ROBUSTNESS TO MULTI-MODAL ENVIRONMENT UNCERTAINTY IN MARL USING CURRICULUM LEARNING

Anonymous authors
Paper under double-blind review

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 one environment variable (i.e. action, state or reward). This is because a multi-agent system itself is highly complex and non-stationary. However, in a real-world scenario 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 two distinct environmental uncertainty simultaneously and present extensive results across both cooperative and competitive MARL environments, demonstrating that our approach achieves state-of-the-art levels of robustness.

1 INTRODUCTION

Multi-Agent Reinforcement Learning (MARL) has shown tremendous success in solving many complex real-world decision making problems such as in robotics(path-planning (Wang et al., 2019), task-allocation (Agrawal et al., 2023; 2022)), traffic management (Kolat et al., 2023); Game Theory and Economics (Nowé et al., 2012) etc. In MARL (Zhang et al., 2021), the agent interacts with the environment as well the other agents to maximize its own long-term return in a shared environment. Thus making it more complex than single agent RL with finding Nash Equilibrium as one of most popular solution concept (Busoniu et al., 2008).

Real-world MARL applications require training the agent in simulations and transferring it to the real-world. In this case, its possible that the agent does not have accurate knowledge or there is shift in the environment parameters (eg. reward function/transition dynamics function is not exactly same as in the simulations) or there is noise (eg. the information about state, action, reward are not transferred with perfect accuracy) or issues in the hardware, etc. Thus, leading to environment uncertainty. Though there has been work on robustness to uncertainty in reward, transition dynamics (Zhang et al., 2020), state (He et al., 2023) and action (Li et al., 2019a), uncertainty in multiple parameters has not been studied simultaneously. In a real world setting, uncertainty will be present in multiple environmental parameters necessitating the need to develop methods that are robust to multi-modal uncertainty.

Contributions. In this work, **(1)** we focus on developing robustness to two uncertain parameter at a time, thus, making this the first work to handle multi-modal uncertainty in MARL. **(2)** We also define and theoretically formalise the general problem of finding optimal policy in MARL with multi-modal uncertainty. **(3)** Generally, finding optimal policy in MARL is characterized by finding Nash Equilibrium for the markov game however finding NE for this generalised problem is highly complex. In this work, we formally investigate and design an efficient curriculum learning (CL) to solve this problem. **(4)** We also show experimentally that our method is able to find optimal solution for the given robust Markov games and generate state-of-the-art robustness for reward, state and action uncertainty. **(5)** As a by product, this is the also the first work to handle action uncertainty in MARL.

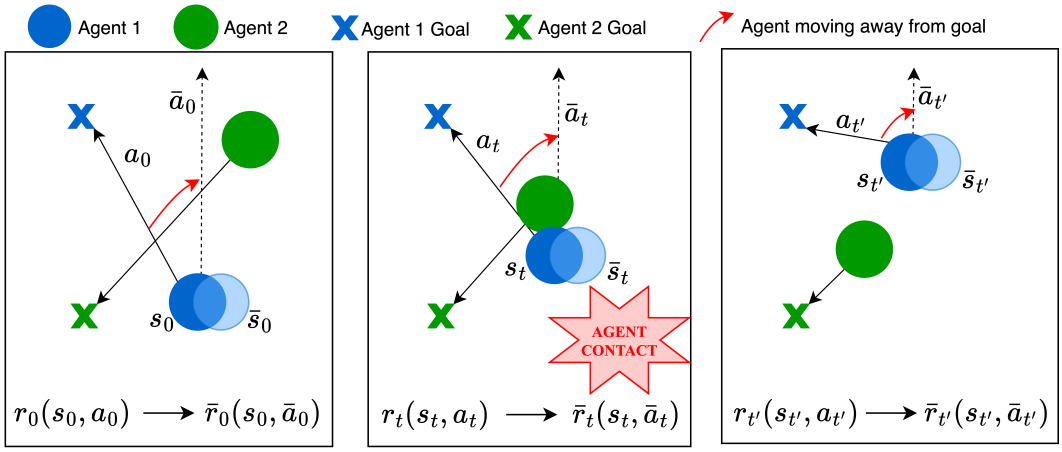


Figure 1: In this figure we try to motivate the problem of solving multi-modal uncertainty for MARL. We show here some of the problems that can occur due to state, action and reward uncertainty. For example, at time = 0 (first image) agent 2 observes the perturbed state $\bar{s}_0$ of agent 1 and not the true state s_0, leading to a collision at the time = t (middle image). Agent 1 needs to reach the goal but its perturbed actions take it away from the goal (last image). Reward uncertainty does not give accurate expected return for an action in a given state, thus making the training more difficult.

