Robustness to Multi-Modal Environment Uncertainty in MARL using Curriculum Learning

Aakriti Agrawal[*]
agrawal5@umd.edu

Rohith Aralikatti [*]
rohithca@umd.edu

Yanchao Sun [*]
ycs@umd.edu

Furong Huang [*]
furongh@umd.edu

Abstract

Multi-agent reinforcement learning (MARL) plays a pivotal role in tackling real-world challenges. However, the seamless transition of trained policies from simulations to real-world requires it to be robust to various environmental uncertainties. Existing works focus on finding Nash Equilibrium or the optimal policy under uncertainty in a single environment variable (i.e. action, state or reward). This is because a multi-agent system is highly complex and non-stationary. However, in a real-world setting, uncertainty can occur in multiple environment variables simultaneously. This work is the first to formulate the generalised problem of robustness to multi-modal environment uncertainty in MARL. To this end, we propose a general robust training approach for multi-modal uncertainty based on curriculum learning techniques. We handle environmental uncertainty in more than one variable simultaneously and present extensive results across both cooperative and competitive MARL environments, demonstrating that our approach achieves state-of-the-art robustness on three multi-particle environment tasks (Cooperative-Navigation, Keep-Away, Physical Deception).

1 Introduction

MARL has excelled in tackling intricate real-world decision-making challenges, from robotics (e.g., path planning [1], task allocation [2; 3]) to traffic management [4] and Game Theory and Economics [5]. In MARL [6], agents aim to maximize their long-term returns by interacting with both the environment and other agents in a shared setting, thus making it more complex than single-agent RL. In MARL, finding the Nash Equilibrium (NE) is a popular solution concept [7].

Real-world MARL applications often involve training agents in simulations and deploying them in dynamic environments where accurate knowledge may be lacking. This could result from shifts in environmental parameters, information noise (e.g., imprecise state, action, or reward data transfer), or hardware constraints. Such situations give rise to environmental uncertainty, which has been addressed in previous work. However, in existing literature, the effect of uncertainty on reward, transition dynamics [8], state [9], or action [10] has been studied independently. In practical scenarios, multiple environmental parameters may exhibit uncertainty simultaneously, necessitating the development of robustness to multi-modal uncertainty.

[*] Agrawal, Aralikatti, Sun and Huang are from University of Maryland, CP. Agrawal, Sun and Huang are supported by National Science Foundation NSF-IISFAI program, DOD-ONR-Office of Naval Research, DOD Air Force Office of Scientific Research, DOD-DARPA-Defense Advanced Research Projects Agency Guaranteeing AI Robustness against Deception (GARD), Adobe, Capital One and JP Morgan faculty fellowships.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Contributions. In this work, **(1)** We address the challenge of handling multi-modal uncertainty in MARL by developing robustness to two uncertain parameters simultaneously, marking the first work in this domain. **(2)** We also define and theoretically formalize the general problem of finding optimal policy in MARL with multi-modal uncertainty. **(3)** We tackle the complexity of finding NE for this generalised problem and propose an efficient curriculum learning (CL) approach to address this challenge. **(4)** We also show experimentally that our method is able to find optimal solution for the given robust Markov game~~s~~ and generate state-of-the-art robustness for reward, state and action uncertainty on three multi-particle environment tasks (Cooperative-Navigation, Keep-Away, Physical Deception). **(5)** As a by-product, this is the also the first work to handle action uncertainty in MARL.

Related Work. Robustness in RL is crucial for countering adversarial attacks [11] and addressing uncertainty in model/environment parameters. In single-agent RL, robustness to uncertainty is handled by maximin optimization, casting it as a zero-sum game between the agent and the uncertainty set [12; 13; 14; 15]. In MARL, uncertainty is defined as a robust Markov game and agents independently maximize their returns while navigating uncertainty. While MARL research has made significant progress, there is limited work dedicated to handling uncertainty.[16] focuses on multi-modal uncertainty (reward and transition dynamics), albeit primarily in a theoretical context. Some works address single uncertainties, such as reward, transition dynamics [8], or state [9] uncertainty. Additionally, studies like [17; 18] enhance robustness by training agents to handle worst-case actions from adversarial agents in competitive settings. Other literature in the action uncertainty space mainly focus on cooperative environments [19; 20; 21]. In the pursuit of enhancing robustness, **CL** [22; 23] is widely applied across diverse domains, such as object classification [24], automatic speech recognition (ASR) [25] etc. [26] aligns closely with our work, employing CL to enhance robustness in RL. Notably, our work represents the first instance of CL being used to address multi-modal uncertainty, and also first in the context of MARL.

2 Robust Markov Game

The interaction among multiple agents is modeled as a **Markov game** $\mathcal{G}$ [27]. Real-world scenarios often involve uncertainty, affecting various components such as reward, state, action, and transition probability. Hence, we define **Robust Markov game** [28] as,

$$\bar{\mathcal{G}}_{general} = \langle \mathcal{N}, \{\mathcal{S}^i\}_{i\in\mathcal{N}}, \{\bar{\mathcal{O}}^i\}_{i\in\mathcal{N}}, \{\mathcal{A}^i\}_{i\in\mathcal{N}}, \{\bar{\mathrm{A}}^i\}_{i\in\mathcal{N}}, \{\bar{\mathcal{R}}^i_s\}_{(i,s)\in\mathcal{N}\times\mathcal{S}}, \{\bar{\mathcal{P}}_s\}_{s\in\mathcal{S}}, \gamma\rangle, \tag{1}$$

where $\mathcal{N}$, $\mathcal{S}^i$, $\mathcal{A}^i$, and $\gamma \in [0, 1)$ denote the set of agents, the state space, the action space for each agent i, and the discounting factor, respectively. $\bar{\mathcal{R}}^i_s \in \mathbb{R}^{|A|}$ and $\bar{\mathcal{P}}_s$ denotes the uncertainty sets of all possible reward function values and that of all possible transition probabilities at state s. $\bar{\mathcal{O}}^i$ and $\bar{\mathrm{A}}^i$ denotes the uncertainty sets of all possible values of perturbed state $\bar{s}^i$ and $\bar{a}^i$ respectively. The state space and action space of the uncertainty sets is similar to that of the true state s and true action a respectively. $\gamma \in [0, 1)$ is the discounting factor. Note that the state perturbation reflects the state uncertainty from the perspective of each agent so it does not change the true state of multi-agent systems. Each agent is associated with a policy $\pi^i : S \to \Delta(A^i)$ to choose an action $a^i \in A^i$ given the perturbed state $\bar{s}$. The agents' joint policy $\pi = \Pi_{i\in\mathcal{N}}\pi^i : \mathcal{S} \to \Delta(\mathcal{A})$.

