoting the set of all possible probability measures on $\mathcal{S}$. Each agent is associated with a policy $\pi^i : \mathcal{S} \to \Delta(\mathcal{A}^i)$ to choose an action $a^i \in \mathcal{A}^i$ given the perturbed state $\rho^i(s)$. Note that the input of π^i is the perturbed state $\rho^i(s) \in \mathcal{S}$. The perturbed state affects each agent's action. The set $\Delta(\mathcal{A}^i)$ includes all possible probability measures on $\mathcal{A}^i$. We use $\pi(a|\rho(s)) = \Pi_{i=1}^n \pi^i(a^i|\rho^i(s))$ to denote the joint policy, as all agents make decisions independently.

We consider each agent is associated with an adversary aiming to minimize the total reward of the agent. The adversary policy, i.e. the state perturbation policy, associated with agent i is $\chi^i : \mathcal{S} \to \Delta(\mathcal{P}^i)$, where the input of χ^i is the true state and $\Delta(\mathcal{P}^i)$ is a probability simplex denoting all possible probability measures on the admissible state perturbations. The probability distribution of the admissible state perturbation on state s is $\chi^i(\rho^i|s)$. Denote the joint adversary as $\chi(\rho|s) = \Pi_{i=1}^n \chi^i(\rho^i|s)$, as all adversaries make decisions independently. Agents want to find a policy π to maximize their total reward while adversaries want to find a policy χ to minimize agents' total reward.

Our problem cannot be solved by the existing work in the Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [45] or Markov games [5], [21], [22]. In contrast, the policy in our problem needs to be robust under a set of admissible state perturbations. The adversary aims to find the worst-case state perturbation policy χ to minimize the reward of each agent. In the following theorem, we show the connection between our problem, the Dec-POMDP, and Markov games.

Proposition 1. *When the joint state perturbation function ρ is fixed, the above problem becomes a Dec-POMDP [45]. Moreover, when ρ is a fixed injective mapping, the above problem becomes a Markov game.*

Proof. When the joint state perturbation function ρ is fixed, this problem becomes a Dec-POMDP with a conditional observational function $O(o|s) = \rho(s)$, where o is the observation given the state s.

Moreover, when ρ is a fixed injective mapping, the above problem becomes a Markov game that is constructed as follows: Taking $s_{new} = \rho(s)$ as the new state, the new global joint state set is $\mathcal{S}_{new} := \mathcal{S} \times \cdots \times \mathcal{S}$. The global joint action set $\mathcal{A}_{new} = \mathcal{A} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^n$ and the agent set $\mathcal{N}_{new} = \mathcal{N}$ stay the same.

We can construct a new reward function $r^i_{new} : \mathcal{S}_{new} \times \mathcal{A}_{new} \to \mathbb{R}$ for each agent i as

$$r^i_{new}(s_{new} = \rho(s), a_{new} = a) = r^i(s, a), \tag{5}$$

and a new state transition function $p_{new} : \mathcal{S}_{new} \times \mathcal{A}_{new} \to \Delta(\mathcal{S}_{new})$ defined as

$$p_{new}(\rho'(s')|\rho(s), a) = \sum_{\eta'\in\{\eta'=\rho'(s')\}} \sum_{\eta\in\{\eta=\rho(s)\}} p(\eta'|\eta, a). \tag{6}$$

Each agent uses a policy $\pi^i_{new} : \mathcal{S}_{new} \to \Delta(\mathcal{A}^i)$ to choose an action based on the new state. Hence, this problem becomes a Markov game when ρ is a fixed injective mapping. □

Different from Dec-POMDP or Markov games, the state perturbation ρ^i for each agent is selected by the adversary

policy χ^i in our problem. Note that agents cannot get the true state s in Dec-POMDP, but in our problem, the true state s is known by the adversaries. Adversaries can take the true state information and use it to select the optimal attack for the MARL agents. Moreover, our problem is different from robust Markov game that optimizes for the uncertainties from reward [28], transition dynamics [28]–[30], training partner's policies [32]. Hence, it cannot be solved by methods from the existing robust MARL literature.

The state value function for agent i is defined as

$$V^i(s) = \mathbb{E}_{a_t \sim \pi, \rho_t \sim \chi} \left[\sum_{t=0}^{\infty} \gamma^t r^i_{t+1} | s_0 = s \right], \tag{7}$$

where γ is the discount factor. The Bellman-type equation for the state value function V^i is

$$V^i(s) = \max_{\pi^i} \min_{\chi^i} \sum_{\rho \in \mathcal{P}} \chi(\rho|s) \sum_{a \in \mathcal{A}} \pi(a|\rho(s)) \times \left(r^i + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^i(s') \right). \tag{8}$$

The basic idea is that if an agent knew how to act optimally from the next stage on, then, at the current stage, it would choose the policy that maximizes the minimum sum of expected immediate reward and expected reward possibly incurred in future stages.

We aim to find a robust perfect Nash equilibrium, the solution concept for the MARL with adversarial state perturbations, defined as follows:

Definition 2 (Robust Perfect Nash Equilibrium). *A joint agent policy $\pi^* = (\pi^{1*}, \pi^{2*}, ..., \pi^{n*})$ and joint adversary policy $\chi^* = (\chi^{1*}, \chi^{2*}, ..., \chi^{n*})$ is a robust perfect Nash equilibrium if there exists a vector-valued function $V^* = (V^{1*}, V^{2*}, ..., V^{n*})$ with each $V^{i*} : \mathcal{S} \to \mathbb{R}$ and for all $i \in \mathcal{N}$ and $s \in \mathcal{S}$*

$$V^{i*}(s) = \max_{\pi^i} \min_{\chi^i} \sum_{\rho \in \mathcal{P}} \chi^i(\rho^i|s) \prod_{j \neq i} \chi^{j*}(\rho^j|s) \sum_{a \in \mathcal{A}} \pi^i(a^i|\rho^i(s)) \prod_{j \neq i} \pi^{j*}(a^j|\rho^j(s)) \left(r^i(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^{i*}(s') \right), \tag{9}$$

and for all π^i and χ^i

$$\begin{aligned} J^i_s(\pi^i, \pi^{-i*}, \chi^{i*}, \chi^{-i*}) &\leq J^i_s(\pi^{i*}, \pi^{-i*}, \chi^{i*}, \chi^{-i*}) \\ &\leq J^i_s(\pi^{i*}, \pi^{-i*}, \chi^{i}, \chi^{-i*}), \end{aligned} \tag{10}$$

where J^i_s is defined as

$$J^i_s(\pi^i, \pi^{-i}, \chi^i, \chi^{-i}) = \sum_{\rho \in \mathcal{P}} \chi(\rho|s) \sum_{a \in \mathcal{A}} \pi(a|\rho(s)) \times \left(r^i + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^{i*}(s') \right), \tag{11}$$

the notation π^{-i} and χ^{-i} denotes the agent policies and adversary policies of all other agents except agent i respectively.

