Robust Inverse Constrained Reinforcement Learning under Model Misspecification

Sheng Xu [1] **Guiliang Liu** [1 *]

Abstract

To solve safety-critical decision-making problems, Inverse Constrained Reinforcement Learning (ICRL) infers constraints from expert demonstrations and seeks to imitate expert preference by utilizing these constraints. While prior ICRL research commonly overlooks the discrepancy between the training and deploying environments, we demonstrate that such a discrepancy can significantly compromise the reliability of the inferred constraints and thus induce unsafe movements. Motivated by this finding, we propose the Robust Constraint Inference (RCI) problem and an Adaptively Robust ICRL (AR-ICRL) algorithm to solve RCI efficiently. Specifically, we model the impact of misspecified dynamics with an opponent policy and learn a robust policy to facilitate safe control in a Markov Game. Subsequently, we adjust our constraint model to align the learned policies to expert demonstrations, accommodating both soft and hard optimality in our behavioral models. Empirical results demonstrate the significance of robust constraints and the effectiveness of the proposed AR-ICRL algorithm under continuous and discrete domains. The code is available at `https://github.com/Jasonxu1225/AR-ICRL`.

1. Introduction

When deploying a Reinforcement Learning (RL) algorithm to practical applications, an important requirement is that the agent's behaviors should adhere to the constraints of given tasks (García & Fernández, 2015; Liu et al., 2021; Brunke et al., 2022). However, due to the complexity of real-world environments (e.g., auto-diagnosis, robot control, and autonomous driving), the underlying constraints are often time-dependent, context-sensitive, and relevant to human experience. How to accurately specify these constraints and facilitate safe control remains a critical challenge.

To address this challenge, Inverse Constrained Reinforcement Learning (ICRL) (Scobee & Sastry, 2020; Malik et al., 2021) aims to recover the constraints respected by expert agents from their demonstrations and learn the corresponding policies. Compared to directly imitating the expert behaviors, modeling constraints offer better interpretability of experts' preferences, and these constraints can potentially support downstream applications within similar domains.

To study the validity of a learned constraint, a common approach is to examine whether an RL policy can replicate expert demonstrations under this constraint. Existing ICRL methods (Liu & Zhu, 2022; 2023; 2024; Liu et al., 2023; Gaurav et al., 2023; Qiao et al., 2023) typically train and test their constraints within an identical environment. This configuration, nevertheless, contradicts the real-world settings, where a discrepancy often arises between the training dynamics and those encountered during deployment. Our study illustrates such a discrepancy is not trivial in safe control. As the discrepancy scales, the efficacy of a learned constraint is substantially undermined during deployment, thereby causing the agent to perform dangerous movements.

In this work, we investigate how the mismatched environmental dynamics influence the efficacy of constraints in terms of ensuring safe control. Motivated by our findings, we introduce the problem of Robust Constraint Inference (RCI). This problem requires the constraints learned in the training environment to generalize reliably to the testing environments under an uncertainty set.

To effectively enable RCI, we propose the Adaptively Robust Inverse Constrained Reinforcement Learning (AR-ICRL) algorithm. AR-ICRL alternatively solves:

1) Safety-Robust Policy Optimization that learns a control policy that consistently executes safe actions while maintaining robustness against the uncertainty of environmental dynamics. By leveraging the game-theoretic framework, we model the influence of uncertainty with adversarial attacks from an opponent policy, and the robust policy can be

[*]Corresponding Author [1]School of Data Science, The Chinese University of Hong Kong, Shenzhen, Guangdong, 518172, P.R. China . Correspondence to: Guiliang Liu <liuguiliang@cuhk.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

effectively learned by solving a two-player Markov game.

2) Adaptive Constraint Inference from a Mixture Policy that aims at aligning the inferred constraint with expert preference. To achieve this goal, we propose an objective that enables the cost function to maximize the likelihood of generating the expert demonstration under the maximum causal entropy (Ziebart et al., 2010) framework, considering the interplay between the robust policy and its adversarial counterpart. Our objective can be customized to accommodate both soft and hard optimality in behavioral models, which play critical roles in inverse control (Skalse & Abate, 2023).

Upon successfully replicating expert demonstrations with the mixture policy, the robust constraint can be extracted from imitating behaviors with standard ICRL techniques. In our experiment, we compare our AR-ICRL algorithm against other baselines across different scales and types of disturbances between the training and deploying dynamics. The results demonstrate that AR-ICRL can learn a safe control policy and the corresponding robust constraint.

Contribution. Our main contributions are: 1) We study the impact of discrepancies between the training and deployment environmental dynamics on the safety of control policies. Based on our findings, we define the problem of inferring robust constraints. 2) We introduce a novel AR-ICRL algorithm for inferring the safe control policy and the corresponding robust constraint based on the training environment and expert demonstrations. 3) We validate the effectiveness of the inferred constraints in ensuring safe control across a variety of testing environments.

2. Related Works

Robust Reinforcement Learning. Robust RL studies the approach to guarantee the robustness of control policy under uncertainties, disturbances, or structural changes within the environment (Moos et al., 2022). In this study, we primarily study the robustness to *model misspecification* where an agent is trained in one environment and performs in a different, perturbed version of the environment. In this literature, while previous works (Morimoto & Doya, 2005; Grau-Moya et al., 2016) mainly studied the discrete environments where the problem can be solved by robust dynamic programming, some recent works (Mankowitz et al., 2020b; Wang & Zou, 2021; Roy et al., 2017; Zhang et al., 2023) proposed robust RL algorithms for continuous control by optimizing the worst-case performance over the uncertainty set. Additionally, there is a parallel strand of research addressing the inverse problem in the context of model mismatch (Gangwani & Peng, 2020; Viano et al., 2021; 2022). Unlike previous works that primarily focus on the robustness in reward maximization, recent studies (Russel et al., 2020; Mankowitz et al., 2020a; Wang et al., 2022; Meng et al., 2023; Li et al., 2023) expanded the consideration of robustness to safety constraints. However, unlike prior works that study the *forward* control problem under safety-robustness, our work considers an *inverse* constraint inference problem.

Inverse Constrained Reinforcement Learning. Inverse Constrained Reinforcement Learning (ICRL) focuses on recovering the underlying constraint respected by expert agents from their demonstrations. Previous research predominantly concentrated on inferring implicit constraints by identifying the permissibility of actions within certain states. Scobee & Sastry (2020) proposed to infer implicit constraint set under the maximum entropy framework (Ziebart et al., 2008), which was restricted to discrete environments. Subsequent works (Malik et al., 2021; Gaurav et al., 2023) extended the approach to continuous environments by utilizing neural networks to approximate constraints. Towards uncertainty-aware constraint inference, McPherson et al. (2021); Baert et al. (2023) incorporated maximum causal entropy (Ziebart et al., 2010) principle into ICRL. Liu et al. (2023); Papadimitriou et al. (2023) applied variational inference and Bayesian Monte Carlo to infer the posterior distribution of constraints. Xu & Liu (2024) utilized the distributional Bellman operator and a flow-based trajectory generator for handling uncertainties. Notably, these works commonly focused on applying the inferred constraints within the same environment from which they were learned. In contrast, our work aims to infer robust constraints adaptable to various environments with misspecification, offering a more realistic application of ICRL.

3. Problem Formulation

Constrained Markov Decision Process. Our environment is based on a Constrained Markov Decision Process (CMDP) $M_c^{\mathrm{T}} := (\mathcal{S}, \mathcal{A}, \mathrm{T}, r, c, \epsilon, \mu_0, \gamma)$, where: 1) $\mathcal{S}$ and $\mathcal{A}$ denote the space of states and actions. 2) $\mathrm{T}(s'|s,a)$ defines the transition distribution. 3) $r(s,a)$ and $c(s,a)$ denote the reward and cost function (we assume $c \geq 0$). 4) ϵ defines the threshold of the constraint, where $\epsilon = 0$ refers to a hard constraint, enforcing absolute satisfaction, while $\epsilon > 0$ denotes a soft constraint, permitting a certain degree of constraint violation. 5) μ_0 denotes the initial state distribution. 6) $\gamma \in [0, 1)$ is the discount factor. In our empirical study, we mainly study an episodic setting where the game ends at some terminating state or time horizon.

Constrained Reinforcement Learning (CRL). Under a CMDP M_c^{T} with cost function c and transition function T, the goal of CRL is to find an optimal policy $\pi(\cdot|s)$ such that:

$$\arg\max_{\pi} \mathbb{E}_{\mu_0,\mathrm{T},\pi}\Big[\sum_{t=0}^{H}\gamma^t\Big(r(s_t,a_t)+\beta\mathcal{H}[\pi(\cdot|s_t)]\Big)\Big]$$
$$\text{s.t. } \mathbb{E}_{\mu_0,\mathrm{T},\pi}\Big[\sum_{t=0}^{H}\gamma^t c(s_t,a_t)\Big] \leq \epsilon \tag{1}$$

where H and $\mathcal{H}[\pi(\cdot|s_t)]$ denote the planning horizon and the causal entropy (Ziebart et al., 2010).

Definition 3.1. (Mismatched MDPs) We define a CMDP $M_c^{\mathrm{T}} := (\mathcal{S}, \mathcal{A}, \mathrm{T}, r, c, \epsilon, \mu_0, \gamma)$, to be mismatched with another CMDP $M_c^{\mathrm{T}'} := (\mathcal{S}, \mathcal{A}, \mathrm{T}', r, c, \epsilon, \mu_0, \gamma)$ if they differ only in transition functions (i.e., $\mathrm{T} \neq \mathrm{T}'$). We define a set of mismatched CMDPs as $\mathcal{M}_m = \{M_c^{\mathrm{T}}, M_c^{\mathrm{T}'}, \dots\}$, where their transition functions differ from each other.

Under this setting, we define the robustness of a policy:

Definition 3.2. (Absolute Policy Robustness) We claim a policy $\pi_\forall$ to have absolute robustness if it consistently satisfies the constraints across all possibly mismatched CMDPs:

$$\mathbb{E}_{\mu_0,\mathrm{T},\pi_\forall}\Big[\sum_{t=0}^{H}\gamma^t c(s_t,a_t)\Big] \leq \epsilon,\ \forall M_c^{\mathrm{T}} \in \mathcal{M}_m \quad (2)$$

However, identifying such absolutely robust policy presents a significant challenge and may not be feasible in many environments, which we show as follows:

Proposition 3.3. *Let $\{M_c^{\mathrm{T}}, M_c^{\mathrm{T}'}\}$ define two mismatched CMDPs in $\mathcal{M}_m$. Let Π define the space of policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ and $\mathcal{T}$ define the transition space. Let $Q^{c,\pi}_{M_c^{\mathrm{T}}}(s,a) = \mathbb{E}_{\mathrm{T},\pi}[\sum_{t=0}^{H} c(s_t,a_t) \mid s_0 = s, a_0 = a]$ and $V^{c,\pi}_{M_c^{\mathrm{T}}}(s) = \mathbb{E}_{a\sim\pi}[Q^{c,\pi}_{M_c^{\mathrm{T}}}(s,a)]$ define the value functions for discounted cumulative costs under policy π. Let $d^{\pi}_{M_c^{\mathrm{T}}}$ define the normalized occupancy measure by following policy π in the CMDP M_c^{T}. It holds that:*

(a) $\exists\pi \in \Pi$ and $\exists(\mathrm{T},\mathrm{T}') \in \mathcal{T}$ s.t. $\mathbb{E}_{\mu_0,\pi}[Q^{c,\pi}_{M_c^{\mathrm{T}}}(s,a)] \leq \epsilon$, but $\mathbb{E}_{\mu_0,\pi}[Q^{c,\pi}_{M_c^{\mathrm{T}'}}(s,a)] > \epsilon$.

(b) $\forall\pi \in \Pi$, If $d^{\pi}_{M_c^{\mathrm{T}}}(\mathrm{T}' - \mathrm{T})V^{c,\pi}_{M_c^{\mathrm{T}'}} > \frac{\epsilon(1-\gamma)}{\gamma}$, then $\mathbb{E}_{\mu_0,\pi}[Q^{c,\pi}_{M_c^{\mathrm{T}'}}(s,a)] > \epsilon$.

The proof is shown in Appendix B.1. Specifically, Proposition 3.3 (a) shows that there always *exists* a policy that satisfies constraints in one environment but violates the constraint in another. Proposition 3.3 (b) shows that *all* the policies in one environment may violate the constraint in another environment as long as their divergence is big enough.

Accordingly, deriving a policy solely from the learning environment and expecting it to consistently ensure safety across various deployment environments poses substantial challenges. A more realistic approach would be to assume that the discrepancy between the learning and testing environments is bounded under an uncertainty set.

Definition 3.4. (Relative Policy Robustness) We define a policy, denoted as $\pi_{\bar\forall}$, to have relative robustness if it consistently satisfies the constraints across all mismatched CMDPs which have transition dynamics being in the uncertainty set $\mathcal{T}_\alpha^L = \{\alpha\mathrm{T}^L + (1-\alpha)\bar{\mathrm{T}},\ \forall\bar{\mathrm{T}} \in \mathcal{T}\}$, where $0 \leq \alpha \leq 1$. Here T^L signifies the transition of a reference CMDP $M_c^{\mathrm{T}^L}$. Such a relatively robust policy should satisfy:

$$\mathbb{E}_{\mu_0,\mathrm{T},\pi_{\bar\forall}}\Big[\sum_{t=0}^{H}\gamma^t c(s_t,a_t)\Big] \leq \epsilon,\ \forall M_c^{\mathrm{T}} \in \mathcal{M}_c^{\mathcal{T}_\alpha^L} \quad (3)$$

where $\mathcal{M}_c^{\mathcal{T}_\alpha^L}$ denotes the set of CMDPs having their transitions being in the uncertainty set $\mathcal{T}_\alpha^L$.

In our problem setting, T^L denotes the learning environment, which serves as the centroid of the uncertainty set, and α measures the size of the uncertainty set. Note that the set $\mathcal{T}_\alpha^L$ is equivalent to the $(s;a)$-rectangular uncertainty set (Iyengar, 2005) centered around T^L. From now on, we focus on the *relative robustness* unless specifically stated.