In RL, there are three primary sources of aleatoric uncertainty [29]: stochastic rewards, stochastic observations, and stochastic actions. Note that stochastic observations can arise from both uncertain transition dynamics and inherent observation noise. **Stochastic Rewards:** We denote perturbed rewards as $\bar{R}^i(s, a) = \mathcal{N}_{trunc}(R^i(s, a), \epsilon)$, where R^i and $\bar{R}^i$ represent the true and perturbed rewards for agent i. **Stochastic Observations:** Perturbed states are defined as $\bar{s}^i = \mathcal{N}_{trunc}(s^i, \mu)$, with s^i and $\bar{s}^i$ representing true and perturbed states. **Stochastic Actions:** Perturbed actions are expressed as $\bar{a}^i = \mathcal{N}_{trunc}(a^i, \nu)$, where a^i and $\bar{a}^i$ denote true and perturbed actions. In these equations, ϵ, μ, ν denote the standard deviations. Higher ϵ, μ, ν implies more uncertainty. $\mathcal{N}_{trunc}$ refers to a truncated normal distribution, with truncation limits set at 2ϵ, 2μ, and 2ν respectively.

In appendix section 6.2 we define the solution for the general robust Markov game in equation 1. We also show proof for
Theorem 1: Existence of robust Nash equilibrium implies existence of optimal value function.

3 CL based robustness to multi-modal uncertainty

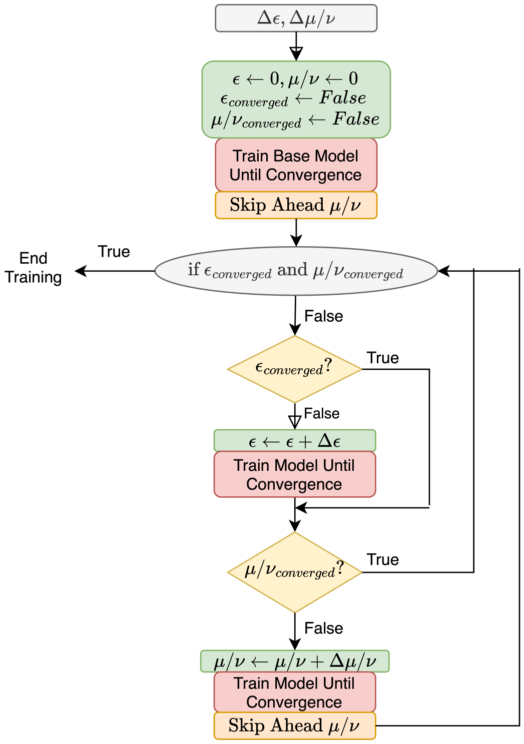

$\Delta\epsilon, \Delta\mu/\nu$
$\epsilon \leftarrow 0, \mu/\nu \leftarrow 0$
$\epsilon_{converged} \leftarrow False$
$\mu/\nu_{converged} \leftarrow False$
Train Base Model Until Convergence
Skip Ahead μ/ν
End Training
True
if $\epsilon_{converged}$ and $\mu/\nu_{converged}$
False
$\epsilon_{converged}$?
True
False
$\epsilon \leftarrow \epsilon + \Delta\epsilon$
Train Model Until Convergence
$\mu/\nu_{converged}$?
True
False
$\mu/\nu \leftarrow \mu/\nu + \Delta\mu/\nu$
Train Model Until Convergence
Skip Ahead μ/ν

Figure 1: Multi-modal CL algorithm with look-ahead

CL [22; 23] optimizes the order of experience accumulation to enhance performance or training speed on a set of final tasks. Measuring task difficulty is a key challenge in curriculum design, and we use noise parameters (ϵ, μ, and ν) to increase task complexity. This allows us to leverage knowledge gained from simpler tasks (lower uncertainty) for faster learning on more complex ones (higher uncertainty). From the curriculum learning training plots (refer to figures 4, 6, 7, 8 and 10), it is observed that complex tasks use the knowledge gained from simpler tasks and hence converge in fewer iterations as compared to training from scratch on the complex task. Our base model is from [8]. **Efficient Lookahead CL Algorithm** is designed for single-parameter uncertainty scenarios. In each CL iteration, the model trains on a specific noise parameter and evaluates at higher noise levels, enabling it to skip already-learned noise values and streamline CL training. **Efficient Lookahead CL for Multiple Uncertainties** The algorithm for curriculum learning with combined reward and state/action perturbations is outlined in 1 and flowchart in 1. Our goal is to efficiently increase both reward and state/action uncertainty in each CL iteration, ultimately training a model capable of handling two uncertainties simultaneously. Note that there is no skip-ahead in the reward uncertainty parameter, as reward uncertainty is not present during evaluation. Similarly, algorithms can be defined for action and reward perturbations.

4 Experiments

In this section, we present results for our curriculum learning-based method on three multi-particle environments (cooperative navigation, keep-away, and physical deception) and compare them with state-of-the-art robustness in those environments. For cooperative navigation, we present detailed results in Table 1. Experimental results for keep-away and physical deception are shown in Fig. 2. We start by comparing the base method (without CL) with our CL method in each environment under varying levels of reward, state, or action uncertainty. Then, we explore dual uncertainty combinations: state + reward, action + reward, and state + action. Notably, this paper is first to handle multi-modal uncertainty in MARL. We report values related to action and state uncertainty by evaluating the trained model 1000 times, reporting its mean and standard deviation. However, for reward uncertainty, we illustrate training plots since rewards aren't involved in evaluation, making reward uncertainty during evaluation irrelevant.

Cooperative Navigation Environment. In this cooperative setting, three agents aim to occupy all three landmarks while avoiding collisions. **Robustness under uncertainty in a single parameter:** Regarding **reward uncertainty**, the success rate (see Figure 3a) shows that the model's learning extends up to $\epsilon = 9$, which matches the current state-of-the-art robustness [8]. However, our lookahead-CL approach achieves robustness up to $\epsilon = 47$ (reward plot in Figure 4a). For **state uncertainty**, the model's learning reaches $\mu = 0.5$, but with CL, it reaches $\mu = 1.1$ (reward plot in Figure 4b,). In the case of **action uncertainty**, the model's learning goes up to $\nu = 2.0$, and with CL, it reaches $\nu = 2.2$ (check Figure 3c). This is the first work addressing action uncertainty, thus lacking a baseline for comparison. Regarding state uncertainty, [9] does not provide comparisons for various uncertainty values. Table 1 shows detailed comparison between CL and the baseline demonstrating that CL achieves state-of-the-art robustness.