This definition shows that π^* is in a robust perfect Nash equilibrium if each agent's policy is a robust best response to the other agents' policies under adversarial state perturbations. When agent i is calculating its robust best response, it assumes a worst-case perspective of the state perturbations. In other words, the n agents are trained during the process where the adversary also aims to find χ^*, the optimal state perturbation policy to minimize the reward obtained by each agent. Hence, the effects of state uncertainties are considered by the policies of the multi-agent system. One major difference between our work and the existing work considering robustness to reward or transition dynamics is that $J^i_{s,a}$ is also a function of χ^{-i}, while the robustness to reward or transition dynamics considers $J^i_{s,a}(\pi^i, \pi^{-i}, \chi^i)$. Adding χ^{-i} as a variable in $J^i_{s,a}$ makes our problem more difficult.

We present the main theoretical result, the existence of the robust perfect Nash equilibrium defined in Definition 2, in the following theorem.

Theorem 1. *For finite state and finite action space, the robust perfect Nash equilibrium defined in Definition 2 exists.*

Proof. See proof in Appendix A. □

In proof, we first show that there exists a unique value function satisfying the Bellman-type equation given π^{-i} and χ^{-i}. Also, we show the admissible state perturbation set is finite. Then, we construct a game with n agent players and n adversary players. Every player in this game wants to maximize its own utility. We show that the robust perfect Nash equilibrium in Definition 2 exists by finding a Nash equilibrium in the constructed game.

V. Robust Multi-Agent Reinforcement Learning Algorithm

In general, it is challenging to develop an algorithm to compute an optimal or equilibrium policy for MARL with parameter uncertainties [28], [34]. Note that it is feasible to find the robust perfect Nash equilibrium in Definition 2 in a simple problem with a very small number of agents, states, and actions by combining the value iteration and the best response method as discussed in the detailed proof of Theorem 1. However, this approach is not practical once the agent, state, or action space becomes large. It is significant and necessary to use neural networks as a function approximation in MARL because the joint action space grows exponentially with the total number of agents. Moreover, it is computationally hard to calculate the equilibrium for a general-sum game even without any uncertainties [46]. In order to deal with massive or even continuous state-action space, we design an actor-critic algorithm based on the recent MARL advances to approximate a robust policy under adversarial state perturbations.

Our algorithm adopts centralized training and decentralized execution paradigm following the popular framework in [5]. During training, there is a centralized critic $Q^i(s,a)$ for each agent i that records the expected total reward given global state s and global action a. The connection between $Q^i(s,a)$ and $V^i(s)$ is that for any $i \in \mathcal{N}, s \in \mathcal{S}, a \in \mathcal{A}$,

$$Q^i(s,a) = r^i(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^i(s'). \tag{12}$$

Each agent's state for the actor is perturbed by an adversarial state perturbation function $\rho^i : \mathcal{S} \rightarrow \mathcal{S}$ that satisfies the constraint $\|\rho^i(s) - s\| \leq d$. During execution, each agent i selects action a^i based on the perturbed state $\rho^i(s) \in \mathcal{S}$ using a trained policy $\pi^i : \mathcal{S} \rightarrow \Delta(\mathcal{A}^i)$. We want to find a policy π^i for each agent that is robust to the adversarial state perturbations.

Algorithm 1: Robust MARL Under Adversarial Perturbations on State

1 Randomly initialize the critic network Q^i, the actor network π^i, and the adversarial perturbation network P^i for agent i. Initialize target networks $Q^{i\prime}, \pi^{i\prime}, P^{i\prime}$;
2 **for** *each episode* **do**
3 Initialize a random process $\mathcal{X}$ for action exploration;
4 Receive initial state s;
5 **for** *each time step* **do**
6 For each agent i, select action $a^i \sim \pi^i(\rho^i(s)) + \mathcal{X}$ w.r.t the current policy and exploration. Execute actions $a = (a^1, ..., a^n)$ and observe the reward $r = (r^1, ..., r^n)$ and the new state information s' and store(s, a, r, s') in replay buffer $\mathcal{D}$. Set $s \leftarrow s'$;
7 **for** *agent i=1 to n* **do**
8 Sample a random minibatch of K samples (s_k, a_k, r_k, s'_k) from $\mathcal{D}$;
9 Set $y^i_k = r^i_k + \gamma Q^{i\prime}(s'_k, a'_k)|_{a^{i\prime}=\pi^{i\prime}(\rho^{i\prime}(s'))}$;
10 Update critic by minimizing the loss $\mathcal{L}(\theta^i) = \frac{1}{K}\sum_k (y^i_k - Q^i(s_k, a_k))^2$;
11 **for** *each iteration step* **do**
12 Update actor $\pi^i_{\theta^i}$ and adversary $P^i_{\xi^i}$ using the gradient descent ascent algorithm:
13 $\theta^i \leftarrow \theta^i + \alpha \frac{1}{K}\sum_k \nabla_{\theta^i}\pi^i(\rho^i(s_k))$ $\nabla_{a^i} Q^i(s_k, a_k)$ where $a^i \sim \pi^i(\rho^i(s_k))$;
14 $\xi^i \leftarrow \xi^i - \alpha \frac{1}{K}\sum_k d\nabla_{\xi^i} P^i(s_k)$ $\nabla_{\rho^i}\pi^i(\rho^i(s_k))\nabla_{a^i} Q^i(s_k, a_k)$ where $a^i \sim \pi^i(\rho^i(s_k))$;
15 **end**
16 **end**
17 Update all target networks: $\theta^{i\prime} \leftarrow \tau\theta^i + (1-\tau)\theta^{i\prime}$.
18 **end**
19 **end**

As shown in Alg. 1, our algorithm has three neural networks for each agent: one critic network Q^i, one actor network π^i, and one adversarial perturbation network P^i. The critic network Q^i takes in the true global state and global action during the training process. It returns a Q-value denoting the expected total reward of agent i given s and a. The state transition experience is represented by (s, a, r, s') where s' is the next state. It is stored in a replay buffer $\mathcal{D}$ for the critic network's training. We apply "replay buffer" and "target network" techniques that are developed in [47]. The critic network is trained by minimizing the Bellman loss in line 10. We use the critic to train an actor π^i and the corresponding perturbation P^i for each agent. The actor network π^i receives perturbed state $\rho^i(s)$ and returns a probability distribution over $\mathcal{A}$. In implementation, the adversarial perturbation network takes in the true state s and outputs a state perturbation $P^i(s) = (\rho^i(s) - s)/d$. We can recover the perturbed state as $\rho^i(s) = s + d \times P^i(s)$. We use the "tanh" function as the last layer's activation function of the perturbation network such that the perturbed state $\rho^i(s)$ satisfies the constraint $\|\rho^i(s) - s\| \leq d$.