Inverse Constrained Reinforcement Learning (ICRL). In the ICRL problem, the cost signals $c(s,a)$ are not readily observable. Instead, the learner has access to expert demonstrations $\mathcal{D}_e$ which is distributed according to the optimal expert policy $\pi^*_{M_{c^*}^{\mathrm{T}^e}}$ that implicitly respects the underlying ground-truth constraints $\mathbb{E}_{\mu_0,\mathrm{T}^e,\pi^*}[\sum_{t=0}^{H}\gamma^t c^*(s_t,a_t)] \leq \epsilon$.

Under this setting, the goal of ICRL is to recover this constraint from expert demonstrations $\mathcal{D}_e$. For brevity, we define the constraint C, parameterised by c, ϵ and T, as a mapping from a policy π to binary variables (feasibility of π). In alignment with previous works (Malik et al., 2021), we study hard constraints such that $\epsilon = 0$, and T can be estimated by interacting with the environment under the online RL setting. Consequently, inferring the constraints can be simplified to learn the cost function c.

Building upon the insights from Proposition 3.3, we contend that it is significant for ICRL algorithms to consider the potential discrepancies in transition dynamics between the training and applied environments. As Figure 1 shows, even if we recover the true constraints c^* from expert demonstrations $\mathcal{D}_e$, there is no guarantee that the corresponding optimal policy $\pi^*_{M_{c^*}^{\mathrm{T}}}$ can remain safe in a different mismatched CMDP. This limitation poses a substantial challenge to the effective deployment of ICRL in practical applications.

To handle the challenge and learn an optimal globally robust policy, we propose the following Robust Constraint Inference problem:

Definition 3.5. (Robust Constraint Inference (RCI)) Suppose that there exists a class of transitions $\mathcal{T}_\alpha^L$ centered around the learning transition T^L. Given a demonstration dataset $\mathcal{D}_e$ that respects c^* (unknown). The goal of the RCI is to infer a robust constraint $c_\forall$ such that the optimal policy trained under the corresponding CMDP $M_{c_\forall}^{\mathrm{T}^L}$ is safe across all environments within $\mathcal{M}_{c^*}^{\mathcal{T}_\alpha^L}$ (i.e., satisfying Eq. (3)).

Note that we do not require explicit knowledge of the expert policy and environment. The core requirement is that these

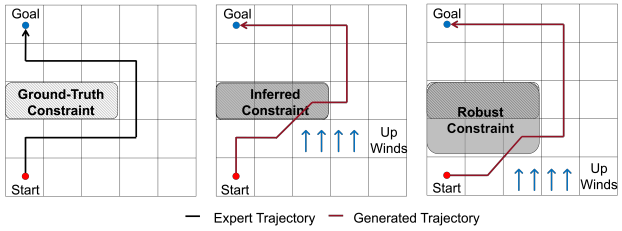

Figure 1. An example of the robust constraint under windy Gridworld environments (check Section 5.1 for details). The training environment (left) has no wind, while the testing environments (middle and right) have stochastic upward winds. The left figure shows the expert trajectory and ground-truth constraint in the training environment. The middle figure shows although the inferred constraint matches the ground-truth constraint, the learned policy still violates constraints during testing. The right figure shows the robust constraint prevents violating the *ground-truth* constraint.

demonstrations respect the ground-truth constraints. For simplicity, we assume the expert environment is identical to the learning environment in this work.

4. Learning Robust Constraint via Inverse Reinforcement Learning

To infer robust constraints that can be generalized to the environment with mismatched dynamics, we propose the Adaptively-Robust Inverse Constrained Reinforcement Learning (AR-ICRL) algorithm. Specifically, AR-ICRL solves the RCI problem by alternatively solving: 1) a forward control problem to perform policy update under the mismatched dynamics and the inferred constraint (Section 4.1); 2) a backward behaviors alignment problem that minimizes the discrepancy between the policy model and expert demonstrations by updating constraints (Section 4.2). AR-ICRL converges when the disturbed policy matches the expert policy. The robust constraint can be extracted from the learned robust policy (Section 4.3).

4.1. Safety-Robust Policy Optimization

In this section, we derive a game-theoretic method to learn the robust policy that can consistently satisfy the inferred constraint across all mismatched CMDP $M \in \mathcal{M}_c^{\mathcal{T}_\alpha^L}$.

Following Malik et al. (2021), the permissibility indicator $\phi(s,a)$ denotes the probability that performing an action a under the state s is feasible, and the cost is defined as $c(s,a) = -\log \phi(s,a)$. Under this definition, adhering to constraints can be interpreted as ensuring feasibility. The CRL problem (Obj. (1)) can then be formalized as follows:

$$\max_{\pi\in\Pi} \mathbb{E}\Big[\sum_{t=0}^{H}\gamma^t\Big(r(s_t,a_t)+\beta\mathcal{H}[\pi(\cdot|s_t)]\Big) \mid \pi, M^L\Big]$$
$$\text{s.t. } \mathbb{E}\Big[\sum_{t=0}^{H}\gamma^t \log\phi(s_t,a_t) \mid \pi, M^L\Big] \geq \epsilon^e(\phi) \quad (4)$$

where $\epsilon^e(\phi) = \mathbb{E}_{\mathcal{D}_e}\Big[\sum_{t=0}^{H}\gamma^t\log\phi(s_t^e,a_t^e)\Big]$, and M^L denotes the learning CMDP with transition T^L. The Lagrange dual problem is given by:

$$g(\lambda) = \max_{\pi\in\Pi}\mathbb{E}\Big[\sum_{t=0}^{H}\gamma^t\Big(r(s_t,a_t)+\beta\mathcal{H}[\pi(\cdot|s_t)]\Big)+ \lambda\log\phi(s_t,a_t) \mid \pi, M^L\Big] - \lambda\epsilon^e(\phi) \quad (5)$$

where $\lambda \geq 0$ denotes the Lagrange multiplier. Since our objective is to find the safety-robust policy that can respect the constraint across all mismatched CMDPs $\mathcal{M}^{\mathcal{T}_\alpha^L}$ (the subscript c is omitted for brevity), we optimize the policy to ensure constraint satisfaction under the worst-case transition among $\mathcal{T}_\alpha^L$ as follows:

$$\max_{\pi\in\Pi}\min_{\bar{M}\in\mathcal{M}^{\mathcal{T}_\alpha^L}}\mathbb{E}\Big[\sum_{t=0}^{H}\gamma^t\lambda\log\phi(s_t,a_t) \mid \pi,\bar{M}\Big]+ \quad (6)$$
$$\mathbb{E}\Big[\sum_{t=0}^{H}\gamma^t\Big(r(s_t,a_t)+\beta\mathcal{H}[\pi(\cdot|s_t)]\Big) \mid \pi, M^L\Big] - \lambda\epsilon^e(\phi)$$

To solve the problem, inspired by Viano et al. (2021), the minimization over the environmental classes can be interpreted as the minimization over the opponent policies π^{op} that take over the agent and perform the worst actions:

Proposition 4.1. *For the mismatched CMDPs $\mathcal{M}^{\mathcal{T}_\alpha^L}$ with the transitions bounded in the uncertainty set $\mathcal{T}_\alpha^L = \{\alpha\mathrm{T}^L + (1-\alpha)\bar{\mathrm{T}},\ \forall\bar{\mathrm{T}}\in\mathcal{T}\}$, we have*

$$\min_{\bar{M}\in\mathcal{M}^{\mathcal{T}_\alpha^L}}\mathbb{E}\Big[\sum_{t=0}^{H}\gamma^t\log\phi(s_t,a_t) \mid \pi,\bar{M}\Big] \quad (7)$$
$$\leq \min_{\pi^{op}\in\Pi}\mathbb{E}\Big[\sum_{t=0}^{H}\gamma^t\log\phi(s_t,a_t) \mid \alpha\pi+(1-\alpha)\pi^{op}, M^L\Big]$$

The proof can be found in Appendix B.2. For clarity, we use π^{pl} to represent the original task policy, which considers both rewards and costs (safety), and π^{op} to denote the opponent policy that specifically targets safety. Since minimizing the objective can be effectively achieved by minimizing its upper bound, the proposition enables us to formulate the Lagrangian problem into a two-player Markov game as follows:

$$\min_{\lambda}\max_{\pi^{pl}\in\Pi}\min_{\pi^{op}\in\Pi} \quad (8)$$
$$\mathbb{E}\Big[\lambda\sum_{t=0}^{H}\gamma^t\log\phi(s_t,a_t) \mid \alpha\pi^{pl}+(1-\alpha)\pi^{op}, M^L\Big]+$$
$$\mathbb{E}\Big[\sum_{t=0}^{H}\gamma^t(r(s_t,a_t)+\beta\mathcal{H}[\pi^{pl}(\cdot|s_t)]) \mid \pi^{pl}, M^L\Big] - \lambda\epsilon^e(\phi)$$

where a^{pl} is the action of the player policy π^{pl} and a^{op} is chosen by the opponent policy π^{op} as the adversarial attack.

4.2. Adaptive Constraint Inference from Mixture Policy

This section focuses on inferring the underlying constraint c under the framework of a two-player Markov game. Specifically, given the expert demonstration dataset $\mathcal{D}_e$, we need to infer the underlying constraint that the expert adheres to in the presence of the opponent policy π^{op}.

Following previous ICRL works (Scobee & Sastry, 2020; Malik et al., 2021), we update the feasibility function ϕ_ω (parameterized with ω) by maximizing the likelihood of generating expert demonstrations. Inspired by Gleave & Toyer (2022), we define the discounted likelihood as follows:

Definition 4.2. For a trajectory $\tau = (s_0, a_0, s_1, \ldots, s_{H+1})$, its discounted likelihood with policy π and transition T is:

$$p^{\pi}_{M^{\mathrm{T}}}(\tau) = \mu_0(s_0) \prod_{t=0}^{H} \mathrm{T}(s_{t+1}|s_t, a_t)\pi(a_t|s_t)^{\gamma^t} \quad (9)$$

Intuitively, when $\gamma = 1$, the discounted likelihood is equivalent to the traditional likelihood of τ under the policy π. For $\gamma \leq 1$, the probabilities of performing later actions in the trajectory are regularized towards 1 by raising them to γ^t.

We can then derive the gradient of the log-likelihood of generating expert demonstrations, which accommodates both hard and soft behaviors from the opponent. We'll start with the scenario involving the hard optimal opponent.

Assumption 4.3. On a state $\tilde{s}$, if $\tilde{a} = \arg\max Q^{c,*}_{M^{\mathrm{T}}_c}(\tilde{s}, a)$ where $Q^{c,*}_{M^{\mathrm{T}}_c}(s, a) = \mathbb{E}_{\pi^c_{\max}, \mathrm{T}}[\sum_{t=0}^{H} \gamma^t c(s_t, a_t) \mid s_0 = s, a_0 = a]$ and $\pi^c_{\max}$ defines the cost maximization policy, then the expert agent will not select $\tilde{a}$, i.e., $\pi^e(\tilde{a}|\tilde{s}) = 0$.

Intuitively, we assume the expert preference π^e will not overlap with that of the deterministic cost maximizing policy $\pi^c_{\max}$. This assumption is reasonable since cost maximization action is not safe and thus sub-optimal for the expert agent. Based on this assumption, we can show that:

Proposition 4.4. *(Log-likelihood gradient with hard optimal opponent) Let $\pi^{mix} = \alpha\pi^{pl} + (1-\alpha)\pi^{op}$ define the mixture policy. Let $\mathbb{E}_{\mathcal{D}_e}\left[\log p^{\pi^{mix}}_{M^{\mathrm{T}}_c}(\tau)\right]$ define its likelihood of generating the expert demonstration in the training environment M^L with estimated costs function $c(\cdot) = -\log \phi_\omega(\cdot)$. Let π^{op} denote a deterministic cost maximization policy, and the player policy π^{pl} follows the soft optimal representation:*

$$\pi^{pl}(a_t|s_t) = \phi_\omega(s_t, a_t)^{\lambda}. \quad (10)$$
$$\exp\left(\gamma \mathbb{E}_{s_{t+1}}[V^{soft,pl}_{M^{\mathrm{T}}_c}(s_{t+1})] + r(s_t, a_t) - V^{soft,pl}_{M^{\mathrm{T}}_c}(s_t)\right)$$

where the value function for player policy is defined as:

$$V^{soft,pl}_{M^{\mathrm{T}}_c}(s) = \mathbb{E}\Big[\sum_{t=0}^{H} \gamma^t [r(s_t, a_t) + \lambda \log \phi_\omega(s_t, a_t) + \beta\mathcal{H}(\pi^{pl}(\cdot|s_t))] \mid s_0 = s\Big] \quad (11)$$

The gradient of our log-likelihood $\nabla_\omega \mathbb{E}_{\mathcal{D}_e}\left[\log p^{\pi^{mix}}_{M^{\mathrm{T}}_c}(\tau)\right]$ can be represented by:

$$\mathbb{E}_{\mathcal{D}_e}\Big[\sum_{t=0}^{H} \gamma^t \lambda \nabla_\omega \log \phi_\omega(\cdot)\Big] - \mathbb{E}_{\pi^{pl}}\Big[\sum_{t=0}^{H} \gamma^t \lambda \nabla_\omega \log \phi_\omega(\cdot)\Big] \quad (12)$$

The proof is in Appendix B.3. Note that even though the opponent's representation is left out in the gradient due to Assumption 4.3, π^{op} implicitly influences the results because of mixture sampling ($\pi^{mix} = \alpha\pi^{pl} + (1-\alpha)\pi^{op}$). Under this setting, the opponent policy π^{op} serves as a worst-case safety attacker, aiming to maximize cumulative costs. Conversely, the player policy π^{pl} incorporates causal entropy into its decision-making process, leading to the adoption of a softmax representation (Ziebart et al., 2010), which aligns well with our forward problem (Obj. (8)). In our implementation, we mainly follow this setting in a discrete environment where examining the value of each action is computationally tractable (see Appendix A.1).

Additionally, in Equation (10), $\phi(s_t, a_t) \in [0, 1]$ determines whether performing action a_t under the state s_t is feasible, and λ effectively controls its strength of behavior regularization (larger λ indicates stronger constraint). Notably, we have $\pi^{pl}(a_t|s_t) = 0$ when $\phi(\cdot) = 0$, which is consistent with the enforcement of hard constraints.