Reward Uncertainty								
ϵ	6	**9**	10	11	12	15	**47**	48
Baseline	✓	✓	×	×	×	×	×	×
CL	✓	✓	✓	✓	✓	✓	✓	×
State Uncertainty								
μ	0.25	0.45	**0.5**	0.55	0.75	1.0	**1.1**	1.2
Baseline	1 ± 0.04	0.97 ± 0.15	**1** ± **0.1**	0.6 ± 0.5	-	-	-	-
CL	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0.1	0.98 ± 0.2	**0.94** ± **0.2**	0.82 ± 0.4
Action Uncertainty								
ν	1.0	1.2	1.4	1.6	1.8	**2.0**	**2.2**	2.4
Baseline	1 ± 0.13	1 ± 0	1 ± 0	1 ± 0	1 ± 0.1	**0.93** ± **0.26**	0.78 ± 0.41	-
CL	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0.1	1 ± 0	**1** ± **0.03**	0.9 ± 0.04
Reward + State Uncertainty								
μ	0.5	0.55	0.6	0.65	**0.7**	0.75	0.8	0.85
	0.97 ± 0.2	0.95 ± 0.2	0.95 ± 0.2	0.96 ± 0.2	**0.95** ± **0.2**	0.94 ± 0.23	0.91 ± 0.3	0.81 ± 0.4
Reward + Action Uncertainty								
ν	1	1.2	1.4	1.6	1.8	2.0	2.2	**2.4**
	0.96 ± 0.2	0.96 ± 0.2	0.97 ± 0.2	0.97 ± 0.2	0.97 ± 0.2	0.96 ± 0.2	0.96 ± 0.2	**0.96** ± **0.2**
State+Action Uncertainty								
μ - ν	0	1	1.4	1.6	2	2.4	2.6	3
0	1 ± 0	1 ± 0.6	1 ± 0.04	1 ± 0.03	1 ± 0.04	**1** ± **0.05**	**1** ± **0.06**	**0.96** ± **0.18**
0.6	1 ± 0.6	0.98 ± 0.1	0.98 ± 0.14	**0.96** ± **0.2**	**0.92** ± **0.28**	0.8 ± 0.4	0.73 ± 0.4	0.6 ± 0.5
0.8	0.98 ± 0.14	**0.95** ± **0.2**	**0.9** ± **0.3**	0.85 ± 0.35	0.8 ± 0.45	0.6 ± 0.5	0.5 ± 0.5	0.4 ± 0.5
1	**0.93** ± **0.25**	0.8 ± 0.4	0.75 ± 0.43	0.67 ± 0.5	0.6 ± 0.5	0.42 ± 0.5	0.36 ± 0.5	0.26 ± 0.23

Table 1: The table shows a detailed comparison between baseline, CL for single and dual uncertainty using success rates at various values of noise. An episode is successful if all landmarks are occupied by agents. The first three blocks in the table are for single uncertainty, followed by three blocks for dual uncertainty combinations.

Multi-modal CL combining two uncertainty: We explore the combinations, reward + state, reward + action, and action + state, uncertainity to assess the effectiveness of CL when dealing with dual uncertainties. Refer to Table 1 for detailed results. In the case of **reward + state** uncertainty , our model successfully learns up to $\mu = 0.7$ when $\epsilon = 0$ and $\epsilon = 29$ during training. Refer to reward plots in Fig. 6. For **reward + action** uncertainty the model achieves learning up to $\nu = 2.4$ when $\epsilon = 0$ and $\epsilon = 50$ during training (reward plots in Figure 7). Finally, in the case of **action + state** uncertainty the model learns up to $\nu = 3$ when $\mu = 0$ and $\mu = 1$ when $\nu = 0$ (reward plots in Figure 8). This demonstrates that even when faced with two uncertain parameter, our method surpasses the baseline performance.

Other Environments: Figure 2 shows results for **Keep Away** (competitive) and **Physical Deception** (competitive and cooperative) environments. We observe that with both multi-parameter and single-parameter CL outperforms the baseline.

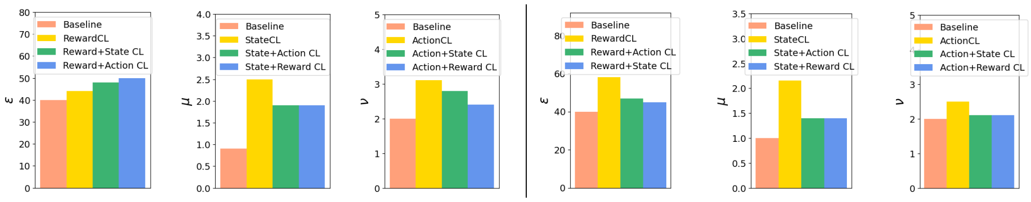


Figure 2: Highest noise values that result in convergence for **Keep Away** (left) and **Physical Deception** (right). For each environment, we show three figures in order - Reward, State and Action uncertainty. For the above environments, success is defined as the agent reaching its goal within 100 steps, with model convergence defined as a success rate exceeding 90%. Noise is only added to the agent's reward, state and action.

5 Conclusion

We explore curriculum learning to enhance the robustness of a MARL model in the presence of multi-modal environmental uncertainty. We devise an efficient curriculum that ultimately achieves state-of-the-art robustness on three multi-particle environment tasks (Cooperative-Navigation, Keep-Away, Physical Deception). Our approach outperforms the baseline across various forms of uncertainty, including state, reward, and action uncertainty, both in single and multi-modal (dual) uncertainty scenarios. This research holds significant promise for applications in sim-to-real scenarios. As future work, we plan to evaluate the model's performance in sim-to-real settings.

References

[1] Qingqing Wang, Hong Liu, Kaizhou Gao, and Le Zhang. Improved multi-agent reinforcement learning for path planning-based crowd simulation. *IEEE Access*, 7:73841–73855, 2019. 1

[2] Aakriti Agrawal, Amrit Singh Bedi, and Dinesh Manocha. Rtaw: An attention inspired reinforcement learning method for multi-robot task allocation in warehouse environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1393–1399. IEEE, 2023. 1

[3] Aakriti Agrawal, Senthil Hariharan, Amrit Singh Bedi, and Dinesh Manocha. Dc-mrta: Decentralized multi-robot task allocation and navigation in complex environments. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11711–11718. IEEE, 2022. 1

[4] Máté Kolat, Bálint Kővári, Tamás Bécsi, and Szilárd Aradi. Multi-agent reinforcement learning for traffic signal control: A cooperative approach. *Sustainability*, 15(4):3479, 2023. 1

[5] Ann Nowé, Peter Vrancx, and Yann-Michaël De Hauwere. Game theory and multi-agent reinforcement learning. *Reinforcement Learning: State-of-the-Art*, pages 441–470, 2012. 1

[6] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pages 321–384, 2021. 1

[7] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008. 1

[8] Kaiqing Zhang, Tao Sun, Yunzhe Tao, Sahika Genc, Sunil Mallya, and Tamer Basar. Robust multi-agent reinforcement learning with model uncertainty. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2020. 1, 2, 3, 7, 9

[9] Songyang Han, Sanbao Su, Sihong He, Shuo Han, Haizhao Yang, and Fei Miao. What is the solution for state adversarial multi-agent reinforcement learning? *arXiv preprint arXiv:2212.02705*, 2022. 1, 2, 3, 7, 9

[10] Shihui Li, Yi Wu, Xinyue Cui, Honghua Dong, Fei Fang, and Stuart Russell. Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4213–4220, 2019. 1, 7

[11] Yongyuan Liang, Yanchao Sun, Ruijie Zheng, and Furong Huang. Efficient adversarial training without attacking: Worst-case-aware robust reinforcement learning. *Advances in Neural Information Processing Systems*, 35:22547–22561, 2022. 2

[12] Chen Tessler, Yonathan Efroni, and Shie Mannor. Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning*, pages 6215–6224. PMLR, 2019. 2

[13] Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. Robust markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013. 2

[14] Annie Xie, Shagun Sodhani, Chelsea Finn, Joelle Pineau, and Amy Zhang. Robust policy learning over multiple uncertainty sets. In *International Conference on Machine Learning*, pages 24414–24429. PMLR, 2022. 2

[15] Arnab Nilim and Laurent El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005. 2