In the inner loop from line 11 to line 15, we use Gradient Descent Ascent (GDA) algorithm [18] to update parameters for each agent's actor network π^i and the perturbation network P^i. Each agent updates the actor network to maximize its total reward, while the corresponding adversary updates the perturbation network to minimize the agent's total reward. How to solve a non-convex non-concave min-max problem is a very challenging and not well-solved problem. To the best of our knowledge, the GDA algorithm is currently the most widely used and accepted algorithm for this problem, though it is not guaranteed to converge [48]. We use the GDA algorithm as a tool in our algorithm by leveraging the existing literature on solving non-convex non-concave min-max problem. Future advances in this kind of min-max problem will also benefit our algorithm by replacing the GDA algorithm from line 11 to line 15 with new advances.

VI. Experiments

To show the effectiveness of our algorithm, we adopt the multi-agent particle environments developed in [5] that have several agents and landmarks in a two-dimensional world as shown in Fig. 1. The host machine adopted in our experiments is a server configured with AMD Ryzen Threadripper 2990WX 32-core processors and four Quadro RTX 6000 GPUs. Our experiments are performed on Python 3.5.4, Gym 0.10.5, Numpy 1.14.5, Tensorflow 1.8.0, and CUDA 9.0. All hyperparameters used in experiments for RMARL are listed in Table I. Our code will be available online upon publication.

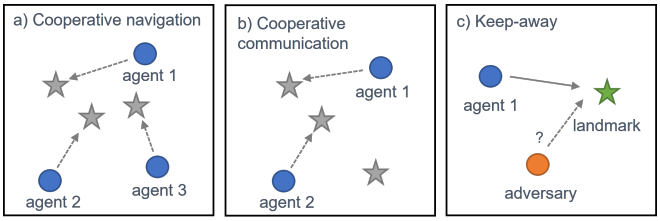


Fig. 1. Some environments to test our algorithm, including a) Cooperative navigation b) Cooperative communication c) Keep-away.

A. Environments

We have tested our algorithm in all scenarios provided by [5]. Our algorithm shows effectiveness in all scenarios. In the experiment section, we report our results in the 3 most representative scenarios as follows.

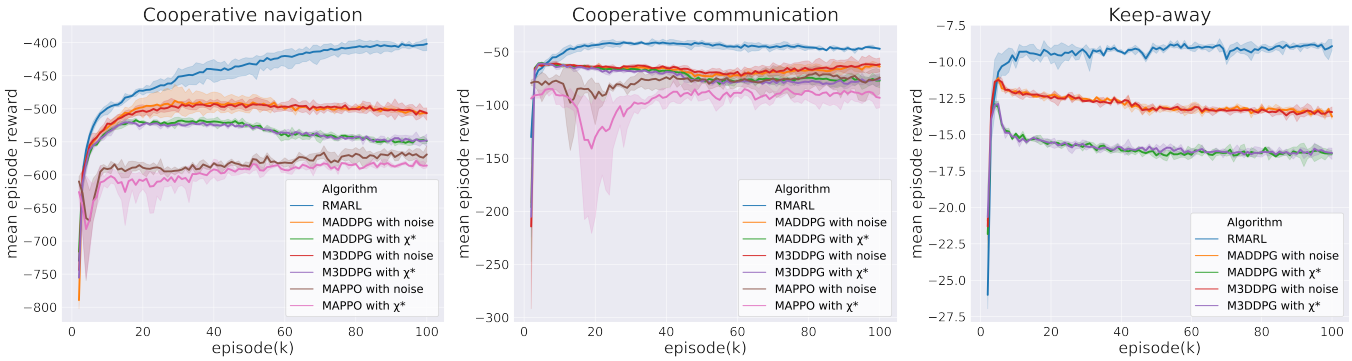


Fig. 2. Our RMARL algorithm compared with baseline algorithms during the training process. The shadow is the 100% confidence interval across 10 runs of each setting. Our RMARL algorithm gets higher mean episode reward and is more robust to the state uncertainty. All baselines are trained under truncated normal state noise or well-trained state perturbation adversary χ^*. Because MAPPO only works in fully cooperative tasks, we only report its results in cooperative navigation and cooperative communication.

TABLE I
RMARL HYPERPARAMETERS

Parameter	Value
optimizer	Adam
learning rate	0.01
discount factor	0.95
replay buffer size	10^6
number of hidden layers	2
number of hidden unites per layer	64
number of samples per minibatch	1024
target network update coefficient τ	0.01
GDA iteration steps	20
radius d	1.0
uncertainty level λ	0.5
upper boundary u	1.0
lower boundary l	-1.0
episodes in training	10k
time steps in one episode	25

1) Cooperative navigation (CN): This is a cooperative task. Agents want to occupy/cover all the landmarks. They need to cooperate through physical actions about their preferred landmark to cover. Also, they will be penalized when collisions happen.

2) Cooperative communication (CC): This is a cooperative task. Each agent needs to get to its target landmark, which is known only by another agent. They have to learn communication and get to landmarks. Besides, both of them are generous agents that pay more attention to helping others, i.e. rewarded more if the other agent gets closer to the target landmark.

3) Keep-away (KA): This is a competitive task. The agent knows the position of the target landmark and wants to reach it. The adversary does not know the target landmark and wants to prevent the agent from reaching the target by pushing them away or occupying the target temporarily.

B. Baselines

We compare the performance of our algorithm with MADDPG [5], M3DDPG [32], and MAPPO [49] and follow their open-source implementation. We have a brief introduction of the first two methods in the preliminary in Section III-A. There is no robustness considered in MADDPG and MAPPO. The M3DDPG considers the robustness to training partner's policies, but it does not consider state uncertainty. The MAPPO is the multi-agent version of the Proximal Policy Optimization (PPO), a popular policy gradient algorithm. Because MAPPO only works in fully cooperative tasks, we only report its results in cooperative navigation and cooperative communication. Note that MAPPO is also used in [50] but they do not provide an open-source implementation. Therefore, we select the latest implementation in [49] with the open-source code. For training and testing, we both report statistics that are averaged across 10 runs in each scenario and algorithm.

We have a totally 9 baselines in our experiment: MADDPG, M3DDPG, MAPPO, MADDPG/M3DDPG/MAPPO with truncated normal perturbation, MADDPG/M3DDPG/MAPPO with well-trained adversarial perturbation. To test the robustness to the state uncertainty, we impose state noise to MADDPG and M3DDPG produced by a truncated normal distribution $\mathcal{N}(0, \lambda, u, l)$ where λ is the uncertainty level, u and l are respectively upper and lower boundaries to ensure noise compact. We use the truncated normal noise to simulate the adversaries selecting a random state perturbation. While in our RMARL algorithm, agents are trained under adversaries that try to minimize the total reward of each agent. For each scenario, we save the well-trained adversaries χ^* in RMARL to represent the optimal state perturbation adversaries. We then use the well-trained adversaries to perturb the states for MADDPG, M3DDPG, and MAPPO to train and test their robustness under adversarial state perturbations.