2 Related Work

Robustness in RL. Robustness in reinforcement learning is due to adversarial attacks (Liang et al., 2022) or uncertainty in model/environment parameters. In single agent RL, robustness to uncertainty is handled by maximin optimisation between the agent and the uncertainty set in the form of zero sum game (Tessler et al., 2019; Wiesemann et al., 2013; Xie et al., 2022; Nilim & El Ghaoui, 2005). In MARL, uncertainty is defined in the form of robust Markov game. The agents individually maximise their return while interacting with each other in presence of uncertainty. MARL research has made tremendous advancements in the recent times, however there is only very few work handling uncertainty. Kardeş et al. (2011) is the only work on multi-modal uncertainty (reward and transition dynamics) but it only handles it theoretically. Some of the robustness to single uncertainty works handle reward, transition dynamics (Zhang et al., 2020) or state (He et al., 2023) uncertainty. Li et al. (2019b); Sun et al. (2022) develop robustness to the adversarial agent's actions by training the agent to handle the worst case action of the adversarial agents in a competitive setting. Robustness to action perturbations in cooperative environment with adversaries had been studied in Nisioti et al. (2021); Phan et al. (2020; 2021).

Curriculum learning Wang et al. (2022); Narvekar et al. (2020) has been widely used for improving robustness in various different domains, such as object classification (Sitawarin et al., 2021), automatic speech recognition (Braun et al., 2017), etc. **CL for robustness in RL.** Wu & Vorobeychik (2022) is the closest work to ours on using CL to improve robustness in RL. They implement bootstrapped opportunistic curriculum learning to improve robustness in single agent RL with RADIAL-DQN as baseline. Mysore (2019) develops robustness by developing generalisation to multiple tasks. They formulate the training curriculum as a multi-armed bandit problem which selects the task to train the RL model to give maximum reward gain.

There is no existing work which uses CL for multi-modal uncertainty. It is also first time it is being used to handle robustness in MARL.

3 Problem Formulation

This section gives the problem formulation with multiple uncertainties (reward, state, action and transition dynamics) and also introduce the different uncertainties in RL.

3.1 ROBUST MARKOV GAME

The interaction among multiple agents can be modeled as **markov game** $\mathcal{G}$ (Littman, 1994), 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. $\mathcal{R}^i \colon \mathcal{S} \times \mathcal{A}^1 \times \cdots \times \mathcal{A}^N \to \mathbb{R}$ 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 $\pi = \Pi_{i\in\mathcal{N}}\pi^i : \mathcal{S} \to \Delta(\mathcal{A})$.

In real-world settings, a Markov game will not have accurate information available and thus contain uncertainties in model such as reward, state, action and/or transition probability model. Thus we define **Robust markov game** (Kardeş et al., 2011) as,

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

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.

Note that the state perturbation reflects the state uncertainty from the perspective of each agent, but it does not change the true state of multi-agent systems. The state still transits from the true state to the next state. 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 $\bar{s}$.

3.2 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 (Lockwood & Si, 2022) 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:

$$\bar{R}^i(s, a) \sim \mathcal{N}_{trunc}(R^i(s, a), \epsilon), \quad (2)$$

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:

$$\bar{s}^i \sim \mathcal{N}_{trunc}(s^i, \mu), \tag{3}$$

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:

$$\bar{a}^i \sim \mathcal{N}_{trunc}(a^i, \nu), \tag{4}$$

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.

4 Robust Nash Equilibrium with multimodal uncertainty

4.1 Nash Equilibrium in MARL

NE is one of the commonly used solution concepts in multi-agent stochastic 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 $\pi^* = \{\pi^*_1, \pi^*_2, \dots \pi^*_N\}$ is nash equilibrium. π^* is called Markov perfect Nash Equilibrium. Optimal value function is defined by,

$$V^i_*(s) = \max_{\pi^i(.|s)} \sum_{a \in \mathcal{A}} \prod_{j=1}^{N} \pi^j(a^j|s^j)(R^i(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^i_*(s')). \tag{5}$$

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

4.2 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 which includes the 4 kinds of aleatoric uncertainty in a MARL system. Uncertainty in one model parameter directly or indirectly affects the others, but there could be additional stochasticity in other parameters in the real world. Ideally, we would like to make our model robust to all four aleotoric uncertainties, but handling uncertainty in all 4 variables is a complex problem. Therefore, most existing work focuses only on single uncertainty (state **?**, actionLi et al. (2019a), reward Zhang et al. (2020)). This is mostly due to the complexity of finding a Nash equilibrium and the solution to the optimal Bellman equation.

We now define the Bellman equation for the value function including all four uncertainties discussed in the previous section - state, action, reward, and transition dynamics. We follow the maximin approach of optimization where we minimize the bellman equation 5 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 $\bar{\mathcal{P}}_s$, $\bar{\mathcal{R}}^i_s$, $\bar{\mathcal{O}}$, $\bar{\mathrm{A}}$ that minimises the expected return. Thus, the optimal Bellman equation will be,

$$\bar{V}^i_*(s^i) = \max_{\pi^i(.|s^i)} \min_{\substack{\bar{P}(.|s,.)\in\bar{\mathcal{P}}_s \\ \bar{R}^i_s\in\bar{\mathcal{R}}^i_s \\ \bar{s}\in\bar{\mathcal{O}} \\ \bar{a}\in\bar{\mathrm{A}}}} \sum_{a\in\mathcal{A}} \prod_{j=1}^{N} \pi^j(a^j|\bar{s}^j)(\bar{R}^i(s,\bar{a}) + \gamma \sum_{s'\in S} \bar{P}(s'|s,\bar{a})\bar{V}^i_*(s')) \tag{6}$$

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 unchanged but only the state perceived by other agents is perturbed, therefore only 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. The policy $\pi^i(.|\bar{s}^i)$ then produces the true actions a which are then perturbed by the environment into $\bar{a}$. The reward R and transition dynamics P function takes perturbed $\bar{a}$ as input and eventually gets perturbed itself.