[16] Erim Kardeş, Fernando Ordóñez, and Randolph W Hall. Discounted robust stochastic games and an application to queueing control. *Operations research*, 59(2):365–382, 2011. 2, 11

[17] Shihui Li, Yi Wu, Xinyue Cui, Honghua Dong, Fei Fang, and Stuart Russell. Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019. 2

[18] Chuangchuang Sun, Dong-Ki Kim, and Jonathan P How. Romax: Certifiably robust deep multiagent reinforcement learning via convex relaxation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5503–5510. IEEE, 2022. 2

[19] Eleni Nisioti, Daan Bloembergen, and Michael Kaisers. Robust multi-agent q-learning in cooperative games with adversaries. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. 2

[20] Thomy Phan, Thomas Gabor, Andreas Sedlmeier, Fabian Ritz, Bernhard Kempter, Cornel Klein, Horst Sauer, Reiner Schmid, Jan Wieghardt, Marc Zeller, et al. Learning and testing resilience in cooperative multi-agent systems. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1055–1063, 2020. 2

[21] Thomy Phan, Lenz Belzner, Thomas Gabor, Andreas Sedlmeier, Fabian Ritz, and Claudia Linnhoff-Popien. Resilient multi-agent reinforcement learning with adversarial value decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11308–11316, 2021. 2

[22] X. Wang, Y. Chen, and W. Zhu. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 44(09):4555–4576, sep 2022. 2, 3

[23] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *The Journal of Machine Learning Research*, 21(1):7382–7431, 2020. 2, 3

[24] Chawin Sitawarin, Supriyo Chakraborty, and David Wagner. Sat: Improving adversarial training via curriculum-based loss smoothing. In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, pages 25–36, 2021. 2

[25] Stefan Braun, Daniel Neil, and Shih-Chii Liu. A curriculum learning method for improved noise robustness in automatic speech recognition. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 548–552. IEEE, 2017. 2

[26] Junlin Wu and Yevgeniy Vorobeychik. Robust deep reinforcement learning through bootstrapped opportunistic curriculum. In *International Conference on Machine Learning*, 2022. 2

[27] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994. 2, 7

[28] Erim Kardeş, Fernando Ordóñez, and Randolph W Hall. Discounted robust stochastic games and an application to queueing control. *Operations research*, 59(2):365–382, 2011. 2

[29] Owen Lockwood and Mei Si. A review of uncertainty for deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 18, pages 155–162, 2022. 2, 8

[30] Sihong He, Songyang Han, Sanbao Su, Shuo Han, Shaofeng Zou, and Fei Miao. Robust multi-agent reinforcement learning with state uncertainty. *Transactions on Machine Learning Research*, 2023. 10

6 Notations

6.1 Markov game

The interaction among multiple agents can be modeled as **markov game** $\mathcal{G}$ [27], which can be defined as a tuple,

$$\mathcal{G} = \langle \mathcal{N}, \{\mathcal{S}^i\}_{i\in\mathcal{N}}, \{\mathcal{A}^i\}_{i\in\mathcal{N}}, \{\mathcal{R}^i_s\}_{(i,s)\in\mathcal{N}\times\mathcal{S}}, \{\mathcal{P}_s\}_{s\in\mathcal{S}}, \gamma\rangle.$$

Here $\mathcal{N} = [N]$ denotes the set of N agents, $\mathcal{S}^i$ and $\mathcal{A}^i$ denotes the state space and action space of agent $i \in \mathcal{N}$ respectively. $\mathcal{S} = \mathcal{S}^1 \times \cdots \times \mathcal{S}^N$ is the joint state space. $\mathcal{R}^i : \mathcal{S} \times \mathcal{A}^1 \times \cdots \times \mathcal{A}^N \to \mathbb{R}$ represents the reward function of agent i, which depends on the current state and the joint action of all agents. $\mathcal{P} : S \times A^1 \times \cdots \times A^N \to \Delta(S)$ represents the state transition probability that is a mapping from the current state and the joint action to the probability distribution over the state space. $\gamma \in [0, 1)$ is the discounting factor. At time t, agent i chooses its action a^i_t according to policy $\pi^i : \mathcal{S}^i \to \Delta(A^i)$. The agents' joint policy $\pi = \Pi_{i\in\mathcal{N}}\pi^i : \mathcal{S} \to \Delta(\mathcal{A})$.

6.2 Robust Nash Equilibrium in MARL with multi-modal uncertainty

In this section, we define the solution for the general robust Markov game in equation 1. Uncertainty in one model parameter influences others, while real-world scenarios may involve additional stochasticity. Ideally, we'd like to make our model robust to all four model uncertainties, but dealing with all four is complex. Most prior work focuses on single uncertainties (state [9], action [10], reward [8]). This is due to the complexity of finding Nash equilibrium and the optimal Bellman equation. Now, we define the Bellman equation for the value function, encompassing all four uncertainties: state, action, reward, and transition dynamics. We follow the maximin approach of optimization where we minimize the Bellman equation for each agent i with respect to the four uncertainty sets and maximize with respect to its policy π^i. Our aim is to select the entries $\bar{P}$, $\bar{R}^i_s$, $\bar{s}$, $\bar{a}$, from uncertainty set $\bar{\mathcal{P}}_s$, $\bar{\mathcal{R}}^i_s$, $\bar{\mathcal{O}}$, $\bar{\text{A}}$ that minimises the expected return. Thus, the optimal Bellman equation will be,

$$\bar{V}^i_*(s^i) = \max_{\pi^i(.|s^i)} \min_{\substack{\bar{P}(.|s,.)\in\bar{\mathcal{P}}_s \\ \bar{R}^i_s\in\bar{\mathcal{R}}^i_s \\ \bar{s}\in\bar{\mathcal{O}} \\ \bar{a}\in\bar{\text{A}}}} \sum_{a\in\mathcal{A}} \prod_{j=1}^{N} \pi^j(a^j|\bar{s}^j)(\bar{R}^i(s,\bar{a}) + \gamma \sum_{s'\in S} \bar{P}(s'|s,\bar{a})\bar{V}^i_*(s')),$$

where $\bar{s} = \{\bar{s}^1, \bar{s}^2, ...\bar{s}^N\}$ and $\bar{a} = \{\bar{a}^1, \bar{a}^2, ...\bar{a}^N\}$. The true state of an agent remains unaltered, with only the state perceived by other agents being perturbed. As a result, policy π^i takes perturbed state $\bar{s}^i$ as input whereas reward R and transition dynamics P function takes true state s as input. Policy $\pi^i(.|\bar{s}^i)$ generates true actions a which are then perturbed by the environment to become $\bar{a}$. The reward R and transition dynamics P function takes perturbed $\bar{a}$ as input and undergo perturbation themselves. If an optimal value function exists, then we define the existence of robust Nash equilibrium (RNE). RNE is the solution for the robust Markov game. Check appendix section 6.5 for the definition.

Theorem 1: Existence of robust Nash equilibrium → Existence of optimal value function.

For detailed proof check appx 7. Theoretically proving the existence of NE policy for the $\bar{\mathcal{G}}_{general}$ is out of the scope of this work. [9] find NE for state uncertainty (shown in appx. 8) and [8] find NE for reward/transition dynamics uncertainty (shown in appx 9).