To drive comprehensive results, we derive an additional objective to update ϕ_ω when the opponent policy π^{op} is also soft-optimal. This approach is independent of Assumption 4.3, broadening the applicability of our methods.

Proposition 4.5. *(Log-likelihood gradient with soft optimal opponent) Let $\pi^{mix} = \alpha\pi^{pl} + (1-\alpha)\pi^{op}$ define the mixture policy. Let the player policy π^{pl} follow the soft optimal representation (Equation (10)), and the opponent policy π^{op} follow the soft optimal representation as follows:*

$$\pi^{op}(a_t|s_t) = \quad (13)$$
$$\exp\left(\gamma \mathbb{E}_{s_{t+1}}[V^{soft,op}_{M^{\mathrm{T}}_c}(s_{t+1})] - \log \phi_{\omega,t} - V^{soft,op}_{M^{\mathrm{T}}_c}(s_t)\right)$$
$$V^{soft,op}_{M^{\mathrm{T}}_c}(s) = \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t [-\log \phi_{\omega,t} + \beta\mathcal{H}(\pi^{op}(\cdot|s_t))] \,|\, s_0 = s\right]$$

where for brevity, we denote $\phi_{\omega,t} = \phi_\omega(s_t, a_t)$. We can derive a lower bound for $\mathbb{E}_{\mathcal{D}_e}[\log p^{\pi^{mix}}_{M^{\mathrm{T}}_c}(\tau)]$:

$$\mathbb{E}_{\mathcal{D}_e}\left[\log p^{\pi^{mix}}_{M^{\mathrm{T}}_c}(\tau)\right] \geq \mathcal{L}(\mathcal{D}_e, M^{\mathrm{T}}_c, \alpha) = \quad (14)$$
$$\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H} \gamma^t \alpha \left(Q^{soft,pl}_{M^{\mathrm{T}}_c}(s_t, a_t) - V^{soft,pl}_{M^{\mathrm{T}}_c}(s_t)\right)\right] +$$
$$\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H} \gamma^t (1-\alpha) \left(Q^{soft,op}_{M^{\mathrm{T}}_c}(s_t, a_t) + V^{soft,op}_{M^{\mathrm{T}}_c}(s_t)\right)\right] + \mathbb{C}$$

where $\mathbb{C}$ indicates a constant. The gradient of our log-likelihood $\nabla_\omega \mathcal{L}(\mathcal{D}_e, M_c^{\mathrm{T}}, \alpha)$ can be represented by:

$$\mathbb{E}_{\mathcal{D}_e}[\sum_{t=0}^{H} \gamma^t(\alpha\lambda + \alpha - 1)\nabla_\omega \log \phi_{\omega,t}] - \tag{15}$$
$$\alpha\lambda \mathbb{E}_{\pi^{pl}}[\sum_{t=0}^{H} \gamma^t \nabla_\omega \log \phi_{\omega,t}] + (1-\alpha)\mathbb{E}_{\pi^{op}}[\sum_{t=0}^{H} \gamma^t \nabla_\omega \log \phi_{\omega,t}]$$

The proof is shown in Appendix B.4. When $\alpha = 1$, it simplifies to the traditional ICRL objective. In this framework, we can update $\phi_\omega(\cdot)$ by maximizing the lower bound of log-likelihood $\mathbb{E}_{\mathcal{D}_e}[\log p^{\pi^{mix}}_{M_c^{\mathrm{T}}}(\tau)]$. In our implementation, we follow this setting in the continuous state-action space (Appendix A.2) where softmax policy representation performs better than hardmax ones (Haarnoja et al., 2018).

4.3. Extract Robust Constraint

Robust Constraint Distillation. In the proposed AR-ICRL algorithm, we infer a robust policy by alternatively updating the policy models (π^{pl} and π^{op} (see Section 4.1) and the permitsibility function ϕ_ω (representing constraint, see Section 4.2) until the mixture policy π^{mix} can reproduce expert demonstration (Line 8 in Algorithm 1). After this process, the player's policy π^{pl} is a robust policy that follows expert movements under the influence of adversarial attack from π^{op}. However, our goal is to infer a constraint that enables safe control without modeling the opponent, to achieve it, we consider distilling the robustness from policy to constraint. Specifically, we use π^{pl} as the expert policy and run regular ICRL algorithms to recover the robust cost function $c_\forall = -\log \phi_\forall$ in the learning environment M^L. To accelerate learning, we can initialize $\phi_\forall$ with the estimated ϕ_ω, and adapt the scale of constraints (represented by $\phi_\forall$) until a regular imitating policy $\hat{\pi}$ can reproduce the robust behaviors π^{pl}.

Proposition 4.6. *(Constraint Robustness) Let $\pi^{me}_{\tilde{c}}$ define the optimal policy under the learning CMDP $M_{\tilde{c}}^{\mathrm{T}^L}$ with the cost function $\tilde{c}$ inferred by MEICRL. Let $\pi^{ro}_{\hat{c}}$ define the optimal policy under $M_{\hat{c}}^{\mathrm{T}^L}$ under the robust cost function $\hat{c}$ inferred by Algorithm 1. Let $V^{c,\pi}_{M_{c^*}^{\mathrm{T}'}}(s)$ define the value function of the true cost under the CMDP $M_{c^*}^{\mathrm{T}'}$ and policy π. Assume that $\tilde{c} = c^*$, and $\mathbb{E}_{\mu_0}[V^{c,\pi^e}_{M_{c^*}^{\mathrm{T}^e}}(s)] = \epsilon$. The following inequality must hold:*

(a) $\exists \mathrm{T}' \in \{\mathrm{T} : \max_{s,a} |\mathrm{T} - \mathrm{T}^L| \leq 2(1-\alpha)\}$ *such that* $\mathbb{E}_{\mu_0}[V^{c,\pi^{me}_{\tilde{c}}}_{M_{c^*}^{\mathrm{T}'}}(s)] \geq \epsilon$.

(b) $\forall \mathrm{T}' \in \{\mathrm{T} : \max_{s,a} |\mathrm{T} - \mathrm{T}^L| \leq 2(1-\alpha)\}$ *such that* $\mathbb{E}_{\mu_0}[V^{c,\pi^{ro}_{\hat{c}}}_{M_{c^*}^{\mathrm{T}'}}(s)] \leq \epsilon$.

The proof is in Appendix B.5. This demonstrates that the standard MEICRL algorithm faces challenges in deducing the robust constraint when confronted with perturbed environments. In contrast, the proposed algorithm is able to ensure the constraint robustness in the face of such environmental mismatches.

Leveraging the aforementioned ideas for robust constraint inference, Algorithm 1 shows the procedure of AR-ICRL.

Algorithm 1 Adaptively Robust Inverse Constrained Reinforcement Learning (AR-ICRL)

1: **Input:** Expert trajectories $\mathcal{D}_e$, opponent strength $1-\alpha$, learning environment M^L and stopping criteria ξ.
2: Initialize player and opponent policies (π^{pl} and π^{op}).
3: Set $\forall(s,a), \phi_\omega(s,a) = 1$ (e.g., $c_\omega(\cdot) = 0$).
4: **repeat**
5: Update π^{pl} and π^{op} with objective (8).
6: Collect nominal trajectories $\hat{\mathcal{D}}$ by rollout with $\pi^{mix} = \alpha\pi^{pl} + (1-\alpha)\pi^{op}$ in the M^L.
7: Update c_ω by ascending the gradient (12) or (15).
8: **until** $|\rho_{\hat{\mathcal{D}}} - \rho_{\mathcal{D}_e}| \leq \xi$ (ρ is the occupancy measure).
9: Set π^{pl} as the expert policy, and extract the robust cost $c_\forall$ in M^L with classical ICRL algorithms.
10: **Output:** Robust policy π^{pl}, robust constraint $c_\forall$.

5. Empirical Evaluation

In this section, we empirically evaluate the efficacy of the proposed AR-ICRL algorithm in both discrete and continuous environments under transition dynamics mismatch. Furthermore, we furnish a series of visualization outcomes to offer a more in-depth insight into AR-ICRL's effectiveness.

Experimental Settings. Our experiment results are mainly based on a public ICRL benchmark (Liu et al., 2023). However, in contrast to previous ICRL works that use the same environment for training and testing, our study evaluates our model in testing environments where their dynamics differ from those used for training. This discrepancy allows us the study whether the robust constraint can facilitate safe control. Throughout our experiment, we mainly use the following metrics: 1) *Constraint Violation Rate* measures the probability that a policy violates the constraint in a trajectory, which is the top concern in this work. 2) *Episode Costs* accumulates the total costs in the whole episode. 3) *Feasible Rewards* calculates the total rewards obtained by the agent before violating any constraints.

Comparison Methods. We utilize the following baselines besides the proposed AR-ICRL: 1) *Maximum Entropy Inverse Constrained Reinforcement Learning* (**MEICRL**) (Malik et al., 2021) follows the maximum entropy (Ziebart et al., 2008) framework with PPO-Lagrange method for constrained policy optimization. 2) *Binary Classifier Constraint Learning* **(BC2L)** simply utilizes a binary classifier as the constraint model. 3) *Variational ICRL* **(VICRL)** (Liu et al., 2023) captures the epistemic uncertainty

in constraint inference using a Beta distribution. 4) *Inverse Reachable Constrained Optimization* (**IRCO**) extends the Reachable Constrained Optimization (RCO) (Yu et al., 2022) with the maximum entropy inverse constraint inference, which utilizes reachability analysis and feasible sets for constrained policy optimization.

5.1. Model Performance in Discrete Environments

In this experiment, we develop three distinct Gridworld environments, each characterized by unique constraints. As depicted in the top row of Figure 2, the primary task for the agent in these environments is to navigate from an initial location (red) to a designated target (blue), while avoiding the unobservable constraints (black). Given the expert demonstration, we *train* the agents under the transition dynamics *without disturbances* and *evaluate* these agents under *upward winds (disturbances)*. Specifically, the scale of t wind is controlled by a predetermined probability (denoted as p_w). A stronger wind (with larger p_w) exerts an upward force on the agent's selected action with a higher probability. Based on this discrete environment, we compare our method exclusively with the well-known MEICRL method to examine the significance of robustness in ICRL.

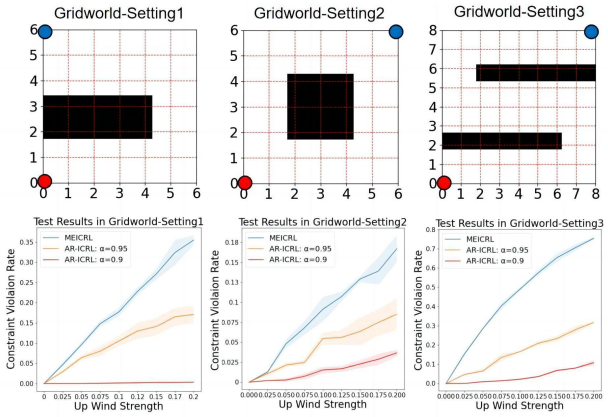

Figure 2. Three Gridworld settings and the testing performance of different methods under different levels of upwind.

Figure 2 shows the constraint violation rate in three stochastic windy Gridwords with different levels of wind strength p_w during testing. The corresponding reward curve is shown in Figure 6 in Appendix D.1. The results show that as the wind strength p_w increases, all the methods exhibit a greater probability of violating constraints across the environments. However, by incorporating the safety attacks into the training process, AR-ICRL shows a notably lower incidence of constraint violations compared to MEICRL. It also benefits the accumulation of higher feasible rewards by achieving the goal while avoiding the constraint. Moreover, larger opponent strength (lower α) can enhance the policy's safety under the mismatch between training and testing environments. This indicates the significance of modeling the adversarial attacks from the introduced opponent.

5.2. Model Performance in Continuous Environments

Based on the ICRL benchmark (Liu et al., 2023), we conduct experiments on three continuous robot control tasks with predefined constraints, including Blocked Half-Cheetah, Blocked Ant, and Crippled Walker. For *Blocked Half-Cheetah* and *Blocked Ant* tasks, we implement a constraint that confines the X-coordinate position of the agents to ≥ 3, and for the *Crippled Walker* task, we impose a constraint on the thigh angle θ of the agent, limiting it to $|\theta| \leq 0.6$. Each experiment is repeated with four random seeds, over which the mean ± standard deviation (std) results are reported. Additionally, for fairness and simplicity, we keep $\alpha = 0.95$ in the AR-ICRL across the environments. Please check Appendix C.1 for more environmental details.

To assess the model's performance under misspecified environments, we train the agents in a noise-free environment. Subsequently, we test their performance by introducing three different types of noises to simulate the discrepancies that might be encountered in practice: 1) *Fully random noise*: We integrate the Gaussian noise into the transition function, formulated as $p_{\mathcal{T}}(s_{t+1}|s_t, a_t) = T(s_t, a_t) + \mathcal{N}(\mu, \sigma)$. This method introduces randomness across the entire spectrum of state transitions, introducing variability in the agents' responses. 2) *Partially random noise*: This type of noise differs from the fully random variant by being more targeted. Instead of affecting the entire agent, it is applied only to the specific constrained feature in the state space (e.g., the position of the Blocked Half-cheetah). 3) *Attack noise:* This type of noise encourages the agent to violate safety constraints (e.g., the noise in the Blocked Half-Cheetah environment is applied to push the agents backward). For each type of noise, we define three scales - large, medium, and small - to represent varying degrees of mismatch between the training and testing environments. Note that the fully random noise remains consistent across the environments whereas partially random noise and attack noise are customized for each specific task (refer to Appendix C.2). Under such designs, we evaluate the model's adaptability and robustness under different scales of environmental discrepancy.

Table 1 shows the evaluation results with large-scale noises. Check Table 3, Table 4, and Table 5 in Appendix D.2 for the results without noise, and with medium and small scales of noises. The bolding value means the best result (the highest reward, lowest cost, and lowest violation rate) in each setting. We can find that almost all the methods encounter challenges in maintaining consistent safety when tested in a disparate environment, which highlights the significance of considering the potential mismatch during the training process. Moreover, larger randomness tends to result in a more pronounced decrease in model performance. How-

Table 1. Evaluation results in three environments with large-scale random noises. Each value is reported as the mean ± standard deviation for 50 episodes and 4 seeds.