TABLE II
MEAN EPISODE REWARD OF THE LAST 1000 EPISODES DURING TRAINING. OUR RMARL ALGORITHM ACHIEVES UP TO 58.46% HIGHER MEAN EPISODE REWARDS THAN THE BASELINES.

	CN	CC	KA
RMARL	-401.7	-47.02	-8.93
MADDPG w/ $\mathcal{N}$	-506.48	-63.76	-13.76
M3DDPG w/ $\mathcal{N}$	-506.54	-61.71	-13.45
MAPPO w/ $\mathcal{N}$	-569.07	-94.28	-
MADDPG w/ χ^*	-548.80	-77.01	-16.30
M3DDPG w/ χ^*	-547.99	-75.87	-16.26
MAPPO w/ χ^*	-585.83	-113.19	-

C. Comparison

We first compare our algorithm with baselines during the training process to show that our RMARL algorithm can outperform baselines to get a higher mean episode reward under adversarial state perturbations. Note that our RMARL algorithm has a built-in adversary to perturb states, so we do

TABLE III
MEAN EPISODE REWARD OF 2000 EPISODES DURING TESTING UNDER RANDOM STATE UNCERTAINTY. OUR RMARL ALGORITHM ACHIEVES UP TO 38.86% HIGHER MEAN EPISODE REWARDS THAN THE BASELINES WITH RANDOM STATE NOISE.

		CN	CC	KA
MADDPG	mean	-388.59	-45.79	-8.8
	std	60.72	23.5	5.07
M3DDPG	mean	-390.94	-39.55	-8.54
	std	59.83	20.53	5.04
MAPPO	mean	-381.70	-37.62	-
	std	54.06	18.94	-
MADDPG w/ $\mathcal{N}$	mean	-487.67	-55.79	-11.21
	std	72.28	26.78	6.82
M3DDPG w/ $\mathcal{N}$	mean	-478.96	-54.40	-11.28
	std	70.27	26.64	6.71
MAPPO w/ $\mathcal{N}$	mean	-523.83	-86.51	-
	std	78.45	30.86	-
RMARL w/ $\mathcal{N}$	mean	-437.42	-52.89	-9.89
	std	65.15	25.09	5.92

TABLE IV
MEAN EPISODE REWARDS OF 2000 EPISODES DURING TESTING UNDER WELL-TRAINED ADVERSARIAL STATE PERTURBATIONS. OUR RMARL ALGORITHM ACHIEVES UP TO 56.50% HIGHER MEAN EPISODE REWARD THAN THE BASELINES WITH WELL-TRAINED ADVERSARIES.

		CN	CC	KA
MADDPG w/ χ^*	mean	-537.56	-71.65	-14.72
	std	72.28	42.50	5.44
M3DDPG w/ χ^*	mean	-515.85	-70.68	-13.51
	std	74.58	41.54	5.30
MAPPO w/ χ^*	mean	-572.39	-109.26	-
	std	79.34	47.07	-
RMARL w/ χ^*	mean	-395.3	-47.53	-9.19
	std	63.52	27.64	5.10

not train it under truncated normal noise. After the training of RMARL, we save the well-trained adversary χ^* and use it to train other baselines. Comparing RMARL to other baselines with the same χ^*, the RMARL gets a higher mean episode reward. It shows our RMARL algorithm is more robust under adversarial state perturbations. We also implement baselines with a truncated normal noise, because adding random noise is often used when studying various uncertainties for MARL. Comparing each baseline with random noise to the baseline with the well-trained adversary χ^*, we can see the adversary trained by the RMARL is more powerful than the random state perturbation. Because the adversary in the RMARL intentionally selects state perturbations to minimize agents' rewards. The mean episode reward of the last 1000 episodes during training is shown in Table II. Our RMARL algorithm achieves up to 58.46% higher mean episode rewards than the baselines. Compared RMARL with other baselines with the same well-trained adversary χ^*, the results show that our RMARL algorithm is more robust to the adversarial state uncertainty.

We then test the learned policy with baselines to show the learned policy is more robust under different state uncertainties. As shown in Table III, the mean episode reward is averaged across 2000 episodes and 10 test runs in each environment. We put all the well-trained agents using different algorithms into environments with injected truncated normal noise. The original MADDPG, M3DDPG, and MAPPO's results are shown as a reference for no state perturbation scenario. The MADDPG, M3DDPG, and MAPPO get much lower mean episode reward after injecting random state perturbations (represented by noise). It shows that state uncertainty has a large impact on the MARL and an algorithm to handle state uncertainty is in demand. As shown in Table III, the RMARL algorithm has lower mean episode reward than the reference but achieves up to 38.86% higher mean episode rewards than MADDPG, M3DDPG, and MAPPO with random state noise. It shows that the policy trained by RMARL is more robust than baselines in environments with random state noise. Note that our RMARL's reward is slightly lower than MADDPG and M3DDPG because there is no uncertainty in the baselines' original settings. We also test the well-trained agents using different algorithms in environments with well-trained adversaries to perturb states. We first train the adversary χ^* by RMARL. Then we train and test the MADDPG, M3DDPG, and MAPPO with the same well-trained χ^*. The result is shown in Table IV. Our RMARL algorithm achieves up to 56.50% higher mean episode reward than the baselines with well-trained adversaries. The result from the above two tests shows that our RMARL algorithm achieves higher robustness under both random and well-trained adversarial state perturbations.

VII. CONCLUSION

In this work, we study the fundamental properties of robust multi-agent reinforcement learning with adversarial state perturbations, which are not studied by the existing robust MARL literature. We prove that the robust perfect Nash equilibrium defined in this work exists for finite state and finite action space. This is the major theoretical contribution of our work. We also propose a Robust Multi-Agent Reinforcement Learning (RMARL) algorithm that can find a robust policy for the MARL agents under adversarial state perturbations. We show in numerical experiments that the RMARL algorithm can improve the robustness of the trained policies under random state noise or adversarial state perturbations in both cooperative and competitive environments.

APPENDIX

We prove that the equilibrium defined in Definition 2 exists for finite state and action sets in this appendix.

Assumption 1. *The global state set $\mathcal{S}$ and the global action set $\mathcal{A}$ are finite sets.*

Denote the agent policies and adversary policies of all other agents except agent i as π^{-i} and χ^{-i} respectively. We first show that there exists a unique robust state value function for agent i given any π^{-i} and χ^{-i} in the following theorem.