6.3 Nash Equilibrium in MARL

NE is one of the commonly used solution concept in multi-agent static games. We will now introduce NE in MARL The expected return in case of multi agent RL for i^{th} agent is

$$V^i_\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r^i_t | s_0 = s, a^i_t \sim \pi^i(.|s_t), a^{-i}_t \sim \pi^{-i}(.|s_t)],$$

where $-i$ represents the indices of all agents except agent i, and $\pi^{-i} = \Pi_{i\neq j}\pi_j$ refers to the joint policy of all agents except agent i. In order to find the optimal robust value function for the single agent the other agent policies are considered stationary. Since all policies are evolving continuously and expected return is dependent on all agent policies, one commonly used solution for optimal policy $\pi^* = \{\pi^*_1, \pi^*_2, \ldots \pi^*_N\}$ is nash equilibrium. π_* is called Markov perfect Nash Equilibrium. Optimal value function is defined by,

$$V^i_*(s) = \max_{\pi^i(.|s)} \sum_{a\in\mathcal{A}} \prod_{j=1}^{N} \pi^j(a^j|s^j)(R^i(s,a) + \gamma \sum_{s'\in S} P(s'|s,a)V^i_*(s')). \tag{2}$$

Non-stationarity is one of the main reasons for difficulty in MARL convergence, which is further attenuated with uncertainty.

6.4 Uncertainty in RL

In this section we explain the different types of uncertainty in RL and introduce our uncertainty model for each.

Uncertainty can be categorised into two kinds, aleatoric and epistemic. Aleatoric uncertainty or statistical uncertainty originates from the statistic nature of the environment and interactions with the environment. This uncertainty can be modeled and evaluated but cannot be reduced. Whereas epistemic uncertainty or model uncertainty originates from current limitations of training the neural network and is reducible. There are 3 main potential sources of aleatoric uncertainty in RL [29] which corresponds to the three main components of the MDP, stochastic rewards, stochastic observations, and stochastic actions. However, stochastic observations can occur due to stochastic transition dynamics as well as stochastic observations itself.

Stochastic rewards would mean that for every state and action pair, we now have a distribution of the reward rather than a fixed reward. Rewards only play a role during training and not during testing. Thus developing a robust system to reward uncertainties imply that we are able to train the model to converge and achieve the goal even with high reward uncertainty. The model we have used for **stochastic reward function** is defined as:

$$\bar{R}^i(s,a) = \mathcal{N}_{trunc}(R^i(s,a), \epsilon), \tag{3}$$

where R^i is true reward for agent i, $\bar{R}^i$ is perturbed reward, ϵ is standard deviation and $\mathcal{N}_{trunc}$ is truncated normal distribution truncated at 2ϵ. Increasing ϵ will increase the uncertainty.

The **stochastic observations** can stem from stochastic transition dynamics or stochastic observations itself. If the $\mathcal{P}$ function in the MDP is non-deterministic, then the transition from one state to the next is a source of uncertainty. Thus, **stochastic transition dynamics** means that for every current state and action pair, we now have a distribution over the next state and not a fixed specific next state.

In the other scenario of stochastic observations, the true state of the system remains unchanged and only the observed state is perturbed. The input to the policy network is the perturbed state, but for the reward and transition dynamics functions input is the true state. The model we have used for **stochastic observation function** is:

$$\bar{s}^i = \mathcal{N}_{trunc}(s^i, \mu), \tag{4}$$

where s is true state, $\bar{s}$ the perturbed state, μ is standard deviation and $\mathcal{N}_{trunc}$ is truncated normal distribution truncated at 2μ. Increasing μ will increase the uncertainty.

The **stochastic actions** means that there is uncertainty in about the next state due to uncertain actions. One example is any stochastic policy algorithm (PPO, SAC) in which the action is chosen from a distribution instead of a deterministic point. We define our model for stochastic actions as:

$$\bar{a}^i = \mathcal{N}_{trunc}(a^i, \nu), \tag{5}$$

where a is true action, $\bar{a}$ the perturbed action, ν is standard deviation and $\mathcal{N}_{trunc}$ is truncated normal distribution truncated at 2ν. Increasing ν will increase the uncertainty.

Note that the range of value of observations and actions is quite small as compared to that of reward. Thus, the magnitude of robustness is different for different uncertainty parameters.

6.5 Robust Nash Equilibrium

Definition 1: (Robust Nash Equilibrium) Given a Markov game $\bar{\mathcal{G}}_{general}$ with state, reward, action and transition dynamics uncertainty, a joint policy $\pi_* = \{\pi_*^1, \pi_*^2 \dots \pi_*^N\}$ is said to be RNE for $i \in \mathcal{N}$, $s \in \mathcal{S}$, iff there exists optimal value function $V_* = \{V_*^1, V_*^2, \dots V_*^N\}$ and satisfies,

$$\pi_*^i(.|s^i) \in \underset{\pi^i(.|s^i)}{\arg\max} \min_{\substack{\bar{P}(.|s,.)\in\bar{\mathcal{P}}_s \\ \bar{R}_s^i \in \bar{\mathcal{R}}_s^i \\ \bar{s}\in\bar{\mathcal{O}} \\ \bar{a}\in\bar{\mathrm{A}}}} \sum_{a\in\mathcal{A}} \pi^i(a^i|\bar{s}^i) \prod_{i\neq j} \pi_*^j(a^j|\bar{s}^j)(\bar{R}^i(s,\bar{a}) + \gamma \sum_{s'\in S} \bar{P}(s'|s,\bar{a})\bar{V}_*^i(s')).$$

6.6 Algorithm

Algorithm 1 Reward With State/Action Uncertainty

```
Require: $\Delta\epsilon, \Delta\mu/\nu$
  $\epsilon \leftarrow 0, \mu/\nu \leftarrow 0$
  $\epsilon_{converged} \leftarrow False,$
  $\mu/\nu_{converged} \leftarrow False$
  $TrainTillSuccess(\epsilon, \mu/\nu)$
  $\mu/\nu \leftarrow SkipAhead(\mu/\nu)$
  while not ($\epsilon_{converged}$ and $\mu/\nu_{converged}$) do
    if not $\epsilon_{converged}$ then
      $\epsilon \leftarrow \epsilon + \Delta\epsilon$
      $\epsilon_{converged} \leftarrow TrainToSucc(\epsilon, \mu/\nu)$
    end if
    if not $\mu/\nu_{converged}$ then
      $\mu/\nu \leftarrow \mu/\nu + \Delta\mu/\nu$
      $\mu/\nu_{converged} \leftarrow TrainToSucc(\epsilon, \mu/\nu)$
      $\mu/\nu \leftarrow SkipAhead(\mu/\nu)$
    end if
  end while
```

7 Proof for Theorem 1: Existence of robust Nash Equilibrium → Existence of optimal Value Function

This proof has been conducted for reward and transition dynamics uncertainty [8] but not for partially observable games. It has also been explored for state uncertainty [9]. We focus on developing the general proof when all possible uncertainties are present in MARL.