Env	Method	Large Scale Fully Random Noise			Large Scale Partially Random Noise			Large Scale Attack Noise		
		Feasible Reward	Episode Cost	Constraint Violation Rate	Feasible Reward	Episode Cost	Constraint Violation Rate	Feasible Reward	Episode Cost	Constraint Violation Rate
Blocked Half-Cheetah	MEICRL	464.3±140.1	144.0±89.2	40%±18%	2431.2±2032.8	11.3±25.2	48%±12%	1223.6±1137.0	308.2±216.8	76%±23%
	BC2L	591.8±117.6	261.1±83.8	60%±6%	708.6±812.0	95.2±193.0	65%±10%	22.5±7.5	973.1±12.7	100%±0%
	VICRL	426.0±236.1	266.4±163.7	52%±25%	1668.4±1583.5	37.9±105.5	50%±8%	803.9±712.1	411.6±211.2	80%±13%
	IRCO	540.4±56.9	133.2±49.4	43%±13%	2682.3±2185.1	**8.4±21.4**	43%±8%	1476.2±428.5	78.8±16.2	72%±13%
	AR-ICRL	**727.8±87.3**	**90.6±10.1**	**31%±10%**	**2968.9±1842.7**	10.1±27.6	**34%±2%**	**1691.0±398.4**	**47.6±16.2**	**63%±8%**
Blocked Ant	MEICRL	1950.1±551.8	130.7±13.7	59%±5%	3472.8±399.0	96.5±26.6	70%±5%	1420.2±1223.1	154.0±103.4	56%±30%
	BC2L	1418.9±227.6	114.9±22.2	58%±9%	**6218.7±1089.1**	88.7±45.1	65%±8%	5487.8±2880.4	146.1±123.5	55%±42%
	VICRL	1388.3±82.4	130.5±15.8	68%±4%	5650.6±2227.9	91.1±57.6	69%±12%	**5698.3±4870.1**	191.3±169.5	54%±45%
	IRCO	1702.0±288.4	118.1±41.1	60%±10%	3508.4±546.5	91.8±38.1	67%±7%	1780.5±1123.9	178.6±110.9	58%±20%
	AR-ICRL	**2326.2±56.8**	**109.3±26.6**	**53%±4%**	4382.9±366.3	**75.4±17.6**	**59%±6%**	2498.1±1075.7	**113.8±98.0**	**44%±16%**
Crippled Walker	MEICRL	45.6±8.8	4.45±2.86	38%±18%	223.0±69.7	11.5±8.1	74%±24%	93.1±34.7	16.7±11.0	75%±39%
	BC2L	46.6±6.6	4.73±1.77	39%±17%	211.4±107.7	11.8±6.1	77%±17%	36.5±16.0	14.8±9.7	71%±39%
	VICRL	50.8±5.9	3.54±1.54	40%±8%	259.8±39.7	9.4±4.1	74%±5%	24.7±9.0	42.4±47.7	74%±22%
	IRCO	44.2±2.5	**3.43±2.67**	34%±16%	171.1±21.8	11.4±9.2	69%±24%	63.4±85.2	30.5±20.1	68%±14%
	AR-ICRL	**52.4±11.3**	3.79±1.10	**31%±9%**	**272.6±65.1**	**6.8±2.3**	**65%±12%**	**143.6±19.2**	**10.8±8.9**	**53%±37%**

ever, compared to other methods, AR-ICRL consistently exhibits fewer constraint violations across the environments regardless of the extent of the noise scales and types. This underscores the safety robustness of AR-ICRL and the effectiveness of utilizing the robust constraint for addressing the transition dynamics mismatch. Interestingly, we find BC2L and VICRL demonstrate relatively higher rewards in the Blocked Ant environment. This is due to their unique approach to predicting step-wise costs, which differs from the primarily MEICRL-based mechanisms employed by the other two methods. Nevertheless, despite their ability to attain higher rewards, it is essential to note that they can not guarantee safety, which is our primary concern.

5.3. Results Visualization

To better understand the effectiveness of AR-ICRL, we visualize the learned constraints and trajectories.

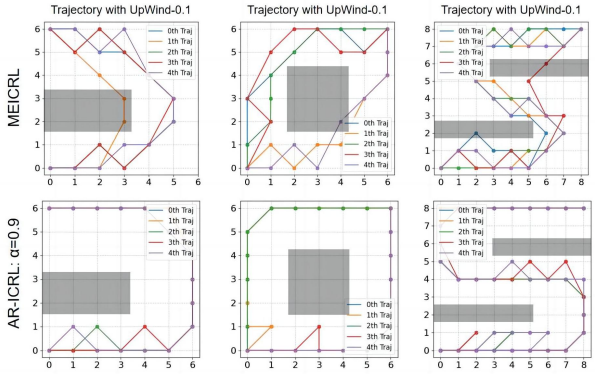

Figure 3. The trajectories generated by MEICRL and AR-ICRL in three Gridworld settings under the UpWind $p_w = 0.1$.

Trajectories in Gridworlds. We visualize the trajectories generated by MEICRL and AR-ICRL in three windy Gridworlds in Figure 3. Check Figure 7 in Appendix D.1 for trajectories without wind. We can find that by training with the safety opponent, AR-ICRL tends to consistently maintain a larger distance from the constrained locations. This strategy effectively reduces the likelihood of violating constraints when deployed to a perturbed environment.

Robust Constraint Visualization. Figure 4 illustrates the robust constraints in three Gridworlds recovered by AR-ICRL. We can find that compared to the original constraints (Figure 2), the recovered constraints offer greater robustness against underlying perturbations.

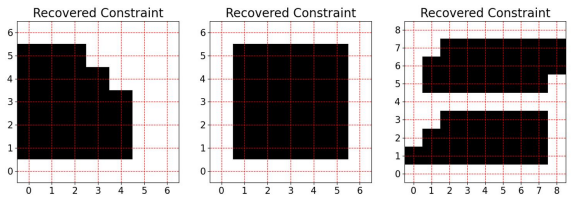

Figure 4. The robust constraints recovered by AR-ICRL under three Gridworld environments.

In terms of the continuous environments, we visualize the constraints inferred by MEICRL and AR-ICRL by sampling from the state-action spaces. Figure 5 shows the results in the Blocked Ant environment, where red curves are generated by synthetic sampling data and blue points denote the agent's recorded position with the corresponding predicted cost during testing. We find that the constraint recovered by AR-ICRL is generally more conservative than MEICRL, leading to safer policy strategies.

6. Limitation

Requirement for environment interaction. AR-ICRL operates in an online learning framework, where direct interaction with the environment is necessary for policy improvement and constraint inference. This approach may not be feasible in situations where interactions are costly or hazardous. Extending our model to offline settings could offer a safer and more practical alternative.

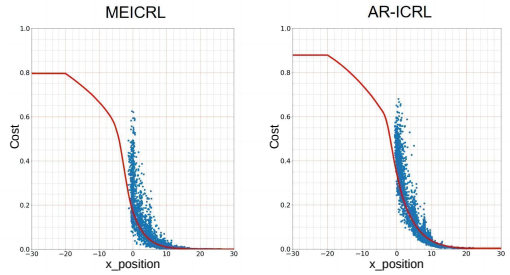

Figure 5. The recovered constraint in the Blocked Ant environment by MEICRL and AR-ICRL. Check Figure 8, 9, and 10 in Appendix D.2 for complete results.

Assumption of the uncertainty set. Our approach assumes that the discrepancy between training and deployment environments is bounded. This assumption may not always hold in real-world scenarios where environmental dynamics can be highly unpredictable and not easily represented within a predefined uncertainty set.

7. Conclusion

In this study, we highlight the critical importance of Inverse Constrained Reinforcement Learning (ICRL) algorithms in acknowledging and addressing the potential discrepancies between training and deployment environments. We introduce the Adaptively Robust Inverse Constrained Reinforcement Learning (AR-ICRL) approach as a pioneering solution to the Robust Constraint Inference problem, facilitating safe control. By alternatively solving the safety-robust policy optimization and adaptive constraint inference problem, AR-ICRL effectively learns a robust policy that adheres to inferred constraints and further deduces a robust constraint. Future efforts will focus on enhancing the algorithm's generalization capabilities, computational efficiency, and adaptability to real-world applications, aiming for broader applicability and improved safety.

Acknowledgements

This work is supported in part by the Shenzhen Fundamental Research Program (General Program) under grant JCYJ20230807114202005, Guangdong-Shenzhen Joint Research Fund under grant 2023A1515110617, GuangDong Basic and Applied Basic Research Foundation under grant 2024A1515012103, National Key R&D Program of China under grant No.2022ZD0116004, Shenzhen Science and Technology Program ZDSYS20211021111415025, and Guangdong Provincial Key Laboratory of Mathematical Foundations for Artificial Intelligence (2023B1212010001).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Agarwal, A., Jiang, N., Kakade, S. M., and Sun, W. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 32, 2019.

Baert, M., Mazzaglia, P., Leroux, S., and Simoens, P. Maximum causal entropy inverse constrained reinforcement learning. *arXiv preprint arXiv:2305.02857*, 2023.

Brunke, L., Greeff, M., Hall, A. W., Yuan, Z., Zhou, S., Panerati, J., and Schoellig, A. P. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5:411–444, 2022.

Gangwani, T. and Peng, J. State-only imitation with transition dynamics mismatch. In *International Conference on Learning Representations (ICLR)*, 2020.

García, J. and Fernández, F. A comprehensive survey on safe reinforcement learning. *Jounral of Machine Learning Research*, 16:1437–1480, 2015.

Gaurav, A., Rezaee, K., Liu, G., and Poupart, P. Learning soft constraints from constrained expert demonstrations. In *International Conference on Learning Representations (ICLR)*, 2023.

Gleave, A. and Toyer, S. A primer on maximum causal entropy inverse reinforcement learning. *arXiv preprint arXiv:2203.11409*, 2022.

Grau-Moya, J., Leibfried, F., Genewein, T., and Braun, D. A. Planning with information-processing constraints and model uncertainty in markov decision processes. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, pp. 475–491, 2016.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning (ICML)*, pp. 1352–1361, 2017.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, pp. 1856–1865, 2018.

Iyengar, G. N. Robust dynamic programming. *Mathematics of Operations Research*, 30:257–280, 2005.

Li, Z., Hu, C., Wang, Y., Yang, Y., and Li, S. E. Safe reinforcement learning with dual robustness. *arXiv preprint arXiv:2309.06835*, 2023.

Liu, G., Luo, Y., Gaurav, A., Rezaee, K., and Poupart, P. Benchmarking constraint inference in inverse reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2023.

Liu, S. and Zhu, M. Distributed inverse constrained reinforcement learning for multi-agent systems. In *Neural Information Processing Systems (NeurIPS)*, pp. 33444–33456, 2022.

Liu, S. and Zhu, M. Learning multi-agent behaviors from distributed and streaming demonstrations. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Liu, S. and Zhu, M. Meta inverse constrained reinforcement learning: Convergence guarantee and generalization analysis. In *International Conference on Learning Representations (ICLR)*, 2024.

Liu, Y., Halev, A., and Liu, X. Policy learning with constraints in model-free reinforcement learning: A survey. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 4508–4515, 2021.

Malik, S., Anwar, U., Aghasi, A., and Ahmed, A. Inverse constrained reinforcement learning. In *International Conference on Machine Learning (ICML)*, pp. 7390–7399, 2021.

Mankowitz, D. J., Calian, D. A., Jeong, R., Paduraru, C., Heess, N., Dathathri, S., Riedmiller, M., and Mann, T. Robust constrained reinforcement learning for continuous control with model misspecification. *arXiv preprint arXiv:2010.10644*, 2020a.

Mankowitz, D. J., Levine, N., Jeong, R., Abdolmaleki, A., Springenberg, J. T., Shi, Y., Kay, J., Hester, T., Mann, T. A., and Riedmiller, M. A. Robust reinforcement learning for continuous control with model misspecification. In *International Conference on Learning Representations (ICLR)*, 2020b.

McPherson, D. L., Stocking, K. C., and Sastry, S. S. Maximum likelihood constraint inference from stochastic demonstrations. In *IEEE Conference on Control Technology and Applications (CCTA)*, pp. 1208–1213, 2021.

Meng, J., Zhu, F., Ge, Y., and Zhao, P. Integrating safety constraints into adversarial training for robust deep reinforcement learning. *Information Sciences*, 619:310–323, 2023.

Moos, J., Hansel, K., Abdulsamad, H., Stark, S., Clever, D., and Peters, J. Robust reinforcement learning: A review of foundations and recent advances. *Machine Learning and Knowledge Extraction*, pp. 276–315, 2022.

Morimoto, J. and Doya, K. Robust reinforcement learning. *Neural Computation*, 17(2):335–359, 2005.

Narasimhan, H., Cotter, A., Zhou, Y., Wang, S., and Guo, W. Approximate heavily-constrained learning with lagrange multiplier models. In *Neural Information Processing Systems (NeurIPS)*, pp. 8693–8703, 2020.

Papadimitriou, D., Anwar, U., and Brown, D. S. Bayesian methods for constraint inference in reinforcement learning. *Transactions on Machine Learning Research*, 2023.

Perolat, J., Scherrer, B., Piot, B., and Pietquin, O. Approximate dynamic programming for two-player zero-sum markov games. In *International Conference on Machine Learning (ICML)*, pp. 1321–1329, 2015.

Qiao, G., Liu, G., Poupart, P., and zhiqiang xu. Multi-modal inverse constrained reinforcement learning from a mixture of demonstrations. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Roy, A., Xu, H., and Pokutta, S. Reinforcement learning under model mismatch. In *Neural Information Processing Systems (NeurIPS)*, pp. 3043–3052, 2017.

Russel, R. H., Benosman, M., and Van Baar, J. Robust constrained-mdps: Soft-constrained robust policy optimization under model uncertainty. *arXiv preprint arXiv:2010.04870*, 2020.

Schulman, J., Moritz, P., Levine, S., Jordan, M. I., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations (ICLR)*, 2016.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Scobee, D. R. R. and Sastry, S. S. Maximum likelihood constraint inference for inverse reinforcement learning. In *International Conference on Learning Representations, (ICLR)*, 2020.

Skalse, J. and Abate, A. Misspecification in inverse reinforcement learning. In *AAAI Conference on Artificial Intelligence*, pp. 15136–15143, 2023.