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.

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 robust Nash equilibrium 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,

$$\pi^*_i(.|s^i) \in \arg\max_{\pi_i(.|s^i)} \min_{\substack{\bar{P}(.|s,.)\in\bar{\mathcal{P}}_s \\ \bar{R}^i_s\in\bar{\mathcal{R}}^i_s \\ \bar{s}\in\bar{\mathcal{O}} \\ \bar{a}\in\bar{\mathrm{A}}}} \sum_{a\in\mathcal{A}} \pi_i(a^i|\bar{s}^i) \prod_{i\neq j} \pi^*_j(a^j|\bar{s}^j)(\bar{R}^i(s,\bar{a}) + \gamma \sum_{s'\in S} \bar{P}(s'|s,\bar{a})\bar{V}^i_*(s')).$$

Theorem 1: Existence of robust Nash Equilibrium $\rightarrow$ Existence of optimal Value Function.

For detailed proof check appendix 12. This proof has been conducted for reward and transition dynamics uncertainty Zhang et al. (2020) but not for partially observable games. It has also been explored for state uncertainty **?**. We focus on developing the general proof when all possible uncertainties are present in MARL.

Theoretically proving the existence of NE policy for the $\bar{\mathcal{G}}_{general}$ is out of the scope of this work. **?** find NE for state uncertainty, also shown in the appendix. 10 and Zhang et al. (2020) find NE for reward/transition dynamics uncertainty, also shown in appendix 11.

5 CL BASED ROBUSTNESS TO MULTI-MODAL UNCERTAINTY

Curriculum learning (Wang et al., 2022; Narvekar et al., 2020) is a methodology to optimize the order in which experience is accumulated by the agent, so as to increase performance or training speed on a set of final tasks. An important challenge while designing the curriculum for training is developing a strategy to measure the difficulty level of our task. In our case, we use the noise parameter (ϵ, μ and ν) to increase the task difficulty level. Through generalization, knowledge acquired by training in simple tasks can be leveraged to reduce the exploration of more complex tasks. Therefore by training for lower levels of noise parameter, the model learns faster for higher levels of noise. Thus making our method sample efficient.

Our base model for applying CL is Zhang et al. (2020) and we introduce state, reward, and action uncertainty in it as shown in equations 3, 2, 4 respectively.

5.1 EFFICIENT LOOKAHEAD CL

This section presents our efficient curriculum learning algorithm for the case when only one uncertain parameter (state, reward, or action) is present. There are a total of four possible uncertain parameters, however, most work considers uncertainty in transition dynamics and state to be similar and does not

study it separately. We do not have transition dynamics uncertainty because our transition dynamics are deterministic. The algorithm is described in 1. The functions used in the algorithm are described below.

TrainTillSuccess(noise_parameters) - For the given noise parameters, train the model till it converges (success rate $>= 0.95$) or it reaches a pre-defined a maximum number of episodes (100,000 episodes) without converging. **SkipAhead**(noise_parameter) - Keep increasing the given noise parameter by pre-defined delta values (0.05 for state/action uncertainty, 1.0 for reward uncertainty) and evaluate the success_rate of the model. If success_rate $>= 0.95$, continue to the next (higher) value of noise_parameter without training the model. Stop when we reach a value of noise where the success_rate < 0.9, skip back to the previous value where success_rate $>= 0.95$.

Algorithm 1 Lookahead CL for single uncertainty

Require: $\Delta\lambda$ where $\lambda \in \{\epsilon, \mu, \nu\}$
  $\lambda \leftarrow 0$
  $\lambda_{converged} \leftarrow False$
  $TrainToSuccess(\lambda)$
  while not $\lambda_{converged}$ **do**
    $\lambda \leftarrow \lambda + \Delta\lambda$
    $\lambda_{converged} \leftarrow TrainToSuccess(\lambda)$
  end while

5.2 Efficient Lookahead CL for multiple uncertainties

The algorithm for efficient curriculum learning for reward with state/action perturbations is described in 2. Our aim is to simultaneously increase both reward and state/action uncertainty in an efficient manner in each iteration of curriculum learning, to eventually train a model that can handle two uncertainties at the same time. Note there is no skip ahead in the case of the reward uncertainty parameter since we do not have reward uncertainty during the evaluation phase. In a similar way algorithm for action and reward perturbations is shown in algorithm 3.