Lets define the non-linear operator on $\mathcal{L}$ such that,

$$\mathcal{L}^i v^i(s) = \max_{\pi^i(.|s^i)} \min_{\rho} \left[\sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a}) + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a}) v^i(s') \right]$$

, where $\rho = \{\bar{P}, \bar{R}, \bar{s}, \bar{a}\}$

We can think of ρ as adversarial strategy that is playing against the good policy π by selecting the values $\{\bar{P}, \bar{R}, \bar{s}, \bar{a}\}$ from their respective uncertainty sets such that it minimises the expected return.

Let u and v be two value functions in $\mathbb{V}$. Let $\{\pi^u_*, \rho^u_*\}$ and $\{\pi^v_*, \rho^v_*\}$ be two different Nash Equilibrium with respect to $\bar{\mathcal{G}}_{general}$.

$$\mathcal{L}^i v^i(s) = \sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a})_{(\pi^v_*, \rho^v_*)} + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^v_*, \rho^v_*)} v^i(s')$$

, where $\rho^v_* = \{\bar{P}, \bar{R}, \bar{s}, \bar{a}\}$ is the optimal value that minimises the value function equation.

$$\mathcal{L}^i u^i(s) = \sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a})_{(\pi^u_*, \rho^u_*)} + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^u_*, \rho^u_*)} u^i(s')$$

, where $\rho^u_* = \{\bar{P}, \bar{R}, \bar{s}, \bar{a}\}$ is the optimal value that minimises the value function equation.

Its intuitive that optimal π_* maximizes the above equation, whereas optimal ρ_* minimises the above equation. Therefore we can write the following equation,

$$\sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a})_{(\pi^u_*, \rho^v_*)} + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^u_*, \rho^v_*)} v^i(s') \leq \mathcal{L}^i v^i(s) \leq \sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a})_{(\pi^v_*, \rho^u_*)} + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^v_*, \rho^u_*)} v^i(s')$$

$$\sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a})_{(\pi^v_*, \rho^u_*)} + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^v_*, \rho^u_*)} u^i(s') \leq \mathcal{L}^i u^i(s) \leq \sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a})_{(\pi^u_*, \rho^v_*)} + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^u_*, \rho^v_*)} u^i(s')$$

(a) Now lets assume, $\mathcal{L}^i v^i(s) \leq \mathcal{L}^i u^i(s)$

$$\begin{aligned}
0 &\leq \mathcal{L}^i u^i(s) - \mathcal{L}^i v^i(s) \\
&\leq \left[\sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a})_{(\pi^u_*, \rho^v_*)} + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^u_*, \rho^v_*)} u^i(s')\right] - \left[\sum_{a \in \mathcal{A}} \bar{R}^i(s, \bar{a})_{(\pi^u_*, \rho^v_*)} + \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^u_*, \rho^v_*)} v^i(s')\right] \\
&\leq \gamma \sum_{s' \in S} \bar{P}(s'|s, \bar{a})_{(\pi^u_*, \rho^v_*)} (u^i(s') - v^i(s')) \\
&\leq \gamma ||u^i(s') - v^i(s')||
\end{aligned}$$

(b) Assuming , $\mathcal{L}^i u^i(s) \leq \mathcal{L}^i v^i(s)$ and following the same argument as before we get,

$$\mathcal{L}^i v^i(s) - \mathcal{L}^i u^i(s) \leq \gamma ||v^i(s') - u^i(s')||$$

Thus, combining (a) and (b), we get,

$$||\mathcal{L}^i v^i(s) - \mathcal{L}^i u^i(s)|| \leq \gamma ||v^i(s') - u^i(s')||$$

Thus, $\mathcal{L}^i$ is a contraction mapping on V

Now since $||v|| = \sup_i ||v^i||$, we can write the following -

$$||\mathcal{L}v - \mathcal{L}u|| = \sup_i ||\mathcal{L}^i v^i - \mathcal{L}^i u^i|| \leq \gamma \sup_i ||v^i - u^i|| = \gamma ||v - u||$$

Thus, $\mathcal{L}$ is a contraction mapping on $\mathbb{V}$

8 Nash Equilibrium for state uncertainty in MARL

A nice proof for the conditional existence of Nash equilibrium is done in [30] for the case of state uncertainty. They define the following robust Markov game,

$$\mathcal{G} = \{\mathcal{N}, \mathcal{M}, \{\mathcal{S}^i\}_{i \in \mathcal{N}}, \{\mathcal{A}^i\}_{i \in \mathcal{N}}, \{\mathcal{B}^i\}_{i \in \mathcal{N}}, \{r^i\}_{i \in \mathcal{N}}, p, \gamma\}$$

$\mathcal{N} = \{1, 2, ..., N\}$ is the set of N agents and $\mathcal{M} = \{\bar{1}, \bar{2}, ..., \bar{N}\}$ is the corresponding set of N adversaries. $\gamma \in [0, 1)$ is the discount factor. $S = S_1 \times S_2 ... \times S_N$ is the joint state space. $A = A_1 \times A_2 ... \times A_N$ is the joint action space. $p : S \times A \rightarrow \Delta(S)$ are the state transition probabilities. r^i is the reward function for each agent. Every agent i is associated with an adversary $\bar{i}$. The adversary perturbs the true state of each agent $s^i \in S^i$ by producing an action $b^i \in B^i$. The perturbed state $\bar{s}^i = f(s^i, b^i)$ where f is a unique bijection given the state s^i.

The Markov game $\mathcal{G}$ is shown to be equivalent to a zero-sum two-person extensive-form game with finite strategies and perfect recall in [30].

8.1 Extensive-form game

An extensive-form game (EFG) is a tree-based representation of a game. An EFG has one root node which indicates the start of the game. Each node branches out into multiple children nodes and each branch represents one possible action. The leaf nodes indicate the end of the game and contain the pay-off/reward for the actions specified by the path from the root node to the leaf node.

The robust optimization equation can be decomposed into a two-player EFG. The first player is the nature/combined adversary who selects the perturbed state and the next player is the combined agent which chooses the best action according to the policy to be learned. The nature player has $|\bar{\mathcal{S}}|$ possible choices for the action and the agent player has $|\mathcal{A}|$ choices where $A = A^1 \times A^2 \times ... \times A^n$ i.e. the space of all possible actions for all agents. The reward for the nature player is the negative of the reward obtained by the action taken by the combined agent.

The Bellman equation for the above game $\mathcal{G}$ is written as below:

$$v^i(s) = \max_{\pi^i} \min_{\rho^i} \mathbb{E}\left[\sum_{s^{'} \in S} p(s^{'}|s, a, b)[r^i(s, a, b) + \gamma v^i(s)]|a \sim \pi(\cdot|\bar{s}), b \sim \rho(\cdot|s)\right]$$

In order for the NE (and the optimal solution to the above equation) to exist, below conditions need to be met:

- $\mathcal{S}^i$, $\mathcal{A}^i$ and $\mathcal{B}^i$ must be finite sets $\forall i \in \mathcal{N}$.
- $|r^i(s, a, b)| < M_i < M < \infty \; \forall \; i \in N, a \in A, b \in B$ and $s \in S$
- Stationary reward and transition probabilities
- f is a bijection for a given s^i
- All agents have the same reward function.