Tessler, C., Mankowitz, D. J., and Mannor, S. Reward constrained policy optimization. In *International Conference on Learning Representations (ICLR)*, 2019.

Viano, L., Huang, Y., Kamalaruban, P., Weller, A., and Cevher, V. Robust inverse reinforcement learning under transition dynamics mismatch. In *Neural Information Processing Systems (NeurIPS)*, pp. 25917–25931, 2021.

Viano, L., Huang, Y., Kamalaruban, P., Innes, C., Ramamoorthy, S., and Weller, A. Robust learning from observation with model misspecification. In *International Conference on Autonomous Agents and Multiagent Systems*, pp. 1337–1345, 2022.

Wang, Y. and Zou, S. Online robust reinforcement learning with model uncertainty. In *Neural Information Processing Systems (NeurIPS)*, pp. 7193–7206, 2021.

Wang, Y., Miao, F., and Zou, S. Robust constrained reinforcement learning. *arXiv preprint arXiv:2209.06866*, 2022.

Xu, S. and Liu, G. Uncertainty-aware constraint inference in inverse constrained reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2024.

Yu, D., Ma, H., Li, S., and Chen, J. Reachability constrained reinforcement learning. In *International Conference on Machine Learning (ICML)*, pp. 25636–25655, 2022.

Zhang, J., Zheng, Y., Zhang, C., Zhao, L., Song, L., Zhou, Y., and Bian, J. Robust situational reinforcement learning in face of context disturbances. In *International Conference on Machine Learning (ICML)*, pp. 41973–41989, 2023.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. Maximum entropy inverse reinforcement learning. In *AAAI Conference on Artificial Intelligence*, pp. 1433–1438, 2008.

Ziebart, B. D., Bagnell, J. A., and Dey, A. K. Modeling interaction via the principle of maximum causal entropy. In *International Conference on Machine Learning (ICML)*, pp. 1255–1262, 2010.

A. Practical Implementation

In this section, we introduce the practical implementation of our AR-ICRL algorithm.

A.1. Solving Markov Game in Discrete Environment

In the forward pass of our AR-ICRL algorithm (Section 4.1), optimizing the objective (8) involves updating the player and opponent policies together. This process can be viewed as solving a two-player Markov game in a CMDP. To solve this game in the discrete environment, by following Viano et al. (2021), we model the impact of movements from both players based on the joint cost Q function:

$$Q_c^{\text{two}}(s, a^{pl}, a^{op}) = \mathbb{E}_{\pi^{pl}, \pi^{op}, \mathrm{T}^{\text{two},\alpha}}\Big[-\sum_{t=0}^{H} \gamma^t \log \phi_{\omega,\alpha}(\cdot) \mid s_0 = s, a_0^{pl} = a^{pl}, a_0^{op} = a^{op}\Big] \tag{16}$$

where 1) $\phi_{\omega,\alpha}(s_t, a_t^{pl}, a_t^{op}) = \alpha\phi_\omega(s_t, a_t^{pl}) + (1-\alpha)\phi_\omega(s_t, a_t^{op})$ is the cost at time step t (we use $\phi_{\omega,\alpha,t}(\cdot)$ for shorthand), 2) a^{pl} and a^{op} are the actions of the player and opponent, and 3) $\mathrm{T}^{\text{two},\alpha}\left(s' \mid s, a^{pl}, a^{\text{op}}\right) = \alpha \mathrm{T}\left(s' \mid s, a^{pl}\right) + (1-\alpha)\mathrm{T}\left(s' \mid s, a^{op}\right)$ denotes the transition dynamics of two players. We further define the worst-case Q function as:

$$Q_c^{pl}(s, a^{pl}, a^{op}) = \max_{\pi^{op}} Q_c^{\text{two}}(s, a^{pl}, a^{op}) \tag{17}$$

Note that Q_c^{pl} indicates the ability of the player π^{pl} to minimize the cost (ensure safety) under the worst-case safety attacker in the long term starting from (s, a^{pl}, a^{op}), which can be estimated by the following Bellman equation (Perolat et al., 2015):

$$Q_c^{pl}(s_t, a_t^{pl}, a_t^{op}) = \sum_{s_{t+1}} \mathrm{T}^{\text{two},\alpha}(s_{t+1} \mid s_t, a_t^{pl}, a_t^{op}) \cdot \Big[-\log \phi_{\omega,\alpha,t}(\cdot) + \gamma V_c^{pl}(s_{t+1})\Big] \tag{18}$$

where $V_c^{pl}(s_t) = \sum_{a_t^{pl}} \pi^{pl}(a_t^{pl}|s_t) \max_{a_t^{op}} Q_c^{pl}(s_t, a_t^{pl}, a_t^{op})$.

To better capture the player policy π^{pl} that is robust under attacks, we define the following robust sets (Li et al., 2023):

Definition A.1. (Robust state set) The robust state set for the player policy π^{pl} under the worst-case safety attacks is defined as:

$$S_\forall^{\pi^{pl}} = \Big\{ s \in \mathcal{S} \mid \min_{a^{pl}} \max_{a^{op}} Q_c^{pl}(s, a^{pl}, a^{op}) \le \epsilon \Big\} \tag{19}$$

Note that for $s \in S_\forall^{\pi^{pl}}$, the player policy π^{pl} can be safe in states regardless of any opponent's attacks. For $s \notin S_\forall^{\pi^{pl}}$, there exists some corresponding opponent policy π^{op} that will drive the agent to violate the constraint.

Definition A.2. (Robust action set) The robust action set $\mathcal{A}_\forall^{\pi^{pl}}$ for the player policy π^{pl} specified by the robust state set $S_\forall^{\pi^{pl}}$ is defined as:

$$\mathcal{A}_\forall^{\pi^{pl}} = \Big\{ a^{pl} \mid \forall s \in S_\forall^{\pi^{pl}}, \max_{a^{op}} Q_c^{pl}(s, a^{pl}, a^{op}) \le \epsilon \Big\} \tag{20}$$

Proposition A.3. *Given a state $s \in S_\forall^{\pi^{pl}}$, if the player always adopts the action in the robust action set (i.e., $a^{pl} \in \mathcal{A}_\forall^{\pi^{pl}}$), the system will consistently stay in $S_\forall^{\pi^{pl}}$ no matter what opponent policy π^{op} is adopted.*

The proof is shown in Appendix B.6. Based on this proposition, we divide $s \in \mathcal{S}$ into two sets:

1) if $s \in S_\forall^{\pi^{pl}}$, we optimize the player π^{pl} to maximize rewards within the robust action set $\mathcal{A}_\forall^{\pi^{pl}}$:

$$\pi^{pl}(a^{pl}|s) = \frac{\mathbb{1}_{a^{pl} \in \mathcal{A}_\forall^{\pi^{pl}}} \exp Q_r^{pl}(s, a^{pl})}{\sum_{a'} \exp Q_r^{pl}(s, a')} \tag{21}$$

where the reward Q function is defined by:

$$Q_r^{pl}(s, a^{pl}) = \mathbb{E}_{\pi^{pl}, \mathrm{T}}\Big[\sum_{t=0}^{H} \gamma^t r(s_t, a_t) + \beta\mathcal{H}[\pi^{pl}(\cdot|s_t)] | s_0 = s, a_0 = a^{pl}\Big] \tag{22}$$

which can be updated by the soft Bellman equation:

$$Q_r^{pl}(s_t, a_t^{pl}) = \sum_{s_{t+1}, r_t} \mathrm{T}(s_{t+1} \mid s_t, a_t^{pl}) \Big[r_t + \gamma V_r^{pl}(s_{t+1}) \Big] \tag{23}$$

where $V_r^{pl}(s_t) = \sum_{a_t^{pl}} \pi^{pl}(a_t^{pl}|s_t) \Big[Q_r^{pl}(s_t, a_t^{pl}) + \beta \mathcal{H}[\pi^{pl}(\cdot|s_t)] \Big]$.

2) if $s \notin S_\forall^{\pi^{pl}}$, which means that the constraint will be violated sooner or later under the worst-case safety opponent, the player should be updated to execute the safest action to drive the agent back to $S_\forall^{\pi^{pl}}$ as soon as possible (minimize costs).

Since we do not have a robust set from the beginning, following Li et al. (2023), we train a safety robust player policy π_c^{pl} in advance by using the cost Q function and then fix it, which aims to minimize the cost values. On the other hand, the target policy π^{pl} is designed to maximize rewards within the robust state set $S_\forall^{\pi_c^{pl}}$ determined by the fixed safety policy π_c^{pl}, and strive for the minimize cost value if outside of the set. Under such designs, our algorithm can solve the constrained optimization problem without the Lagrange multiplier, which can better ensure safety without violations. The procedure of the proposed safety-robust policy iteration algorithm is shown in Algorithm 2.

Algorithm 2 Safety-Robust Policy Iteration

Input: Target player π^{pl}, safety player π_c^{pl} and opponent π_c^{op}, reward Q-function $Q_r^{pl}(s,a)$, cost Q-function $Q_c^{pl}(s,a,o)$, cost function c_ω, opponent strength $1-\alpha$.
while not converged **do**
  while not converged **do**
    Solve the cost Q function by Equation (18).
    for $s \in \mathcal{S}$ **do**
      $\pi_c^{pl}(a|s) = \arg\min_a \{\max_o Q_c^{pl}(s,a,o)\}$.
      $\pi_c^{op}(o|s) = \arg\max_o \{\min_a Q_c^{pl}(s,a,o)\}$.
    end for
  end while
  Solve the reward Q function by Equation (23).
  for $s \in S_\forall^{\pi_c^{pl}}$ **do**
    $\pi^{pl}(a^{pl}|s) = \frac{\mathbb{1}_{a^{pl} \in \mathcal{A}_\forall^{\pi^{pl}}} \exp Q_r^{pl}(s,a^{pl})}{\sum_{a'} \exp Q_r^{pl}(s,a')}$
  end for
  for $s \notin S_\forall^{\pi_c^{pl}}$ **do**
    $\pi^{pl}(s) = \pi_c^{pl}(s)$.
  end for
end while

A.2. Two Players' Game in Continuous Environment

In this section, we solve the above Markov Game to the continuous state-action space under a CMDP. Instead of applying the value-based policy representation (e.g., $\arg\max Q$-function), we parameters both player's and opponent's policies (e.g., construct π_θ^{pl} and π_ψ^{op}). We further define the joint value function for costs as $V_c(s) = \mathbb{E}_{\pi_\theta^{pl}, \pi_\psi^{op}}[Q_c(s, a^{pl}, a^{op})]$ and similarity for the joint value function for rewards V_r.

During learning, given a state s_t, the agent executes the mixture policy $\pi_{\theta,\psi}^{mix}(a|s) = \alpha\pi_\theta^{pl}(a|s) + (1-\alpha)\pi_\psi^{op}(a|s)$, , transits to the next state s_{t+1}, and received the reward r_t and the cost c_t accordingly. Based on these observations, we extend Proximal Policy Optimization (PPO) (Schulman et al., 2017) to optimize both the player policy π_θ^{pl} and the opponent policy π_ψ^{op} with the following objectives:

$$\min_\zeta \max_\theta \mathbb{E}_{\pi_{\theta,\psi}^{mix}, \mathrm{T}, \mu_0} \left[L_r^{clip}(s,a) - \lambda_\zeta(s) L_c^{clip}(s,a) \right] \tag{24}$$

$$\max_\psi \mathbb{E}_{\pi_{\theta,\psi}^{mix}, \mathrm{T}, \mu_0} [L_c^{clip}(s,a)] \tag{25}$$

where 1) $\lambda_\zeta(s) \geq 0$ denotes the Lagrange multiplier network parameterized with ζ (Narasimhan et al., 2020), and 2) $L_r^{clip}(s,a) = \min\left[\frac{\pi^{mix}_{\theta,\theta'}(a|s)}{\pi^{mix}_{old}(a|s)}A_r(s,a), \text{clip}(\frac{\pi^{mix}_{\theta,\theta'}(a|s)}{\pi^{mix}_{old}(a|s)}, 1-\omega, 1+\omega)A_r(s,a)\right]$, where the reward advantage is computed with an entropy term (Baert et al., 2023). Its cost counterpart $L_c^{clip}(s,a)$ is based on cost advantages A_c computed with the Generative Advantage Estimation (GAE) (Schulman et al., 2016).

Under this formulation, intuitively, if $s \notin S_\forall^{\pi^{pl}}$ (i.e., the robust policy set), the safety can not be ensured, and we have its cost advantage $A_c(s,a) > 0$. Therefore, the Lagrange multiplier $\lambda_\zeta(s)$ will be significantly amplified until the cost penalty (i.e., $\lambda_\zeta(s)L_c^{clip}$) dominates objective and forces the agent to adopt the safest action and move back to the robust state set. Practically, we set an upper bound λ_{max} to avoid hazardous situations. The proposed Safety-Robust Proximal Policy Optimization algorithm is summarized in Algorithm 3. We also implement a conventional PPO-Lagrangian-based (check Algorithm 1 in (Liu et al., 2023)) version of the proposed robust policy optimization, where the Lagrange multiplier λ is updated by dual ascent instead of objective (24).