Algorithm 2 Reward+State/Action Uncertainty

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

Algorithm 3 State+Action Uncertainty

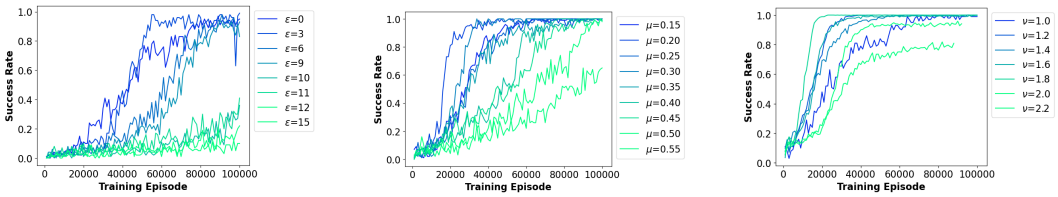
Require: $\Delta\nu, \Delta\mu$
  $\nu \leftarrow 0, \mu \leftarrow 0$
  $\nu_{converge} \leftarrow False,$
  $\mu_{converge} \leftarrow False$
  $TrainToSuccess(\nu, \mu)$
  $\mu \leftarrow SkipAhead(\mu)$
  while not ($\nu_{converge}$ **and** $\mu_{converge}$) **do**
    if not $\nu_{converge}$ **then**
      $\nu \leftarrow \nu + \Delta\nu$
      $\nu_{converge} \leftarrow TrainToSucc(\nu, \mu)$
      $\nu \leftarrow SkipAhead(\nu)$
    end if
    if not $\mu_{converge}$ **then**
      $\mu \leftarrow \mu + \Delta\mu$
      $\mu_{converge} \leftarrow TrainToSucc(\nu, \mu)$
      $\mu \leftarrow SkipAhead(\mu)$
    end if
  end while

6 Experiments

In this section, we show results for our curriculum learning-based method on three multi-particle environments and compare the results with state-of-the-art robustness in those environments. The three environments are cooperative navigation, keep-away, and physical deception. For each of these

environments, we first compare results for the base method (without CL) with our CL method for varying levels of either reward, state, or action uncertainty. Then we show results for the combination of two uncertainties: state + reward, action + reward, and state + action. This paper is the first to handle multi-modal uncertainty in MARL. We note values related to reward and state uncertainty by evaluating the trained model 1000 times and reporting its mean and standard deviation. However, for reward uncertainty, we show training plots since rewards do not play a role in evaluation and therefore uncertainty during evaluation does not make sense. We show

6.1 Robustness under uncertainty in a single parameter.

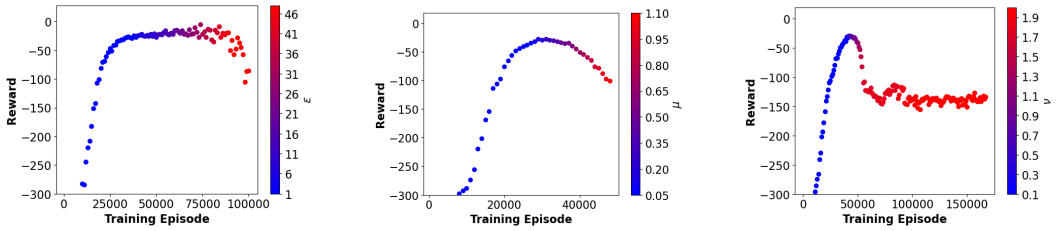


(a) Success rate for various ϵ in reward uncertainty.

(b) Success rate for various μ in state uncertainty.

(c) Success rate for various ν in action uncertainty.

Figure 2: 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 shows good performance until μ=0.5 (middle) and action uncertainty shows good performance until ν=2.0 (right).



(a) Reward Uncertainty.

(b) State Uncertainty.

(c) Action Uncertainty.

Figure 3: 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).

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	$\mathbf{1 \pm 0.1}$	0.6 ± 0.5	-	-	-	-
CL	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0.1	0.98 ± 0.2	$\mathbf{0.94 \pm 0.2}$	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	$\mathbf{1 \pm 0.03}$	0.9 ± 0.04

Table 1: Cooperative Navigation. The table shows a detailed comparison between CL and baseline for reward uncertainty (top), state uncertainty (middle), and action uncertainty (bottom) using success rates. For reward uncertainty, the CL-based method learned to converge until $\epsilon = 48$, whereas baseline learned until $\epsilon = 9$. For state uncertainty, the CL-based method gave a high success rate until $\mu = 1.1$, whereas baseline learned until $\mu = 0.5$. For action uncertainty, the CL-based method gave a high success rate until $\nu = 2.4$, whereas baseline learned until $\nu = 2.0$.