9 NE for reward and transition dynamics uncertainty in MARL

In this section, we show how uncertainty in reward and transition dynamics is handled in a multi-agent setting. We follow [16] and use the following definition of robust Markov game.

$$\bar{\mathcal{G}} = \langle \mathcal{N}, \mathcal{S}, \{\mathcal{A}^i\}_{i \in \mathcal{N}}, \{\bar{\mathcal{R}}^i_s\}_{(i,s) \in \mathcal{N} \times \mathcal{S}}, \{\bar{\mathcal{P}}_s\}_{s \in \mathcal{S}}, \gamma \rangle$$

Note: In this proof following [16] s_t denotes the system state and not the individual agent state. The expected return in case of multi-agent RL with no uncertainty for i^{th} agent is -

$$V^i_\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r^i_t | s_0 = s, a^i_t \sim \pi^i(.|s_t), a^{-i}_t \sim \pi^{-i}(.|s_t)]$$

where $-i$ represents the indices of all agents except agent i, and $\pi^{-i} = \Pi_{i \neq j} \pi_j$ refers to the joint policy of all agents except agent i. In order to find the optimal robust value function for the single agent the other agent policies are considered stationary. Since all policies are evolving continuously and expected return is dependent on all agent policies, one commonly used solution for optimal policy $\pi^* = \{\pi^*_1, \pi^*_2, \ldots \pi^*_N\}$ is Nash equilibrium. Non-stationarity is also one of the main reasons for difficulty in MARL convergence as compared to single agent RL which also reflects when uncertainty is added.

We now introduce uncertainty in rewards and transition dynamics. Thus, the desired policy should now not only be able to play against other agents' policies but also robust to the possible uncertainty of the MARL model. Each player considers a distribution-free Markov game to be played using robust optimization. To find the optimal value function we focus on the following idea from [16]. If the player knows how to play in the robust Markov game optimally starting from the next stage on, then it would play to maximize not only the worst-case (minimal) expected immediate reward, due to the model uncertainty set at the current stage, but also the worst-case expected reward incurred in the future stages. Formally, such a recursion property leads to the following Bellman-type equation:

$$\bar{V}^i_*(s) = \max_{\pi^i(.|s)} \min_{\substack{\bar{P}(.|s,.) \in \bar{\mathcal{P}}_s \\ \bar{R}^i_s \in \bar{\mathcal{R}}^i_s}} \sum_{a \in A} \prod_{j=1}^{N} \pi^j(a^j|s)(\bar{R}^i(s, a) + \gamma \sum_{s' \in S} \bar{P}(s'|s, a) \bar{V}^i_*(s'))$$

The corresponding joint policy $\pi^* = \{\pi^1, \pi^2 \ldots \pi^N\}$ is robust Markov perfect Nash equilibrium.

10 Environment Details and Training Parameter Details

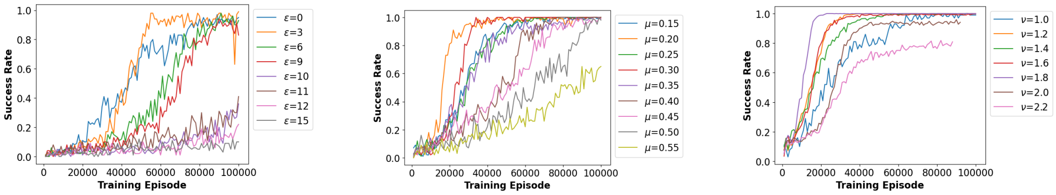
Cooperative navigation (CN): This is a cooperative game. There are 3 agents and 3 landmarks. Agents are rewarded based on how far any agent is from each landmark. Agents are penalized if they collide with other agents. So, agents have to learn to cover all the landmarks while avoiding collisions.

Keep away (KA): This is a competitive task. There is 1 agent, 1 adversary, and 1 landmark. The agent knows the position of the target landmark and wants to reach it. The adversary is rewarded if it is close to the landmark and if the agent is far from the landmark. The adversary should learn to push the agent away from the landmark.

Physical deception (PD): This is a mixed cooperative and competitive task. There are 2 collaborative agents, 2 landmarks, and 1 adversary. Both the collaborative agents and the adversary want to reach the target, but only collaborative agents know the correct target. The collaborative agents should learn a policy to cover all landmarks so that the adversary does not know which one is the true target.

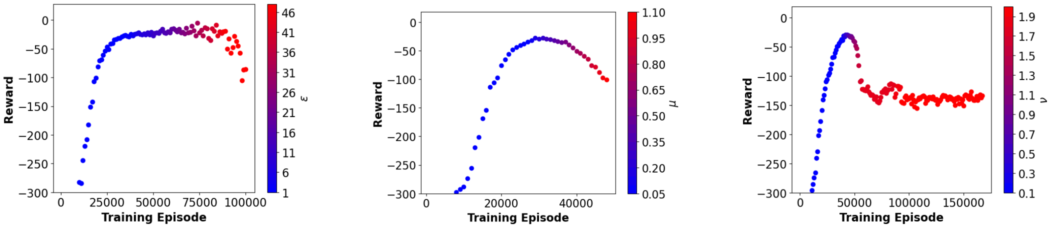
11 Reward Plots for Experiments

11.1 Cooperative Navigation Environment



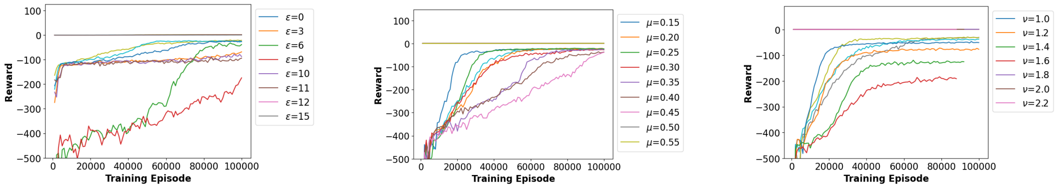
(a) Reward uncertainty. **(b)** State uncertainty. **(c)** Action uncertainty.

Figure 3: Cooperative Navigation: Baseline Performance. Success rate vs training time. An episode is successful if all landmarks are occupied by all agents. Reward uncertainty shows good performance until ϵ=9 (left), state uncertainty until μ=0.5 (middle) and action uncertainty until ν=2.0 (right).



(a) Reward Uncertainty. **(b)** State Uncertainty. **(c)** Action Uncertainty.

Figure 4: Cooperative Navigation: CL Method Performance. This plot shows the changing reward as the noise value is incremented in the CL method for the three uncertain parameters separately. Reward uncertainty learns until ϵ=47 (left), state uncertainty until μ=1.1 (middle), and action uncertainty learns until ν=2.2 (right).