Algorithm 3 Safety-Robust Proximal Policy Optimization

Input: Player policy π_θ^{pl} and opponent policy $\pi_{\theta'}^{op}$, joint reward value critic V^r, joint cost value critic V^c, cost function c_ω, opponent strength $1-\alpha$, Lagrange multiplier λ, rollout rounds B, update rounds K.
Initialize state s_0 from CMDP and the roll-out buffer $\mathcal{D}$.
for $b = 1, 2, \ldots, B$ **do**
  Perform policies π_θ^{pl} and $\pi_{\theta'}^{op}$, and collect trajectories $\tau_b = [s_0, a_0^{pl}, a_0^{op}, r_0, c_0, \ldots, s_H, a_H^{pl}, a_H^{op}, r_H, c_H]$.
  Calculate reward advantages A_t^r, reward returns R_t, cost advantages A_t^c, and cost returns C_t via GAE (Schulman et al., 2016) from the trajectory.
  Add transition samples to the buffer $\mathcal{D} = \mathcal{D} \cup \{s_t, a_t^{pl}, a_t^{op}, r_t, A_t^r, R_t, c_t, A_t^c, C_t\}_{t=0}^H$.
end for
for $k = 1, 2, \ldots, K$ **do**
  Sample a batch of transition data from $\mathcal{D}$.
  Update the reward and cost value functions by minimizing the mean-square loss with returns.
  Update opponent policy parameters θ' by (25).
  Update player policy parameters θ by (24).
  Update Lagrange multiplier parameters ζ by (24).
end for

B. Proof

Lemma B.1. *If* $Q^{soft}(s,a) = \mathbb{E}\left[\sum_{t=0}^H [\bar{r}(s_t,a_t) + \mathcal{H}(\pi(a_t|s_t))] \,|s_0 = s, a_0 = a\right]$ *defines the soft-Q function (Haarnoja et al., 2017). It holds that* $\mathbb{E}_\mathcal{D}\left[\sum_{t=0}^H \gamma^t \left(Q^{soft}(s_t,a_t) - V^{soft}(s_t)\right)\right] = \mathbb{E}_\mathcal{D}\left[\sum_{t=0}^H \gamma^t \bar{r}(s_t,a_t)\right] - V_{M_c}^{soft}(s_0)$, *where* $\mathcal{D}$ *is the demonstration dataset.*

Proof.

$$
\begin{aligned}
&\mathbb{E}_\mathcal{D}\left[\sum_{t=0}^H \gamma^t \left(Q^{soft}(s_t,a_t) - V^{soft}(s_t)\right)\right] \\
&= \mathbb{E}_\mathcal{D}\Big[\sum_{t=0}^{H-1} \gamma^t \Big(\bar{r}(s_t,a_t) + \gamma \mathbb{E}_{\mathrm{T}(\cdot|s_t,a_t)}[V^{soft}(s_{t+1})] - V^{soft}(s_t)\Big) + \gamma^H \bar{r}(s_H,a_H)\Big] \\
&= \mathbb{E}_\mathcal{D}\left[\sum_{t=0}^H \gamma^t \bar{r}(s_t,a_t)\right] + \mathbb{E}_\mathcal{D}\left[\sum_{t=0}^{H-1} \gamma^{t+1} \mathbb{E}_{\mathrm{T}}[V_{M_c}^{soft}(s_{t+1})] - \sum_{t=0}^H \gamma^t V_{M_c}^{soft}(s_t)\right] \\
&= \mathbb{E}_\mathcal{D}\left[\sum_{t=0}^H \gamma^t \bar{r}(s_t,a_t)\right] - V_{M_c}^{soft}(s_0)
\end{aligned}
$$

□

Lemma B.2. *Let* $V^{soft}(s) = \mathbb{E}\left[\sum_{t=0}^H [\bar{r}_\omega(s_t,a_t) + \mathcal{H}(\pi(a_t|s_t))] \,|s_0 = s\right]$ *define the soft-value function (Haarnoja*

et al., 2017). The reward function $\bar{r}_\omega$ is parametered by ω. The gradient of $\nabla_\omega V^{soft}(s_t)$ can be represented as $\mathbb{E}_\pi\left[\sum_{\iota=0}^{H-t}\gamma^\iota \nabla_\omega \bar{r}_\omega(s_\iota, a_\iota)|s_0 = s_t\right]$ where π represents the soft-optimal (soft-max) policy.

Proof.

$$\begin{aligned}
\nabla_\omega V^{soft}(s_t) =& \nabla_\omega \log \sum_{a_t\in\mathcal{A}} \exp Q^{soft}(s_t, a_t) \\
=& \frac{\sum_{a_t\in\mathcal{A}} \nabla_\omega \exp Q^{soft}(s_t, a_t)}{\sum_{a_t\in\mathcal{A}} \exp Q^{soft}(s_t, a_t)} \\
=& \frac{\sum_{a_t\in\mathcal{A}} \left(\exp Q^{soft}(s_t, a_t)\cdot \nabla_\omega Q^{soft}(s_t, a_t)\right)}{\exp V^{soft}(s_t)} \\
=& \sum_{a_t\in\mathcal{A}} \exp\left(Q^{soft}(s_t, a_t) - V^{soft}(s_t)\right)\cdot \nabla_\omega Q^{soft}(s_t, a_t) \\
=& \sum_{a_t\in\mathcal{A}} \pi(a_t|s_t)\cdot \nabla_\omega Q^{soft}(s_t, a_t) \\
=& \mathbb{E}_\pi\left[\nabla_\omega \bar{r}_\omega(s_t, a_t) + \gamma \mathbb{E}_{\mathrm{T}(\cdot|s_t,a_t)}\left[\nabla_\omega V^{soft}(s_{t+1})\right]\right] \\
=& \mathbb{E}_\pi\left[\sum_{\iota=0}^{H-t}\gamma^\iota \nabla_\omega \bar{r}_\omega(s_\iota, a_\iota) \mid s_0 = s_t\right]
\end{aligned}$$

□

B.1. Proof of Proposition 3.3

Proof. **(a)** To prove the existence of such π, T and T′, we assume there exists a policy π and a transition T satisfying $\mathbb{E}_{\mu_0,\pi}[Q^{c,\pi}_{M_c^{\mathrm{T}}}(s,a)] = \epsilon$. In this way, we only need to show $\exists \mathrm{T}', \pi$ s.t. $\mathbb{E}_{\mu_0,\pi}[Q^{c,\pi}_{M_c^{\mathrm{T}'}}(s,a)] - \mathbb{E}_{\mu_0,\pi}[Q^{c,\pi}_{M_c^{\mathrm{T}}}(s,a)] > 0$.

Consider the simulation lemma (Agarwal et al., 2019):

$$Q^{c,\pi}_{M_c^{\mathrm{T}'}} - Q^{c,\pi}_{M_c^{\mathrm{T}}} = \gamma(I - \gamma\pi\mathrm{T}')^{-1}(\mathrm{T}' - \mathrm{T})V^{c,\pi}_{M_c^{\mathrm{T}}}$$

We know that $\forall(s,a)$, $(I-\gamma\pi\mathrm{T}')^{-1}_{(s,a),(s',a')} > 0$[1], and $V^{c,\pi}_{M_c^{\mathrm{T}}}(s) \geq 0$ (since $c(s,a)\geq 0$). Define the safe states set $\mathcal{S}_s = \{s\in\mathcal{S} \mid V^{c,\pi}_{M_c^{\mathrm{T}}}(s) \leq \epsilon\}$ and unsafe states set $\mathcal{S}_u = \{s\in\mathcal{S} \mid V^{c,\pi}_{M_c^{\mathrm{T}}}(s) > \epsilon\}$.

Consider such a T′: 1) $\forall s'\in\mathcal{S}_u$, $\mathrm{T}'_{(s,a),(s')} = \mathrm{T}_{(s,a),(s')} + \alpha$; 2) $\forall s'\in\mathcal{S}_s$, $\mathrm{T}'_{(s,a),(s')} = \mathrm{T}_{(s,a),(s')} - \beta$, where $0\leq\alpha\leq 1$, $0\leq\beta\leq 1$, and $\alpha\cdot|\mathcal{S}_s| = \beta\cdot|\mathcal{S}_u|$. This T′ can ensure that $\forall(s,a), [(\mathrm{T}'-\mathrm{T})V^{c,\pi}_{M_c^{\mathrm{T}}}](s,a)\geq 0$, and thus $(Q^{c,\pi}_{M_c^{\mathrm{T}'}} - Q^{c,\pi}_{M_c^{\mathrm{T}}})(s,a)\geq 0$. The equality holds only when $\forall s'\sim\mathrm{T}(\cdot|s,a)\in\mathcal{S}_s$ (i.e., $\forall s'\in\mathcal{S}_u, \mathrm{T}(s'|s,a) = 0$). In this case, we modify the T′ such that: 1) $\forall s'\in\mathcal{S}_u, \mathrm{T}'(s'|s,a) = \frac{1}{|\mathcal{S}_u|}$; 2) $\forall s'\in\mathcal{S}_s, \mathrm{T}'(s'|s,a) = 0$, which ensures strict inequality.

(b) Consider the simulation lemma (Agarwal et al., 2019):

$$\begin{aligned}
&\mathbb{E}_{\mu_0,\pi^*}[Q^{c,\pi^*}_{M_c^{\mathrm{T}'}}(s,a)] - \mathbb{E}_{\mu_0,\pi^*}[Q^{c,\pi^*}_{M_c^{\mathrm{T}}}(s,a)] \\
&= \rho^{\pi^*}_{\mu_0}\gamma(I-\gamma\pi^*\mathrm{T})^{-1}(\mathrm{T}'-\mathrm{T})V^{c,\pi^*}_{M_c^{\mathrm{T}'}} \\
&= \frac{\gamma}{1-\gamma} d^{\pi^*}_{M_c^{\mathrm{T}}}(\mathrm{T}'-\mathrm{T})V^{c,\pi^*}_{M_c^{\mathrm{T}'}} \\
&> \epsilon
\end{aligned}$$

Since $\mathbb{E}_{\mu_0,\pi^*}[Q^{c,\pi^*}_{M_c^{\mathrm{T}}}(s,a)]\geq 0$, we have $\mathbb{E}_{\mu_0,\pi^*}[Q^{c,\pi^*}_{M_c^{\mathrm{T}'}}(s,a)] > \epsilon$ □

[1] $M_{(x),(y)}$ indicates the element of x^{th} row and y^{th} column

B.2. Proof of Proposition 4.1

For brevity, we define $\mathbb{F} = \sum_{t=0}^{H} \gamma^t \log \phi(s_t, a_t)$ as the safety (feasibility) objective, we can show that:

$$\begin{aligned}
\min_{\bar{M} \in \mathcal{M}^{\mathcal{T}_\alpha^L}} \mathbb{E}\Big[\lambda\mathbb{F} \mid \pi, \bar{M}\Big] &= \min_{\bar{\mathrm{T}} \in \mathcal{T}_\alpha^L} \mathbb{E}\Big[\lambda\mathbb{F} \mid \pi, \mu_0, \bar{\mathrm{T}}\Big] \\
&\overset{(a)}{\leq} \min_{\tilde{\mathrm{T}} \in \widetilde{\mathcal{T}}_\alpha^L} \mathbb{E}\Big[\lambda\mathbb{F} \mid \pi, \mu_0, \tilde{\mathrm{T}}\Big] \\
&\overset{(b)}{=} \min_{\pi^{op} \in \Pi} \mathbb{E}\Big[\lambda\mathbb{F} \mid \alpha\pi + (1-\alpha)\pi^{op}, \mu_0, \mathrm{T}^L\Big]
\end{aligned} \tag{26}$$

where $\mathcal{T}_\alpha^L(s' \mid s, a) = \{\alpha \mathrm{T}^L(s' \mid s, a) + (1-\alpha)\bar{\mathrm{T}}(s' \mid s, a), \forall \bar{\mathrm{T}} \in \mathcal{T}\}$, and $\widetilde{\mathcal{T}}_\alpha^L(s' \mid s, a) = \{\alpha \mathrm{T}^L(s' \mid s, a) + (1-\alpha)\sum_a \pi(a|s)\bar{\mathrm{T}}(s' \mid s, a), \forall \bar{\mathrm{T}} \in \mathcal{T}, \forall \pi \in \Pi\}$. In specific, (a) hold since $\widetilde{\mathcal{T}}_\alpha^L \subset \mathcal{T}_\alpha^L$, (b) hold due to Section 3.1 in (Tessler et al., 2019).

B.3. Proof of Proposition 4.4

Proof. The log-likelihood of generating the expert demonstration $\mathcal{D}_e$ is defined as:

$$\begin{aligned}
\mathbb{E}_{\mathcal{D}_e}\left[\log p_{M_c^{\mathrm{T}}}^{\pi^{mix}}(\tau)\right] &= \left[\sum_{t=0}^{H} \gamma^t \log \pi_{M_c^{\mathrm{T}}}^{mix}(a_t|s_t) + \log \mu_0(s_0) + \sum_{t=0}^{H} \log \mathrm{T}(s_{t+1}|s_t, a_t)\right] \\
&\overset{(1)}{=} \mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H} \gamma^t \log[\alpha \pi_{M_c^{\mathrm{T}}}^{pl}(a_t|s_t) + (1-\alpha)\pi_{M_c^{\mathrm{T}}}^{op}(a_t|s_t)] + \log \mu_0(s_0) + \sum_{t=0}^{H} \log \mathrm{T}(s_{t+1}|s_t, a_t)\right] \\
&\overset{(2)}{=} \mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H} \gamma^t [\log \alpha + \log \pi_{M_c^{\mathrm{T}}}^{pl}(a_t|s_t)] + \log \mu_0(s_0) + \sum_{t=0}^{H} \log \mathrm{T}(s_{t+1}|s_t, a_t)\right] \\
&\overset{(3)}{=} \mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H} \gamma^t \left(Q_{M_c^{\mathrm{T}}}^{soft,pl}(s_t, a_t) - V_{M_c^{\mathrm{T}}}^{soft,pl}(s_t)\right)\right] + \mathbb{C} \\
&\overset{(4)}{=} \mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H} \gamma^t \left[r(s_t, a_t) + \lambda \log \phi(s_t, a_t)\right]\right] - V_{M_c^{\mathrm{T}}}^{soft,pl}(s_0) + \mathbb{C}
\end{aligned}$$

In this way, we can further define:

(1) holds due to the definition of $\pi_{M_c^{\mathrm{T}}}^{mix}(a_t|s_t) = \alpha\pi_{M_c^{\mathrm{T}}}^{pl}(a_t|s_t) + (1-\alpha)\pi_{M_c^{\mathrm{T}}}^{op}(a_t|s_t)$.

(2) holds since the cost maximization policy $\pi_{M_c^{\mathrm{T}}}^{op}(a|s) = \arg\max Q_{M_c^{\mathrm{T}}}^{c,*}(\tilde{s}, a)$ is deterministic, and based on Assumption 4.3, we have that $\forall(s, a)$, if $\rho_{M_c^{\mathrm{T}}}^{\pi^e}(s, a) > 0$, then $\pi_{M_c^{\mathrm{T}}}^{op}(a|s) = 0$.