Cooperative Navigation Environment. This is a cooperative environment where three agents learn to occupy all three landmarks while avoiding collisions. For **reward uncertainty** we see from the success rate figure 2a that the model can learn until $\epsilon = 9$ which is the current state-of-the-art robustness (Zhang et al., 2020). However, with our lookahead-CL method, we were able to achieve robustness up to $\epsilon = 47$ (check reward plot 3a). **For state uncertainty** as we see from the figure 2b the model was able to learn until $\mu = 0.5$ while with CL it reached $\mu = 1.1$ (check reward plot 3b). **For action uncertainty** we see from the figure 2c the model was able to learn until $\nu = 2.0$ while with CL it reached $\nu = 2.2$ (check reward plot 3c). This is the first work on action uncertainty so we do not have a baseline to compare with. For state uncertainty, **?** does not have a comparison for various uncertainty values. Table 1 shows detailed comparison results between CL and baseline, and eventually concludes that CL achieves state-of-the-art robustness.

6.2 MULTI-MODAL CL COMBINING TWO UNCERTAINTY:

Cooperative Navigation Environment: We try the combinations, reward + state, reward + action, and action + state uncertainty to test how CL works when two uncertainties are combined. For **reward+state** uncertainty combination as shown in table 2 we are able to learn until $\mu = 0.7$ when $\epsilon = 0$ and $\epsilon = 29$ while training Check figure 7 for reward plots. For **reward+action** uncertainty combination as shown in table 2 we are able to learn uptil $\nu = 2.4$ when $\epsilon = 0$ and $\epsilon = 50$ while training. Check figure 8 for reward plots. For **action+state** uncertainty combination as shown in table 2 we are able to learn uptil $\nu = 3$ when $\mu = 0$ and $\mu = 1$ when $\nu = 0$. Check figure 9 for reward plots. Thus, we can see that even after introducing two uncertainties at the same time, the method outperforms the baseline.

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	$\mathbf{0.95 \pm 0.2}$	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	$\mathbf{0.96 \pm 0.2}$
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	$\mathbf{1 \pm 0.05}$	$\mathbf{1 \pm 0.06}$	$\mathbf{0.96 \pm 0.18}$
0.6	1 ± 0.6	0.98 ± 0.1	0.98 ± 0.14	$\mathbf{0.96 \pm 0.2}$	$\mathbf{0.92 \pm 0.28}$	0.8 ± 0.4	0.73 ± 0.4	0.6 ± 0.5
0.8	0.98 ± 0.14	$\mathbf{0.95 \pm 0.2}$	$\mathbf{0.9 \pm 0.3}$	0.85 ± 0.35	0.8 ± 0.45	0.6 ± 0.5	0.5 ± 0.5	0.4 ± 0.5
1	$\mathbf{0.93 \pm 0.25}$	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

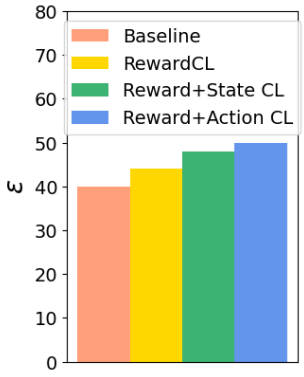
Table 2: Cooperative Navigation. This table shows the success rate performance of the CL-based method when both reward and state uncertainty (top), reward and action uncertainty (middle), or state and action uncertainty (bottom) are present. An episode is successful if all landmarks are occupied by all agents.

Keep Away environment: This is a competitive environment with one agent, one adversary, and two landmarks. One landmark is randomly chosen to be the goal location and the objective of the agent is to reach the goal and that of the adversary is to block the agent from reaching the goal. The agent knows which landmark is the goal while the adversary does not. For all experiments in this environment, noise is added only to the agent's state, reward, and action. Evaluation is also done similarly. An agent is considered successful if it reaches the goal within 100 steps.

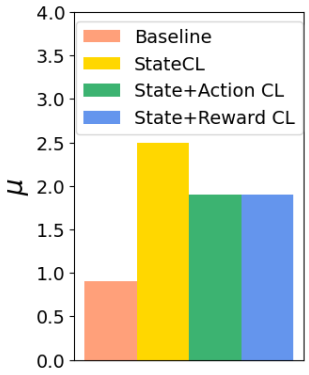
In Fig. 4, we show bar plots indicating the maximum noise level at which the model converged for different training settings. We define that the model has converged if the success rate of the agent is greater than 90%. In each of the sub-figures, the parameter mentioned on the Y-axis is progressively increased until failure while the other parameters are set to zero. Check figure 10 and 11 for reward graphs.

Physical Deception environment: 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.

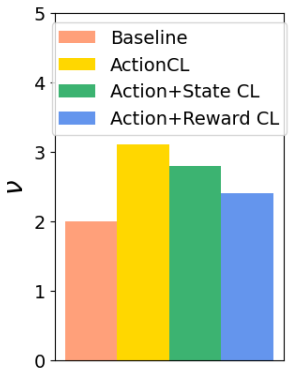
In Fig. 5, we again show barplots indicating the maximum noise level at which the model converged (success rate $> 90\%$) for different training settings. In each of the sub-figures, the parameter



(a) Highest ϵ for which convergence is achieved for different training strategies.