Theorem 2. *For all $i \in \mathcal{N}$, under Assumption 1, for any given π^{-i} and χ^{-i} of other agents except agent i, there exists a unique robust best response state value function V^i such that*

$$V^i(s) = \max_{\pi^i} \min_{\chi^i} \sum_{\rho \in \mathcal{P}} \chi^i(\rho^i|s) \prod_{j \neq i} \chi^j(\rho^j|s) \sum_{a \in \mathcal{A}} \pi^i(a^i|\rho^i(s)) \prod_{j \neq i} \pi^j(a^j|\rho^j(s)) \left(r^i(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^i(s') \right). \quad (13)$$

Proof. We first introduce some notations for this proof. For a given state value function $V^i : \mathcal{S} \to \mathbb{R}$ defined on a finite state set $\mathcal{S}$, we can construct a state value vector $v^i(V^i) = [V^i(s)]_{s\in\mathcal{S}} \in \mathcal{V}^i := \mathbb{R}^{|\mathcal{S}|}$ by traversing all states. The infinity norm on $\mathcal{V}^i$ is $\|v^i(V^i)\|_\infty = \max_{s\in\mathcal{S}} |V^i(s)|$. Define the expected total reward of agent i in state s for π^i and χ^i as

$$f_s^i(v^i, \pi^i, \pi^{-i}, \chi^i, \chi^{-i}) = \sum_{\rho\in\mathcal{P}} \chi^i(\rho^i|s) \prod_{j\neq i} \chi^j(\rho^j|s) \sum_{a\in\mathcal{A}} \pi^i(a^i|\rho^i(s)) \prod_{j\neq i} \pi^j(a^j|\rho^j(s)) \left(r^i + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a) V^i(s') \right), \tag{14}$$

where π^{-i} and χ^{-i} denotes the agent policies and the adversary policies of all other agents except agent i.

Define the robust best response state value of agent i in state s given π^{-i} and χ^{-i} as a function $\psi_s^i : \mathcal{V}^i \to \mathbb{R}$,

$$\psi_s^i(v^i, \pi^{-i}, \chi^{-i}) = \max_{\pi^i} \min_{\chi^i} f_s^i(v^i, \pi^i, \pi^{-i}, \chi^i, \chi^{-i}). \tag{15}$$

Note that ψ_s^i gives a real number that denotes the expected total reward in state s for agent i given π^{-i} and χ^{-i}. We can construct a mapping $\Psi^i_{\pi,\chi} : \mathcal{V}^i \to \mathcal{V}^i$ from any state value vector v^i to a robust best response state value vector $[\Psi^i_{\pi,\chi}(v^i)]_{s\in\mathcal{S}}$ by traversing all s, that is to say, $[\Psi^i_{\pi,\chi}(v^i)]_s = \psi_s^i(v^i, \pi^{-i}, \chi^{-i})$. There exists a state value function V^i satisfying (13) if and only if $v^i(V^i)$ is a fixed point of $\Psi^i_{\pi,\chi}$.

Now we prove the function $\Psi^i_{\pi,\chi} : \mathcal{V}^i \to \mathcal{V}^i$ is a contraction mapping given any π^{-i} and χ^{-i}. Let us consider two vectors $v^i, z^i \in \mathcal{V}^i$. Given any π^{-i} and χ^{-i}, for all $i \in \mathcal{N}$ and $s \in \mathcal{S}$, we have

$$\begin{aligned} \psi_s^i(v^i, \pi^{-i}, \chi^{-i}) &= \max_{\pi^i} \min_{\chi^i} f_s^i(v^i, \pi^i, \pi^{-i}, \chi^i, \chi^{-i}) \\ &= f_s^i(v^i, \pi^{i*}, \pi^{-i}, \chi^{i*}, \chi^{-i}), \end{aligned} \tag{16}$$

where π^{i*} is the corresponding maximizer, and χ^{i*} is the corresponding optimizer for π^{i*}. Similarly, with ω^{i*} and φ_1^{i*}, we have

$$\begin{aligned} \psi_s^i(z^i, \pi^{-i}, \chi^{-i}) &= \max_{\omega^i} \min_{\varphi^i} f_s^i(z^i, \omega^i, \pi^{-i}, \varphi^i, \chi^{-i}) \\ &= f_s^i(z^i, \omega^{i*}, \pi^{-i}, \varphi_1^{i*}, \chi^{-i}) \\ &\geq f_s^i(z^i, \pi^{i*}, \pi^{-i}, \varphi_2^{i*}, \chi^{-i}), \end{aligned} \tag{17}$$

where

$$\varphi_2^{i*} = \arg\min_{\varphi^i} f_s^i(z^i, \pi^{i*}, \pi^{-i}, \varphi^i, \chi^{-i}). \tag{18}$$

Then, we have

$$\begin{aligned} &\psi_s^i(v^i, \pi^{-i}, \chi^{-i}) - \psi_s^i(z^i, \pi^{-i}, \chi^{-i}) \\ &= f_s^i(v^i, \pi^{i*}, \pi^{-i}, \chi^{i*}, \chi^{-i}) - f_s^i(z^i, \omega^{i*}, \pi^{-i}, \varphi_1^{i*}, \chi^{-i}) \\ &\leq f_s^i(v^i, \pi^{i*}, \pi^{-i}, \chi^{i*}, \chi^{-i}) - f_s^i(z^i, \pi^{i*}, \pi^{-i}, \varphi_2^{i*}, \chi^{-i}) \\ &\leq f_s^i(v^i, \pi^{i*}, \pi^{-i}, \varphi_2^{i*}, \chi^{-i}) - f_s^i(z^i, \pi^{i*}, \pi^{-i}, \varphi_2^{i*}, \chi^{-i}) \\ &= \sum_{\rho\in\mathcal{P}} \varphi_2^{i*}(\rho^i|s) \prod_{j\neq i} \chi^j(\rho^j|s) \sum_{a\in\mathcal{A}} \pi^{i*}(a^i|\rho^i(s)) \times \\ &\quad \prod_{j\neq i} \pi^j(a^j|\rho^j(s)) \left(r^i + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a) V^i(s') \right) \\ &\quad - \sum_{\rho\in\mathcal{P}} \varphi_2^{i*}(\rho^i|s) \prod_{j\neq i} \chi^j(\rho^j|s) \sum_{a\in\mathcal{A}} \pi^{i*}(a^i|\rho^i(s)) \times \\ &\quad \prod_{j\neq i} \pi^j(a^j|\rho^j(s)) \left(r^i + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a) Z^i(s') \right) \\ &= \sum_{\rho\in\mathcal{P}} \varphi_2^{i*}(\rho^i|s) \prod_{j\neq i} \chi^j(\rho^j|s) \sum_{a\in\mathcal{A}} \pi^{i*}(a^i|\rho^i(s)) \times \\ &\quad \prod_{j\neq i} \pi^j(a^j|\rho^j(s)) \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a) [V^i(s') - Z^i(s')] \\ &\leq \sum_{\rho\in\mathcal{P}} \varphi_2^{i*}(\rho^i|s) \prod_{j\neq i} \chi^j(\rho^j|s) \sum_{a\in\mathcal{A}} \pi^{i*}(a^i|\rho^i(s)) \times \\ &\quad \prod_{j\neq i} \pi^j(a^j|\rho^j(s)) \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a) \|v^i - z^i\|_\infty \\ &= \gamma \|v^i - z^i\|_\infty, \end{aligned} \tag{19}$$

where $Z^i(s')$ is the s' entry of the vector $z^i \in \mathcal{V}^i$.