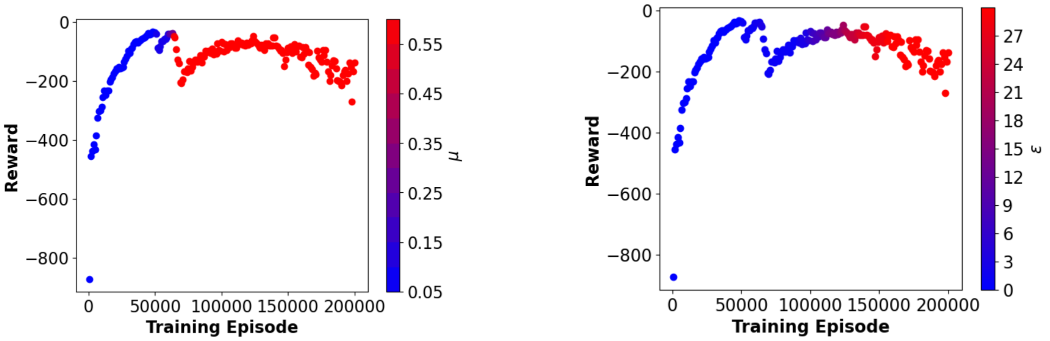


(a) Reward plot for baseline for various ϵ in reward uncertainty. Baseline was able to learn only till $\epsilon = 9$.

(b) Reward plot for baseline for various μ in state uncertainty. The baseline was able to learn only till $\mu = 0.5$.

(c) Reward plot for baseline for various ν in action uncertainty. The baseline was able to learn only till $\mu = 2.0$.

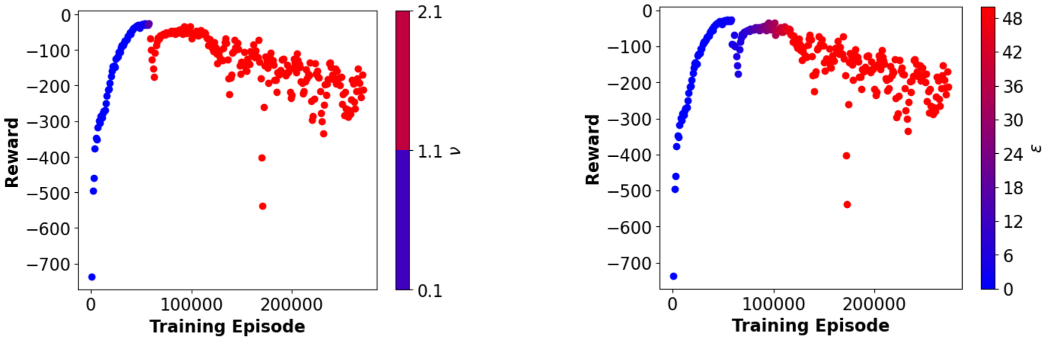
Figure 5: Cooperative Navigation: Baseline Performance. Rewards vs training time. Reward uncertainty shows good performance until ϵ=9 (left), state uncertainty shows good performance until μ=0.5 (middle) and action uncertainty shows good performance until ν=2.0 (right).



(a) Reward changes with μ

(b) Reward changes with ϵ

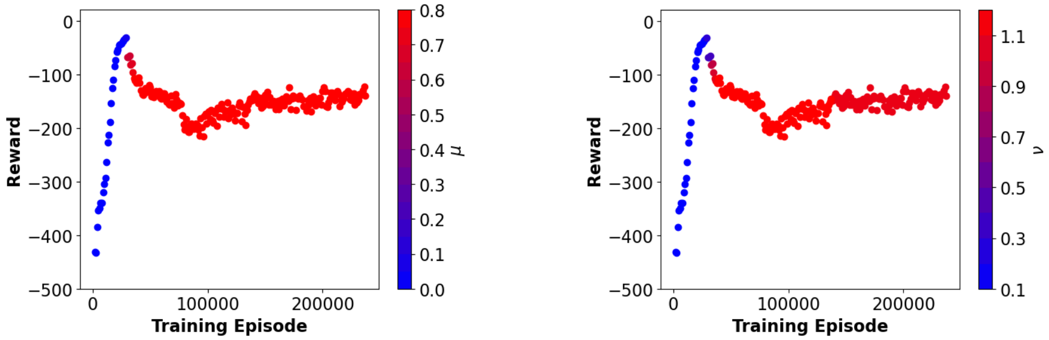
Figure 6: Reward plot for lookahead CL for the case of multiple uncertainties (reward and state) showing the changing reward for various μ (left) and ϵ (right).



(a) Reward changes with ν

(b) Reward changes with ϵ

Figure 7: Reward plot for lookahead CL for the case of multiple uncertainties (reward and action) showing the changing reward for various ν (left) and ϵ (right).

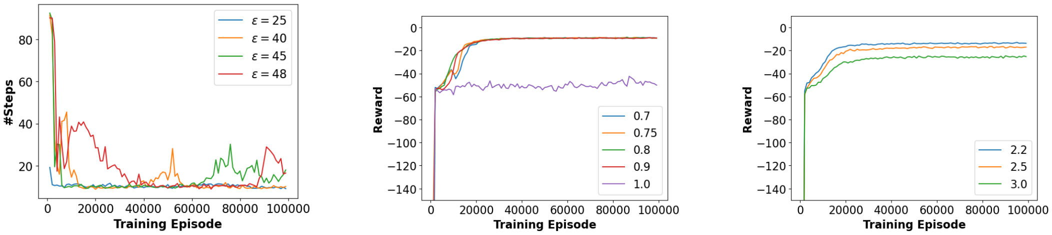


(a) Reward changes with μ

(b) Reward changes with ν

Figure 8: Reward plot for lookahead CL for the case of multiple uncertainties (state and action) showing the changing reward for various μ (left) and ν (right).

11.2 Keep Away Environment

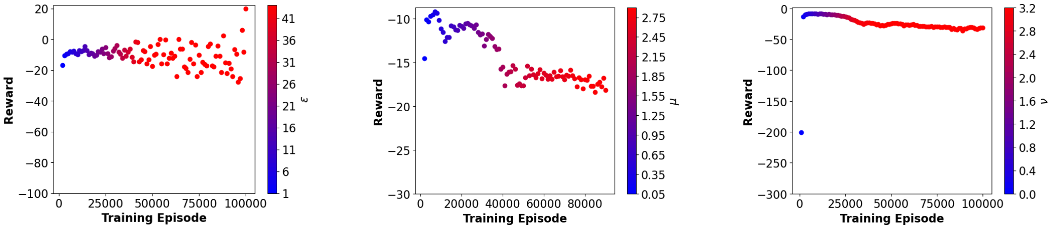


(a) Time taken by agent to reach the goal in reward uncertainty.

(b) Reward changes for various μ in state uncertainty.

(c) Reward changes for various ν in action uncertainty.

Figure 9: Keep Away: Baseline Performance. For reward uncertainty we show the plot between number of steps taken by an agent to reach the goal vs training time. This is because due to reward uncertainty reward is noisy and hence a plot of noisy reward will not give good conclusions. We observe that this number saturates for $\epsilon = 40$ but for number higher that this, its heavily fluctuating hence concluding that reward uncertainty learns until $\epsilon = 40$. For state and action uncertainty we show reward vs training time. State uncertainty shows good performance until μ=0.9 (middle) and action uncertainty shows good performance until ν=2.0 (last).



(a) Reward Uncertainty.

(b) State Uncertainty.

(c) Action Uncertainty.

Figure 10: Keep Away: CL Method Performance. This plot shows the changing reward as the noise value is incremented in the CL method for the three uncertain parameters separately. Reward uncertainty learns until ϵ=43 (left), state uncertainty until μ=2.5 (middle), and action uncertainty learns until ν=3.1 (last).