(3) holds since $\pi_{M_c^{\mathrm{T}}}^{pl}(a_t|s_t) = \exp\left(Q_{M_c^{\mathrm{T}}}^{soft,pl}(s_t, a_t) - V_{M_c^{\mathrm{T}}}^{soft,pl}(s_t)\right)$, and the constant $\mathbb{C}$ is irrelevant to costs.

(4) holds due to Lemma B.1, where $\bar{r}(\cdot) = r(\cdot) + \lambda \log \phi(\cdot)$.

Our constraint function $\phi_\omega(s, a)$ is parameters by ω. We infer the constraint function by maximizing the gradient of the log-likelihood of generating the demonstration data: $\omega = \arg\max_\omega \mathbb{E}_{\mathcal{D}_e}\left[\log p_{M_c^{\mathrm{T}}}(\tau)\right]$.

$$\begin{aligned}
\nabla_\omega \mathbb{E}_{\mathcal{D}_e}\left[\log p_{M_c^{\mathrm{T}}}(\tau)\right] &= \mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H} \gamma^t \lambda \nabla_\omega \log \phi_\omega(s_t, a_t)\right] + \nabla_\omega V_{M_c}^{soft,pl}(s_0) \\
&\overset{(1)}{=} \mathbb{E}_{(s,a)\sim\mathcal{D}_e}\left[\sum_{t=0}^{H} \gamma^t \lambda \nabla_\omega \log \phi_\omega(s_t, a_t)\right] + \mathbb{E}_{(s,a)\sim\pi_{M_c^{\mathrm{T}}}^{pl}}\left[\sum_{t=0}^{H} \gamma^t \nabla_\omega \lambda \log \phi_\omega(s_t, a_t)\right]
\end{aligned}$$

(1) holds due to Lemma B.2, where $\bar{r}_\omega(\cdot) = r(\cdot) + \lambda \log \phi_\omega(\cdot)$.

□

B.4. Proof of Proposition 4.5

Proof. The log-likelihood of generating the expert demonstration $\mathcal{D}_e$ is defined as:

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{D}_e}\left[\log p^{\pi^{mix}}_{M_c^{\mathrm{T}}}(\tau)\right] \\
=&\left[\sum_{t=0}^{H}\gamma^t \log \pi^{mix}_{M_c^{\mathrm{T}}}(a_t|s_t) + \log\mu_0(s_0) + \sum_{t=0}^{H}\log \mathrm{T}(s_{t+1}|s_t,a_t)\right] \\
\overset{(1)}{=}&\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t \log[\alpha\pi^{pl}_{M_c^{\mathrm{T}}}(a_t|s_t) + (1-\alpha)\pi^{op}_{M_c^{\mathrm{T}}}(a_t|s_t)] + \log\mu_0(s_0) + \sum_{t=0}^{H}\log \mathrm{T}(s_{t+1}|s_t,a_t)\right] \\
\overset{(2)}{\geq}&\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t \log[(\pi^{pl}_{M_c^{\mathrm{T}}}(a_t|s_t))^{\alpha}\cdot(\pi^{op}_{M_c^{\mathrm{T}}}(a_t|s_t))^{1-\alpha}] + \log\mu_0(s_0) + \sum_{t=0}^{H}\log \mathrm{T}(s_{t+1}|s_t,a_t)\right] \\
\overset{(3)}{=}&\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t\alpha\left(Q^{soft,pl}_{M_c^{\mathrm{T}}}(s_t,a_t) - V^{soft,pl}_{M_c^{\mathrm{T}}}(s_t)\right)\right] + \mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t(1-\alpha)\left(Q^{soft,op}_{M_c^{\mathrm{T}}}(s_t,a_t) - V^{soft,op}_{M_c^{\mathrm{T}}}(s_t)\right)\right] + \mathbb{C} \\
\overset{(4)}{=}&\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t\alpha\left[r(s_t,a_t) + \lambda\log\phi(s_t,a_t)\right]\right] - \alpha V^{soft,pl}_{M_c^{\mathrm{T}}}(s_0) - \\
&(1-\alpha)\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t\left[\log\phi(s_t,a_t)\right]\right] - (1-\alpha)V^{soft,op}_{M_c^{\mathrm{T}}}(s_0) + \mathbb{C} \\
=&\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t\left[\alpha r(s_t,a_t) + (\alpha\lambda+\alpha-1)\log\phi(s_t,a_t)\right]\right] - \alpha V^{soft,pl}_{M_c^{\mathrm{T}}}(s_0) - (1-\alpha)V^{soft,op}_{M_c^{\mathrm{T}}}(s_0) + \mathbb{C} \\
=&\mathcal{L}(\mathcal{D}_e, M_c^{\mathrm{T}}, \alpha)
\end{aligned}
$$

(1) holds due to the definition of $\pi^{mix}_{M_c^{\mathrm{T}}}(a_t|s_t) = \alpha\pi^{pl}_{M_c^{\mathrm{T}}}(a_t|s_t) + (1-\alpha)\pi^{op}_{M_c^{\mathrm{T}}}(a_t|s_t)$.

(2) holds due to the Weighted AM-GM Inequality:

$$(w_1a_1 + w_2a_2 + \cdots + w_na_n) \geq a_1^{w_1}\cdot a_2^{w_2}\cdots a_n^{w_n}$$

if $w_1 + w_2 + \cdots + w_n = 1$ and $a_1 + a_2 + \cdots + a_n$ are not negative.

(3) holds since both $\pi^{pl}_{M_c^{\mathrm{T}}}$ and $\pi^{op}_{M_c^{\mathrm{T}}}$ are soft-optimal, i.e., $\pi^{pl}_{M_c^{\mathrm{T}}}(a_t|s_t) = \exp\left(Q^{soft,pl}_{M_c^{\mathrm{T}}}(s_t,a_t) - V^{soft,pl}_{M_c^{\mathrm{T}}}(s_t)\right)$, $\pi^{op}_{M_c^{\mathrm{T}}}(a_t|s_t) = \exp\left(Q^{soft,op}_{M_c^{\mathrm{T}}}(s_t,a_t) - V^{soft,op}_{M_c^{\mathrm{T}}}(s_t)\right)$, and the constant $\mathbb{C}$ is irrelevant to costs.

(4) holds due to Lemma B.1. In addition, according to the definition of $Q^{soft,pl}_{M_c^{\mathrm{T}}}$, we have $\bar{r}(\cdot) = r(\cdot) + \lambda\log\phi(\cdot)$. According to the definition of $Q^{soft,op}_{M_c^{\mathrm{T}}}$, we have $\bar{r}(\cdot) = -\log\phi(\cdot)$.

Our constraint function $\phi_\omega(s,a)$ is parameters by ω. We infer the constraint function by maximizing the gradient of the log-likelihood of generating the demonstration data: $\omega = \arg\max_\omega \mathcal{L}(\mathcal{D}_e, M_c^{\mathrm{T}}, \alpha)$.

$$
\begin{aligned}
\nabla_\omega\mathcal{L}(\mathcal{D}_e, M_c^{\mathrm{T}}, \alpha) =&\mathbb{E}_{\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t(\alpha\lambda+\alpha-1)\nabla_\omega\log\phi_\omega(s_t,a_t)\right] - \alpha\nabla_\omega\alpha V^{soft,pl}_{M_c^{\mathrm{T}}}(s_0) - (1-\alpha)\nabla_\omega V^{soft,op}_{M_c^{\mathrm{T}}}(s_0) \\
\overset{(1)}{=}&\mathbb{E}_{(s,a)\sim\mathcal{D}_e}\left[\sum_{t=0}^{H}\gamma^t(\alpha\lambda+\alpha-1)\nabla_\omega\log\phi_\omega(s_t,a_t)\right] - \alpha\lambda\mathbb{E}_{(s,a)\sim\pi^{pl}_{M_c^{\mathrm{T}}}}\left[\sum_{t=0}^{H}\gamma^t\nabla_\omega\log\phi_\omega(s_t,a_t)\right] + \\
&(1-\alpha)\mathbb{E}_{(s,a)\sim\pi^{op}_{M_c^{\mathrm{T}}}}\left[\sum_{t=0}^{H}\gamma^t\nabla_\omega\log\phi_\omega(s_t,a_t)\right]
\end{aligned}
$$

(1) holds due to Lemma B.2. In addition, according to the definition of $V^{soft,pl}_{M_c^{\mathrm{T}}}$, we have $\bar{r}_\omega(\cdot) = r(\cdot) + \lambda \log \phi_\omega(\cdot)$. According to the definition of $V^{soft,op}_{M_c^{\mathrm{T}}}$, we have $\bar{r}_\omega = -\log \phi_\omega(\cdot)$.

□

B.5. Proof of Proposition 4.6

Proof. For brevity, let $\pi^L := \pi^{me}_{\tilde{c}}$, $\pi^{pl} := \pi^{ro}_{\hat{c}}$.

To prove **(a)**, we show the existence of such a policy π^L as follows:

$$\begin{aligned}
\mathbb{E}_{\mu_0}[V^{c,\pi^L}_{M^{\mathrm{T}'}_{c^*}}(s)] - \mathbb{E}_{\mu_0}[V^{c,\pi^L}_{M^{\mathrm{T}^L}_{c^*}}(s)] &= \mathbb{E}_{\mu_0,\pi^L}[Q^{c,\pi^L}_{M^{\mathrm{T}'}_{c^*}}(s,a)] - \mathbb{E}_{\mu_0,\pi^L}[Q^{c,\pi^L}_{M^{\mathrm{T}^L}_{c^*}}(s,a)] \\
&\overset{(1)}{=} \rho^{\pi^L}_{\mu_0}\gamma(I-\gamma\pi^L\mathrm{T}^L)^{-1}(\mathrm{T}'-\mathrm{T}^L)V^{c,\pi^L}_{M^{\mathrm{T}'}_{c^*}} \\
&\overset{(2)}{=} \frac{\gamma}{1-\gamma} d^{\pi^L}_{M^{\mathrm{T}^L}_{c^*}}(\mathrm{T}'-\mathrm{T}^L)V^{c,\pi^L}_{M^{\mathrm{T}'}_{c^*}} \\
&\geq 0
\end{aligned}$$

(1) applies the simulation lemma (Agarwal et al., 2019).

(2) holds due to that $d^{\pi^L}_{M^{\mathrm{T}^L}_{c^*}} \geq 0$ defines the normalized occupancy measure by following policy π^L in the CMDP $M^{\mathrm{T}^L}_{c^*}$.

We know that there exists a $\mathrm{T}' \in \{\mathrm{T} : \max_{s,a} |\mathrm{T} - \mathrm{T}^L| \leq 2(1-\alpha)\}$ that makes $(\mathrm{T}' - \mathrm{T}^L) \geq 0$, and $V^{c,\pi^L}_{M^{\mathrm{T}'}_{c^*}} \geq 0$.

Since $\tilde{c} = c^*$, and there exists a policy π^L such that $\mathbb{E}_{\mu_0}[V^{c,\pi^L}_{M^{\mathrm{T}^L}_{c^*}}(s)] = \mathbb{E}_{\mu_0}[V^{c,\pi^e}_{M^{\mathrm{T}^e}_{c^*}}(s)] = \epsilon$. According to the above inequality, we have that $\mathbb{E}_{\mu_0}[V^{c,\pi^L}_{M^{\mathrm{T}'}_{c^*}}(s)] \geq \epsilon$.

To prove **(b)**, we show that $\forall \mathrm{T}' \in \{\mathrm{T} : \max_{s,a} |\mathrm{T} - \mathrm{T}^L| \leq 2(1-\alpha)\}$, the following inequality must hold:

$$\begin{aligned}
\mathbb{E}_{\mu_0}[V^{c,\pi^{pl}}_{M^{\mathrm{T}'}_{c^*}}(s)] &\leq \max_{\mathrm{T}' \in \mathcal{T}^L_\alpha} \mathbb{E}_{\mu_0}[V^{c,\pi^{pl}}_{M^{\mathrm{T}'}_{c^*}}(s)] \\
&\overset{(1)}{\leq} \max_{\pi^{op}\in\Pi} \mathbb{E}_{\mu_0}[V^{c,\alpha\pi^{pl}+(1-\alpha)\pi^{op}}_{M^{\mathrm{T}^L}_{c^*}}(s)] \\
&\overset{(2)}{=} \mathbb{E}_{\mu_0}[V^{c,\alpha\pi^{pl}+(1-\alpha)\pi^{op}}_{M^{\mathrm{T}^L}_{c^*}}(s)] \\
&= \mathbb{E}_{\mu_0}[V^{c,\pi^{mix}}_{M^{\mathrm{T}^L}_{c^*}}(s)] \\
&= \mathbb{E}_{\mu_0}[V^{c,\pi^e}_{M^{\mathrm{T}^e}_{c^*}}(s)] \\
&= \epsilon
\end{aligned}$$

(1) holds due to Proposition 4.1.

(2) holds due to the fact that the trained opponent policy π^{op} is indeed the cost-maximization policy (see objective (8)). □

B.6. Proof of Proposition A.3

Proof. Given a state $s_t \in S^{\pi^{pl}}_\forall$, if the player policy takes the action $a^{pl}_t \in \mathcal{A}^{\pi^{pl}}_\forall$, according to the definition of robust action set, we have:

$$Q^{pl}_c(s_t, a^{pl}_t, a^{op}_t) \leq \epsilon, \forall a^{op}_t \in \mathcal{A}$$

In this work, since we focus on the hard constraint ($\epsilon = 0$) and the cost function $Q_c^{pl}(s_t, a_t^{pl}, a_t^{op}) \geq 0$, the problem reduces to $Q_c^{pl}(s_t, a_t^{pl}, a_t^{op}) = 0$. By Equation (18), we have:

$$\begin{aligned} Q_c^{pl}(s_t, a_t^{pl}, a_t^{op}) &= \sum_{s_{t+1}} \mathrm{T}^{\mathrm{two},\alpha}(s_{t+1} \mid s_t, a_t^{pl}, a_t^{op}) \cdot \Big[-\log \phi_{\omega,\alpha,t}(\cdot) + \gamma \max_{a_{t+1}^{op}} Q_c^{pl}(s_{t+1}, \pi^{pl}(a_{t+1}^{pl} \mid s_{t+1}), a_{t+1}^{op}) \Big] \\ &= 0 \end{aligned}$$

Since the cost $c = -\log \phi_\omega(\cdot) \geq 0$ and $\gamma \geq 0$, regarding any of the possible state s_{t+1} after the transition, we have

$$\max_{a_{t+1}^{op}} Q_c^{pl}(s_{t+1}, \pi^{pl}(a_{t+1}^{pl} \mid s_{t+1}), a_{t+1}^{op}) = 0$$

which means that

$$\min_{a_{t+1}^{pl}} \max_{a_{t+1}^{op}} Q_c^{pl}(s_{t+1}, a_{t+1}^{pl}, a_{t+1}^{op}) = 0$$

According to the definition of $S_\forall^{\pi^{pl}}$, we have $s_{t+1} \in S_\forall^{\pi^{pl}}$. By recursion, we can prove the proposition completely. □

C. Experimental Details

C.1. Environmental Details

Gridword. The Gridworld environment is a map with a number of grids to move. We establish three unique scenarios, as depicted in Figure 2. At each moment, the agent has the freedom to move to any of the 8 neighboring grids by taking a single step. Commencing from the initial position, the agent receives a reward of 1 when it successfully reaches the target location while adhering to the imposed constraints, and a reward of 0 for any other circumstances. Furthermore, we introduce adjustments to the environment to mimic a degree of randomness when testing. In particular, we construct a Windy Gridworld with an upward-blowing wind in the environment, where with a predetermined probability the agent will be further pushed northward based on its intended action.