The second inequality in Eq. (19) follows

$$\chi^{i*} = \arg\min_{\chi^i} f_s^i(v^i, \pi^{i*}, \pi^{-i}, \chi^i, \chi^{-i}). \tag{20}$$

Because given any π^{-i} and χ^{-i}, for all $i \in \mathcal{N}$ and $s \in \mathcal{S}$

$$\psi_s^i(v^i, \pi^{-i}, \chi^{-i}) - \psi_s^i(z^i, \pi^{-i}, \chi^{-i}) \leq \gamma \|v^i - z^i\|_\infty, \tag{21}$$

we have

$$\|\Psi^i_{\pi,\chi}(v^i) - \Psi^i_{\pi,\chi}(z^i)\|_\infty \leq \gamma \|v^i - z^i\|_\infty, \tag{22}$$

that is to say, the function $\Psi^i_{\pi,\chi}$ is a contraction mapping.

Because any finite dimensional normed vector space is complete [51], the $(\mathcal{V}^i, \|\cdot\|_\infty)$ is a complete Banach space. Also, given any π^{-i} and χ^{-i}, the function $\Psi^i_{\pi,\chi}$ is a contraction mapping. Therefore, by Banach's fixed point theorem, there is a unique fixed point v^i such that $\Psi^i_{\pi,\chi}(v^i) = v^i$. In other words, for any given π^{-i} and χ^{-i}, there exists a unique V^i such that

$$V^i(s) = \max_{\pi^i} \min_{\chi^i} f_s^i(v^i(V^i), \pi^i, \pi^{-i}, \chi^i, \chi^{-i}). \tag{23}$$

□

Before we show the existence of the robust perfect Nash equilibrium, let us first show the admissible state perturbation set is a finite set for each agent under Assumption 1.

Lemma 3. *For each agent, the admissible state perturbation set $\mathcal{P}^i$ is a finite set under Assumption 1.*

Proof. Recall the set of admissible state perturbation set for agent i is $\mathcal{P}^i := \{\rho^i : \|\rho^i(s) - s\| \leq d \text{ for all } s \in \mathcal{S}\}$ where $d \geq 0$ is a radius. For the finite state set $\mathcal{S}$, the total number of the states is $|\mathcal{S}|$, which is a natural number. We can write the state set as $\mathcal{S} = \{s_1, s_2, ..., s_{|\mathcal{S}|}\}$. For each $s_k \in \mathcal{S}$, the admissible perturbed state set $\{\rho^i(s_k) \in \mathcal{S} \mid \|\rho^i(s_k) - s_k\| \leq d\}$ is a finite set; denote the cardinality of this set as M_k which is a natural number. Then, we can calculate the total number of admissible state perturbation as $M_1 \times M_2 \times \cdots \times M_{|\mathcal{S}|}$, which

is a natural number. Therefore, for each agent, the admissible state perturbation set $\mathcal{P}^i$ is a finite set under Assumption 1. □

Finally, we present the main result for the existence of the robust perfect Nash equilibrium defined in Definition 2.

Theorem 4. *For finite state and finite action space, the robust perfect Nash equilibrium defined in Definition 2 exists.*

Proof. Let us construct a $2n$ player game for any $s \in \mathcal{S}$. We have n agents and n adversaries in the player set. The action set for the n agents is $\mathcal{A} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^n$. The action set for the n adversaries is $\mathcal{P} = \mathcal{P}^1 \times \cdots \times \mathcal{P}^n$. We consider the mixed strategy $\pi^i \in \Delta(\mathcal{A}^i)$, $\chi^i \in \Delta(\mathcal{P}^i)$ for each agent i and each adversary i respectively.

Let us introduce uniform notations for the agents and adversaries to describe this $2n$ player game. The player set $\mathcal{I} = \{1, ..., n, n+1, ..., 2n\}$. The set of available actions for player i is

$$A^i = \begin{cases} \mathcal{A}^i, & i = 1, ..., n; \\ \mathcal{P}^{i-n}, & i = n+1, ..., 2n. \end{cases} \tag{24}$$

We consider the mixed strategy $\sigma^i \in \Delta(A^i)$ for player i. Note that the mixed strategy

$$\sigma^i = \begin{cases} \pi^i, & i = 1, ..., n; \\ \chi^{i-n}, & i = n+1, ..., 2n. \end{cases} \tag{25}$$

The utility function for player i is

$$u^i(\sigma^i, \sigma^{-i}) = \begin{cases} f_s^i(v^{i*}, \pi^i, \pi^{-i}, \chi^i, \chi^{-i}), \\ \text{for } i = 1, ..., n; \\ -f_s^{i-n}(v^{(i-n)*}, \pi^{i-n}, \pi^{-(i-n)}, \chi^{i-n}, \chi^{-(i-n)}), \\ \text{for } i = n+1, ..., 2n. \end{cases} \tag{26}$$

where σ^{-i} denotes the strategies of all other players except player i, the function f_s^i is defined in Eq. (14), and v^{i*} satisfies

$$v_s^{i*} = \max_{\pi^i} \min_{\chi^i} f_s^i(v^{i*}, \pi^i, \pi^{-i}, \chi^i, \chi^{-i}). \tag{27}$$

According to Theorem 2, there is a unique robust best response state value function $V^i(s)$ such that

$$v_s^{i*}(V^i) = \max_{\pi^i} \min_{\chi^i} f_s^i(v^{i*}(V^i), \pi^i, \pi^{-i}, \chi^i, \chi^{-i}) \tag{28}$$

given any σ^{-i} (it includes both π^{-i} and χ^{-i} for all $i \in \mathcal{I}$). In other words, there is a unique vector v^{i*} satisfying Eq. (28) given σ^{-i}. Thus, the utility function is well-defined.

According to Lemma 3, $\mathcal{P}^i$ is a finite set for all $i \in \mathcal{N}$. Also, $\mathcal{A}^i$ is a finite set for all $i \in \mathcal{N}$. Therefore, $\Delta(A^i)$ is compact and convex for all $i \in \mathcal{I}$. Moreover, for all $i \in \mathcal{I}$, $u^i(\sigma^i, \sigma^{-i})$ is continuous in σ^{-i}; $u^i(\sigma^i, \sigma^{-i})$ is linear in σ^i and therefore continuous and concave in σ^i. According to the theorem (Debreu [36], Glicksberg [37], Fan [38]), the conditions for the existence of a Nash Equilibrium are satisfied, hence, there exists a Nash equilibrium for this $2n$ player game.