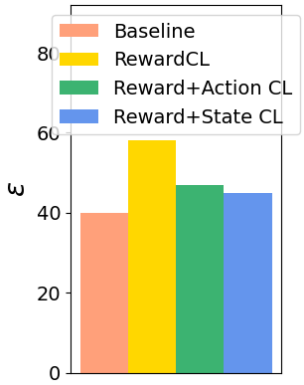
(b) Highest μ for which convergence is achieved for different training strategies.



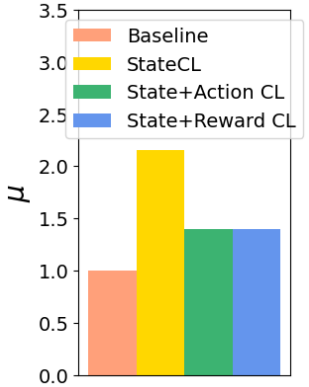
(c) Highest ν for which convergence is achieved for different training strategies.

Figure 4: Keep Away. The highest noise values that lead to convergence (agent success rate is greater than 90%) are shown above. We see that CL applied to a single parameter does better than the baseline training with no CL. When CL is applied to multiple parameters, the highest noise value is usually lower than applying CL to a single parameter. However, the same model now supports uncertainty in two different parameters for a slight reduction in the noise value.

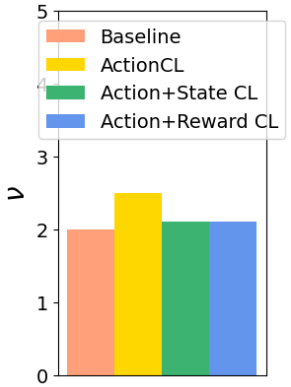
mentioned on the Y-axis is progressively increased until failure while the other parameters are set to zero.



(a) Highest ϵ for which convergence is achieved for different training strategies.



(b) Highest μ for which convergence is achieved for different training strategies.



(c) Highest ν for which convergence is achieved for different training strategies.

Figure 5: Physical Deception. The highest noise values that lead to convergence (agent success rate $> 90\%$) are shown above. We clearly see that CL applied to a single parameter does better than the baseline training with no CL. When CL is applied to multiple parameters, the highest noise value is usually lower than applying CL to a single parameter. However, the same model now supports uncertainty in two different parameters for a slight reduction in the noise value hence is useful.

7 Conclusion and Future Work

We investigate using curriculum learning to develop a robust multi-agent reinforcement learning model for multi-modal environment uncertainty. For this, we develop an efficient curriculum that slowly increases uncertainty of the uncertain parameters and finally achieves state-of-the-art robustness. It has surpassed the baseline (training with a fixed uncertainty with no curriculum learning) for state, reward and action uncertainty significantly both in the case of single uncertainty and multi-modal (two) uncertainty. We show results on three environments to show the validity of our method.

REFERENCES

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)*, pp. 11711–11718. IEEE, 2022. 1

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)*, pp. 1393–1399. IEEE, 2023. 1

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)*, pp. 548–552. IEEE, 2017. 2

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

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. 1, 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, 3, 16, 17

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

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, pp. 4213–4220, 2019a. 1, 4

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*, 2019b. 2

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

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994. 3

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, pp. 155–162, 2022. 3

Siddharth Mysore. Reward-guided curriculum for robust reinforcement learning. 2019. 2

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, 5

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

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

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

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*, pp. 1055–1063, 2020. 2

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, pp. 11308–11316, 2021. 2

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*, pp. 25–36, 2021. 2

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)*, pp. 5503–5510. IEEE, 2022. 2

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

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

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. ISSN 1939-3539. 2, 5

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

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

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

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 (eds.), *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2020. 1, 2, 4, 5, 8

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

8 Environment Details and Training Parameter Details

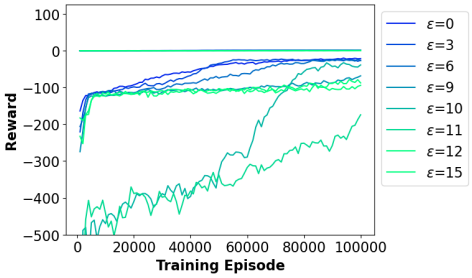
Cooperative navigation (CN): This is a cooperative game. There are 3 agents and 3 landmarks. Agents are rewarded based on how far any agent is from each landmark. Agents are penalized if they collide with other agents. So, agents have to learn to cover all the landmarks while avoiding collisions.

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.