□

Corollary 4.1. *When the robust perfect Nash equilibrium, a joint agent policy $\pi^* = (\pi^{1*}, \pi^{2*}, ..., \pi^{n*})$ and joint adversary policy $\chi^* = (\chi^{1*}, \chi^{2*}, ..., \chi^{n*})$ defined in Definition 2, exists, there exists a vector-valued function $Q^* = (Q^{1*}, Q^{2*}, ..., Q^{n*})$ with each $Q^{i*} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, such that for all $i \in \mathcal{N}$, $s \in \mathcal{S}$, and $a \in \mathcal{A}$,*

$$\begin{aligned} Q^{i*}(s,a) = \max_{\pi^i} \min_{\chi^i} r^i(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) \\ \sum_{\rho \in \mathcal{P}} \chi^i(\rho^i|s') \prod_{j \neq i} \chi^{j*}(\rho^j|s') \sum_{a' \in \mathcal{A}} \pi^i(a^{i\prime}|\rho^i(s')) \\ \prod_{j \neq i} \pi^{j*}(a^{j\prime}|\rho^j(s')) Q^{i*}(s', a'), \end{aligned} \tag{29}$$

and (π^{i*}, χ^{i*}) is the solution policy for (29)*.*

Proof. For any $i \in \mathcal{N}, s \in \mathcal{S}, a \in \mathcal{A}$, we have

$$Q^i(s,a) = \left(r^i(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^i(s') \right). \tag{30}$$

□

REFERENCES

[1] M. Hüttenrauch and A. Šošić, "Guided deep reinforcement learning for swarm systems," *arXiv preprint arXiv:1709.06011*, 2017.

[2] S. Sukhbaatar, A. Szlam, and R. Fergus, "Learning multiagent communication with backpropagation," in *NeurIPS*, 2016.

[3] J. Z. Leibo and V. Zambaldi, "Multi-agent reinforcement learning in sequential social dilemmas," in *AAMAS*, 2017, pp. 464–473.

[4] A. Pretorius, S. Cameron *et al.*, "A game-theoretic analysis of networked system control for common-pool resource management using multi-agent reinforcement learning," in *NeurIPS*, vol. 33, 2020, pp. 9983–9994.

[5] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *NeurIPS*, 2017, pp. 6379–6390.

[6] J. Foerster and G. Farquhar, "Counterfactual multi-agent policy gradients," in *AAAI*, 2018.

[7] O. Kilinc, "Multi-agent deep reinforcement learning with extremely noisy observations," *arXiv preprint arXiv:1812.00922*, 2018.

[8] L. Pinto, J. Davidson, and R. Sukthankar, "Robust adversarial reinforcement learning," in *ICML*. PMLR, 2017, pp. 2817–2826.

[9] H. Zhang, H. Chen, C. Xiao, B. Li, M. Liu, D. Boning, and C.-J. Hsieh, "Robust deep reinforcement learning against adversarial perturbations on state observations," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 024–21 037, 2020.

[10] C. Xiao, X. Pan *et al.*, "Characterizing attacks on deep reinforcement learning," *arXiv preprint arXiv:1907.09470*, 2019.

[11] A. Pattanaik and Z. Tang, "Robust deep reinforcement learning with adversarial attacks," *AAMAS*, 2017.

[12] V. Behzadan and A. Munir, "Vulnerability of deep reinforcement learning to policy induction attacks," in *MLDM*. Springer, 2017, pp. 262–275.

[13] Z. Liu, Y. Cai, H. Wang, L. Chen, H. Gao, Y. Jia, and Y. Li, "Robust target recognition and tracking of self-driving cars with radar and camera information fusion under severe weather conditions," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

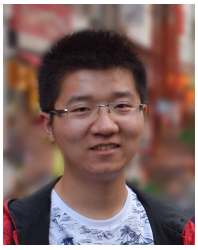
[14] D. Kothandaraman, R. Chandra, and D. Manocha, "Ss-sfda: Self-supervised source-free domain adaptation for road segmentation in hazardous environments," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3049–3059.

[15] Y. Jiang, T. Zhang, D. Ho, Y. Bai, C. K. Liu, S. Levine, and J. Tan, "Simgan: Hybrid simulator identification for domain adaptation via adversarial reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2884–2890.

[16] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.

[17] S. Han, H. Wang, S. Su, Y. Shi, and F. Miao, "Stable and efficient shapley value-based reward reallocation for multi-agent reinforcement learning of autonomous vehicles," *arXiv preprint arXiv:2203.06333*, 2022.

[18] T. Lin, C. Jin, and M. Jordan, "On gradient descent ascent for nonconvex-concave minimax problems," in *ICML*. PMLR, 2020, pp. 6083–6093.
[19] J. Hu, M. P. Wellman *et al.*, "Multiagent reinforcement learning: theoretical framework and an algorithm." in *ICML*, vol. 98. Citeseer, 1998, pp. 242–250.
[20] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 38, no. 2, pp. 156–172, 2008.
[21] P. Sunehag, G. Lever *et al.*, "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *AAMAS*, 2018, pp. 2085–2087.
[22] T. Rashid, G. Farquhar, B. Peng, and S. Whiteson, "Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning," in *NeurIPS*, December 2020.
[23] M. Zhou, Z. Liu, P. Sui, Y. Li, and Y. Y. Chung, "Learning implicit credit assignment for cooperative multi-agent reinforcement learning," in *NeurIPS*, vol. 33, 2020, pp. 11 853–11 864.
[24] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *ICML*. PMLR, 2019, pp. 2961–2970.
[25] G. Qu, Y. Lin, A. Wierman, and N. Li, "Scalable multi-agent reinforcement learning for networked systems with average reward," in *NeurIPS*, vol. 33, 2020, pp. 2074–2086.
[26] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.
[27] J. Morimoto and K. Doya, "Robust reinforcement learning," *Neural computation*, vol. 17, no. 2, pp. 335–359, 2005.
[28] K. Zhang, T. Sun, Y. Tao, S. Genc, S. Mallya, and T. Basar, "Robust multi-agent reinforcement learning with model uncertainty." in *NeurIPS*, 2020.
[29] A. Sinha, M. O'Kelly *et al.*, "Formulazero: Distributionally robust online adaptation via offline population synthesis," in *ICML*. PMLR, 2020, pp. 8992–9004.
[30] J. Yu, C. Gehring, F. Schäfer, and A. Anandkumar, "Robust reinforcement learning: A constrained game-theoretic approach," in *Learning for Dynamics and Control*. PMLR, 2021, pp. 1242–1254.
[31] M. Shen and J. P. How, "Robust opponent modeling via adversarial ensemble reinforcement learning," in *ICAPS*, vol. 31, 2021, pp. 578–587.
[32] S. Li, Y. Wu, X. Cui, H. Dong, F. Fang, and S. Russell, "Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient," in *AAAI*, vol. 33, 2019, pp. 4213–4220.
[33] C. Sun, D.-K. Kim, and J. P. How, "Romax: Certifiably robust deep multiagent reinforcement learning via convex relaxation," *arXiv preprint arXiv:2109.06795*, 2021.
[34] H. Zhang, H. Chen, D. Boning, and C.-J. Hsieh, "Robust reinforcement learning on state observations with learned optimal adversary," *arXiv preprint arXiv:2101.08452*, 2021.
[35] J. Nash, "Non-cooperative games," *Annals of mathematics*, pp. 286–295, 1951.
[36] G. Debreu, "A social equilibrium existence theorem," *Proceedings of the National Academy of Sciences*, vol. 38, no. 10, pp. 886–893, 1952.
[37] I. L. Glicksberg, "A further generalization of the kakutani fixed point theorem, with application to nash equilibrium points," *Proceedings of the American Mathematical Society*, vol. 3, no. 1, pp. 170–174, 1952.
[38] K. Fan, "Fixed-point and minimax theorems in locally convex topological linear spaces," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 38, no. 2, p. 121, 1952.
[39] L. S. Shapley, "Stochastic games," *Proceedings of the national academy of sciences*, vol. 39, no. 10, pp. 1095–1100, 1953.
[40] A. M. Fink, "Equilibrium in a stochastic n-person game," *Journal of science of the hiroshima university, series ai (mathematics)*, vol. 28, no. 1, pp. 89–93, 1964.
[41] A. Nilim and L. El Ghaoui, "Robust control of markov decision processes with uncertain transition matrices," *Operations Research*, vol. 53, no. 5, pp. 780–798, 2005.
[42] G. N. Iyengar, "Robust dynamic programming," *Mathematics of Operations Research*, vol. 30, no. 2, pp. 257–280, 2005.
[43] E. Kardeş, F. Ordóñez, and R. W. Hall, "Discounted robust stochastic games and an application to queueing control," *Operations research*, vol. 59, no. 2, pp. 365–382, 2011.
[44] M. Everett, B. Lütjens, and J. P. How, "Certifiable robustness to adversarial state uncertainty in deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
[45] F. A. Oliehoek, C. Amato *et al.*, *A concise introduction to decentralized POMDPs*. Springer, 2016, vol. 1.
[46] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou, "The complexity of computing a nash equilibrium," *SIAM Journal on Computing*, vol. 39, no. 1, pp. 195–259, 2009.
[47] V. Mnih, K. Kavukcuoglu *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[48] M. Razaviyayn, T. Huang, S. Lu, M. Nouiehed, M. Sanjabi, and M. Hong, "Nonconvex min-max optimization: Applications, challenges, and recent theoretical advances," *IEEE Signal Processing Magazine*, vol. 37, no. 5, pp. 55–66, 2020.
[49] 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.
[50] D. Guo, L. Tang, X. Zhang, and Y.-C. Liang, "Joint optimization of handover control and power allocation based on multi-agent deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13 124–13 138, 2020.
[51] E. Kreyszig, *Introductory functional analysis with applications*. John Wiley & Sons, 1991, vol. 17.

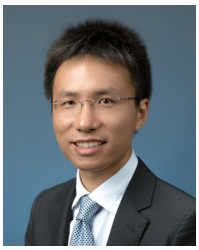


Songyang Han (S'17) received the B.E. degree in automation from Nanjing University, Nanjing, China, in 2015, and the M.S. degree in electrical and computer engineering from the University of Michigan-Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, Shanghai, China, in 2018. He is currently working toward the Ph.D. degree in computer science and engineering at the University of Connecticut, Storrs, CT, USA.

His current research interests include autonomous driving, reinforcement learning, game theory, optimization and optimal control. He is the recipient of the Best Paper Award at the 12th ACM/IEEE International Conference on Cyber-Physical Systems.

Sanbao Su received the bachelor's degree in automation from Nanjing University, Nanjing, China, in 2016, and the master's degree in Electronic Science and Technology from the University of Michigan-Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, Shanghai, China, in 2019. He is currently pursuing the Ph.D. degree in Computer Science and Engineering with the University of Connecticut, Storrs, Mansfield, CT, USA. His current research includes reinforcement learning, and optimal control.

Sihong He is currently a PhD student in Computer Science and Engineering at University of Connecticut, Storrs, CT, USA. She received her bachelor's degree in Financial Mathematics from the Department of Mathematics, Southern University of Science and Technology, Shenzhen, China, in 2017. After that, she obtained her master's degree in Statistics from the Donald Bren School of Information and Computer Sciences, the University of California, Irvine, CA, USA in 2019. Her current research interests include data-driven optimization, robust optimization, reinforcement learning, machine learning, etc.

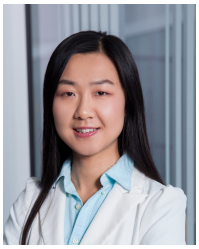


Shuo Han (S'08-M'14) received the B.E. and M.E. degrees in Electronic Engineering from Tsinghua University, Beijing, China in 2003 and 2006, and the Ph.D. degree in Electrical Engineering from the California Institute of Technology, Pasadena, CA, USA in 2013.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering at the University of Illinois, Chicago, IL, USA. Previously, he was a postdoctoral researcher with the Department of Electrical and Systems Engineering at the University of Pennsylvania, Philadelphia, PA, USA. His research interests are control theory and optimization algorithms with applications to distributed and multi-agent systems.

Haizhao Yang received the B.S. degree in mathematics from Shanghai Jiao Tong University, Shanghai, China, in 2010, and the M.A. degree in mathematics from the University of Texas at Austin, and the Ph.D. degree in math from Stanford University, Stanford, CA, USA, in 2015.

He is currently an Assistant Professor in the Department of Mathematics at Purdue University, West Lafayette, IN, USA. Previously, he was an assistant professor at the National University of Singapore and a visiting professor at Duke University. His research interests include machine learning theory and scientific computing. Dr. Yang is the recipient of the NSF CAREER Award and the ONR Young Investigator Award.



Fei Miao (S'13-M'16) received the B.S. degree in automation from Shanghai Jiao Tong University, Shanghai, China, in 2010, and the M.A. degree in statistics and the Ph.D. degree with the "Charles Hallac and Sarah Keil Wolf Award for Best Doctoral Dissertation" in electrical and systems engineering both from the University of Pennsylvania, Philadelphia, PA, USA, in 2015 and 2016.

She is currently an Assistant Professor in the Department of Computer Science and Engineering at the University of Connecticut, Storrs, CT, USA. Previously, she was a Postdoc Researcher in the Department of Electrical and Systems Engineering at the University of Pennsylvania. Her research interests include learning and control of cyber-physical systems under model uncertainties, and CPS security. Dr. Miao is the recipient of the NSF CAREER Award, and Best Paper Award Finalist at the 6th and 12th ACM/IEEE International Conference on Cyber-Physical Systems.