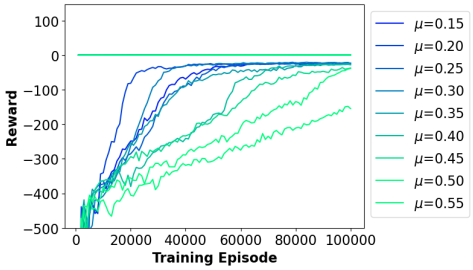
Physical deception (PD): This is a mixed cooperative and competitive task. There are 2 collaborative agents, 2 landmarks, and 1 adversary. Both the collaborative agents and the adversary want to reach the target, but only collaborative agents know the correct target. The collaborative agents should learn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

9 Reward Plots for Experiments

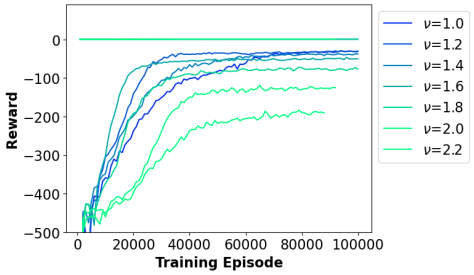
9.1 Cooperative Navigation Environment



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

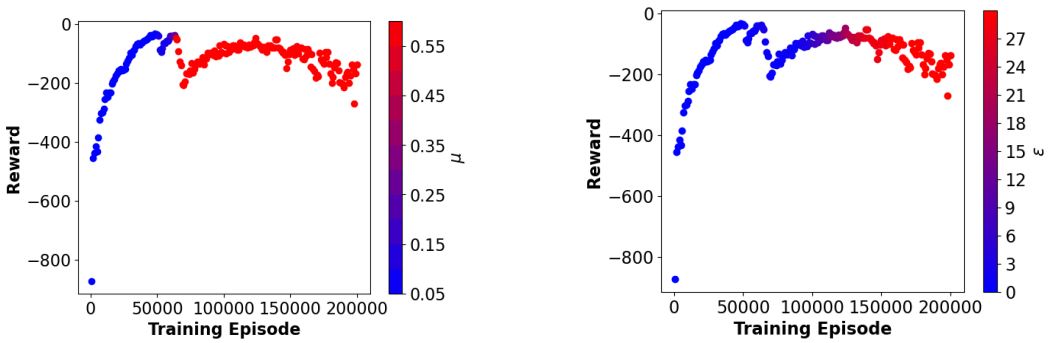


(b) Reward plot for baseline for various μ in state uncertainty. The baseline was able to learn only till μ = 0.5.



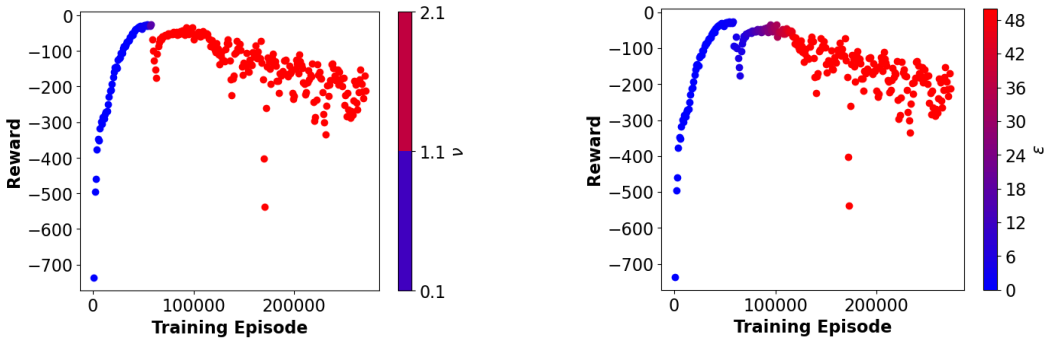
(c) Reward plot for baseline for various ν in action uncertainty. The baseline was able to learn only till μ = 2.0.

Figure 6: 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).



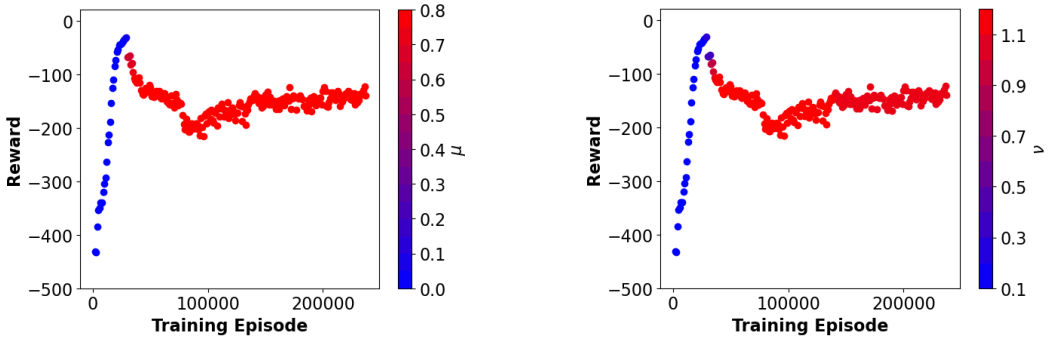
(a) Reward changes with μ **(b)** Reward changes with ϵ

Figure 7: 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

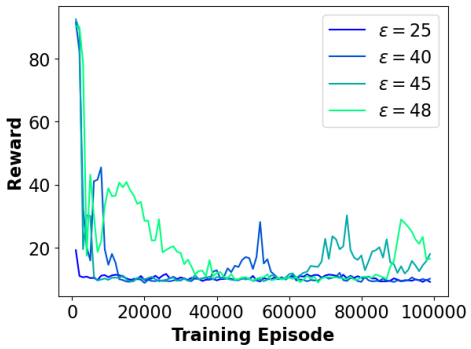
Figure 8: Reward plot for lookahead CL for the case of multiple uncertainties (reward and action) showing the changing reward for various ν (left) and ϵ (right).



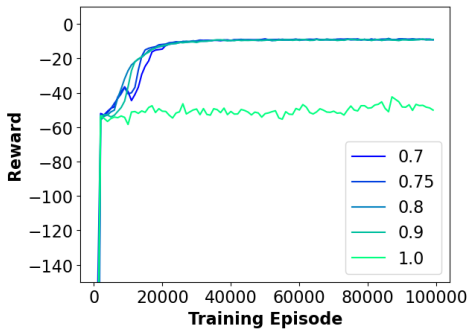
(a) Reward changes with μ **(b)** Reward changes with ν

Figure 9: 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).

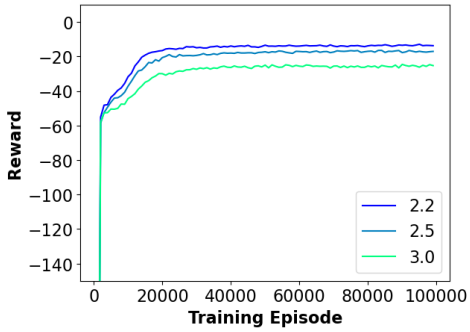
9.2 KEEP AWAY ENVIRONMENT



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

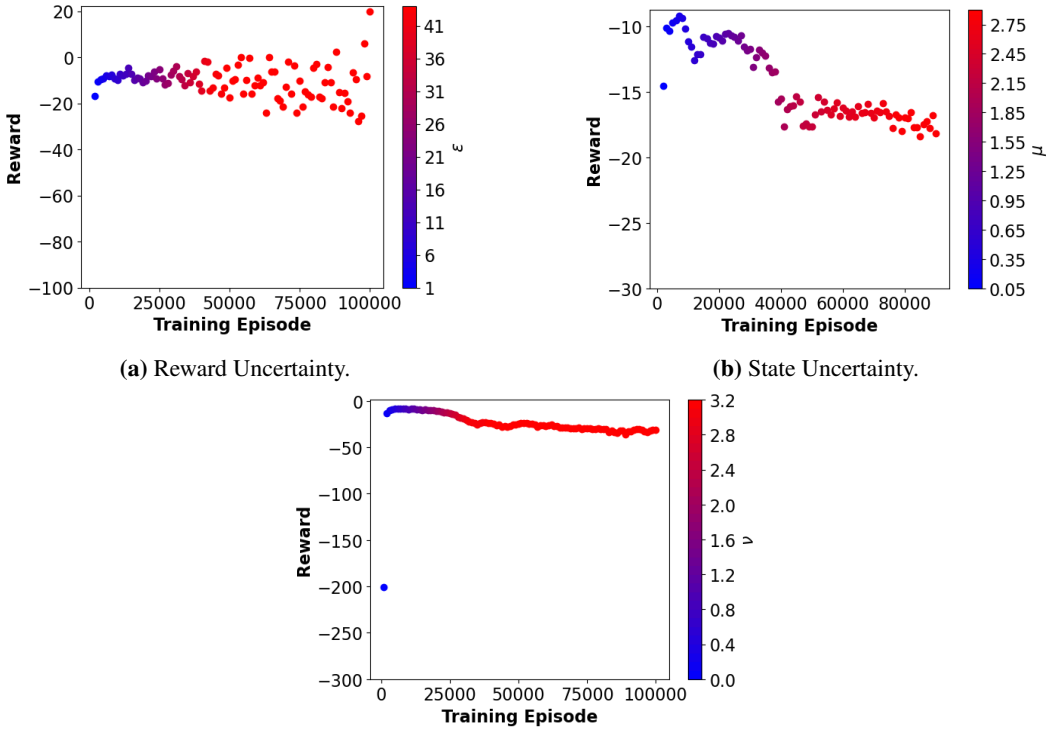


(b) Reward changes for various μ in state uncertainty.



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

Figure 10: 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 11: 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).

10 NASH EQUILIBRIUM FOR STATE UNCERTAINTY IN MARL

A nice proof for the conditional existence of Nash equilibrium is done in He et al. (2023) 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. $p : S \times A \to \Delta(S)$ 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 He et al. (2023).

10.1 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.

11 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 Kardeş et al. (2011) 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 Kardeş et al. (2011) 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 -

$$V_{\pi}^{i}(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^{t} r_{t}^{i} | s_0 = s, a_t^i \sim \pi^i(.|s_t), a_t^{-i} \sim \pi^{-i}(.|s_t)]$$

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 $\pi^* = \{\pi_1^*, \pi_2^*, \dots \pi_N^*\}$ 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 Kardeş et al. (2011). 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:

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

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

12 Proof for Theorem 1

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 $\mathbb{V}$