Blocked Half-Cheetah. Within this setting, the agent commands a bipedal robot with a pair of legs. The agent has 18 observation dimensions and 6 action dimensions. The agent's reward hinges on the distance the robot covers between successive time intervals and is adjusted by a penalty linked to the magnitude of the input action. The game persists until reaching a maximum time step of 1000. Notably, given the robot's ease of backward movement compared to forward, we establish a blocked area where the X-coordinate is less than -3. Consequently, the robot's movement is confined to the region where the X-coordinate is greater than or equal to -3.

Blocked Ant. Within this particular environment, the agent takes charge of a quadrupedal robot. The agent has 113 observation dimensions and 8 action dimensions. The agent's rewards are contingent on both the robot's distance from the origin and the additional benefit gained for maintaining stability. The game concludes upon reaching a maximum time step of 500. Just like the Blocked Half-Cheetah environment, we apply a constraint that restricts movement within the X-coordinate region of less than -3. Consequently, the robot is exclusively allowed to operate within the realm where the X-coordinate is greater than or equal to -3.

Crippled Walker. In this setting, the agent assumes control of a bipedal robot, tasked with the challenge of teaching it how to walk. The agent has 18 observation dimensions and 6 action dimensions. The game concludes under two distinct conditions: when the robot loses its balance or when it reaches the maximum time step of 500. The agent's reward is ascertained based on the distance the robot covers between two consecutive time intervals, offset by a penalty proportional to the magnitude of the input action. Different from the above tasks that constrain the locations, we investigate the constraint with angle in this task. Specifically, we impose a constraint on the thigh angle θ of the agent, limiting it to $|\theta| \leq 0.6$. Consequently, the robot is prevented from running too quickly to keep stable.

C.2. Noise Designs

Fully Random Noise. To simulate the transition dynamics mismatch between the training and testing environment, we directly add the fully random noise into the transition function. Specifically, we incorporate a Gaussian noise $\eta \sim \mathcal{N}(\mu, \sigma)$

into the transition function at each step such that $p\left(s_{t+1} \mid s_t, a_t\right)=T\left(s_t, a_t\right)+\eta$. Since our approach assumes that the mismatch between training and deployment environments is bounded in the uncertainty set $\mathcal{T}_\alpha^L$, where the hyperparameter, $1-\alpha$, serves as a measure of the extent to which our model can effectively handle potential variations in environmental dynamics to maintain safety. In this experiment, we fix $\alpha=0.95$ across the environments, representing the intended mismatch during the training process. To align with it, we utilize the consistent pair of $\mu=0$ with $\sigma=0.1, 0.05$ and 0.01 to represent the larger, equal, and smaller noise than the assumed one.

Partially Random Noise. In contrast to the fully random noise introduced into the overall transition function, partially random noise represents an alternative form of randomness with a more specific focus. To clarify, rather than introducing the Gaussian noise to the entire transition function, it is exclusively applied to the particular restricted elements of the agent, which serves as a more specific disturbance. We carefully select a set of appropriate scales of the partially random noise for each task as follows: 1) Blocked Half-Cheetah: $\sigma=1, 0.8, 0.5$ on the *X-coordinate* of the agent; 2) Blocked Ant: $\sigma=1, 0.5, 0.2$ on the *X-coordinate* of the agent; 3) Crippled Walker: $\sigma=0.1, 0.05, 0.01$ on the *thigh* of the agent.

Attack Noise. Given our primary emphasis on safety, we employ attack noise to replicate scenarios where constraints are more prone to violation during deployment. This assessment tests the model's resilience under worst-case conditions. We assume that the attack noise possesses knowledge of the ground-truth constraints and uses it to prompt the agent to breach those constraints. Specifically, the attack noise is sampled from a uniform distribution $U(0, b)$, where the set of different scales of attack noise for each task is designed as follows: 1) Blocked Half-Cheetah: $b=0.4, 0.35, 0.3$ on the *X-coordinate* of the agent to push it backward; 2) Blocked Ant: $b=0.1, 0.05, 0.02$ on the *X-coordinate* of the agent to push it backward; 3) Crippled Walker: $b=0.1, 0.05, 0.01$ on the *thigh* of the agent to push it larger.

C.3. Hyperparameters

We summarize the main hyperparameters in this work in Table C.3.

Table 2. List of the utilized hyperparameters in this work. To ensure equitable comparisons, we maintain consistency in the parameters of the same neural networks across different models.

Parameters	Blocked Half-Cheetah	Blocked Ant	Crippled Walker
General			
Expert Rollouts	10	50	50
Max Length	1000	500	500
Gamma	0.99	0.99	0.99
Forward Timesteps	200000	200000	200000
PPO			
Steps	2048	2048	2048
Reward-GAE-λ	0.95	0.9	0.9
Cost-GAE-λ	0.95	0.9	0.9
Policy Network	64, 64	64, 64	64, 64
Reward Network	64, 64	64, 64	64, 64
Cost Network	64, 64	64, 64	64, 64
Policy Learning Rate	3e-4	3e-5	1e-4
Lagrangian			
Penalty-based Methods			
Penalty Initial Value	1	0.1	0.1
Penalty Learning Rate	0.1	0.05	0.1
Network-based Methods			
Maximum Lambda Network	50	50	50
Lambda Network Learning Rate	1e-5	3e-5	1e-5
Constraint Function			
Network	20	40, 40	64
Learning Rate	0.005	0.005	0.003
Backward Iterations	10	5	10

D. More Experimental Results

D.1. More Results in Discrete Environments

Figure 6 illustrates the feasible rewards obtained by different methods under different levels of upwind in three Gridworld environments.

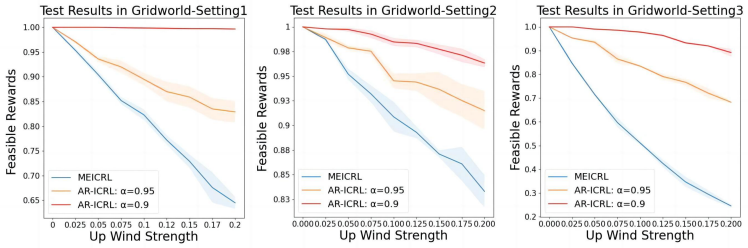

Figure 6. The feasible rewards of different methods under different levels of upwind.

Figure 7 shows the trajectories generated by MEICRL and AR-ICRL in the ideal training environments without wind, serving as a supplementary result of trajectories with wind (Figure 3). We find that larger opponent strength (lower α) could lead to more conservative policies that are more robust against dynamics mismatches.

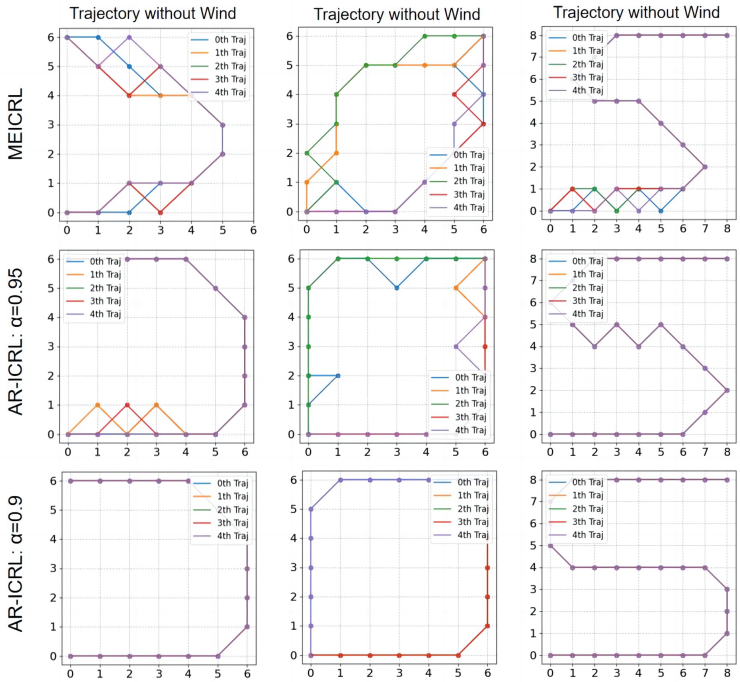

Figure 7. The trajectories generated by MEICRL and AR-ICRL in three Gridworld settings without wind.

D.2. More results in Continuous Environments

Table 3 shows the evaluation results tested within the identical training environment. Table 4 and Table 5 present the results with medium and small scales of noises in the testing environments, respectively.

Table 3. Evaluation results in three environments without noise. Each value is reported as the mean ± standard deviation for 50 episodes and 4 seeds.

Env	Method	Traning Env without Noise		
		Feasible Reward	Episode Cost	Constraint Violation Rate
Blocked Half-Cheetah	MEICRL	3768.2±630.7	0.00±0.00	0%±0%
	BC2L	2467.4±509.4	8.03±13.90	2%±4%
	VICRL	3198.2±1158.6	1.43±2.47	1%±2%
	IRCO	**3818.3±340.1**	0.00±0.00	0%±0%
	AR-ICRL	3621.1±384.6	**0.00±0.00**	**0%±0%**
Blocked Ant	MEICRL	4546.1±1720.9	0.00±0.00	0%±0%
	BC2L	**11478.1±740.8**	0.14±0.24	1%±1%
	VICRL	9246.2±2388.4	0.00±0.00	0%±0%
	IRCO	6698.6±761.0	1.15±1.98	2%±3%
	AR-ICRL	6082.3±1381.9	**0.00±0.00**	**0%±0%**
Crippled Walker	MEICRL	2483.2±221.8	0.06±0.10	1%±2%
	BC2L	1618.5±865.3	3.31±0.65	12%±4%
	VICRL	2223.8±358.0	0.00±0.00	0%±0%
	IRCO	2059.5±266.1	0.07±0.07	2%±2%
	AR-ICRL	**2918.8±99.3**	**0.00±0.00**	**0%±0%**

Table 4. Evaluation results in three environments with medium scale noises. Each value is reported as the mean ± standard deviation for 50 episodes and 4 seeds.

Env	Method	Medium Scale Fully Random Noise			Medium Scale Partially Random Noise			Medium Scale Attack Noise		
		Feasible Reward	Episode Cost	Constraint Violation Rate	Feasible Reward	Episode Cost	Constraint Violation Rate	Feasible Reward	Episode Cost	Constraint Violation Rate
Blocked Half-Cheetah	MEICRL	1092.8±457.1	10.1±15.4	5%±9%	2956.2±689.6	6.9±2.9	32%±6%	1894.2±1260.1	100.2±84.3	57%±29%
	BC2L	1076.2±278.4	45.5±51.9	17%±12%	957.7±486.9	66.7±60.2	40%±15%	76.6±81.7	941.9±50.1	98%±2%
	VICRL	1026.3±460.1	17.6±28.8	7%±10%	2235.1±1050.5	18.5±14.8	35%±10%	1588.2±1367.7	397.5±348.1	62%±27%
	IRCO	1128.6±163.9	12.5±8.9	5%±3%	2528.6±332.7	5.0±0.5	27%±2%	3228.5±611.2	43.9±16.8	25%±12%
	AR-ICRL	**1589.0±324.4**	**3.6±3.2**	**1%±1%**	**3178.0±322.3**	**4.4±1.3**	**24%±4%**	**3679.0±323.1**	**35.2±10.2**	**20%±8%**
Blocked Ant	MEICRL	1198.8±408.2	25.7±18.5	18%±9%	4701.7±1179.6	35.8±18.6	35%±11%	2770.4±1634.2	73.9±53.7	37%±28%
	BC2L	955.4±98.1	29.2±10.3	17%±6%	**8721.9±1618.5**	33.5±27.2	37%±10%	**9030.8±2697.9**	62.5±58.2	31%±30%
	VICRL	1032.6±289.1	28.8±6.6	19%±8%	8083.1±3712.2	41.0±20.1	39%±20%	8785.1±4349.5	85.2±82.1	34%±36%
	IRCO	757.5±207.9	41.5±17.3	18%±7%	4758.5±886.8	35.2±33.2	34%±16%	3978.6±1142.2	81.2±78.4	37%±27%
	AR-ICRL	**1292.4±263.2**	**18.1±3.7**	**12%±5%**	5782.6±1039.3	**13.2±2.8**	**21%±9%**	4986.0±1659.4	**34.8±29.9**	**19%±11%**
Crippled Walker	MEICRL	164.6±40.1	3.46±2.97	34%±23%	835.0±438.1	4.46±2.54	41%±27%	409.2±241.1	12.2±9.6	55%±21%
	BC2L	171.4±55.1	4.09±2.15	38%±14%	694.4±437.5	5.73±4.94	47%±27%	240.8±156.5	3.5±1.8	57%±33%
	VICRL	197.8±36.1	2.75±0.88	28%±8%	**1246.8±185.3**	1.22±0.67	27%±8%	355.4±262.7	6.5±4.1	54%±31%
	IRCO	104.2±21.4	7.88±6.28	30%±18%	910.1±232.7	1.87±0.75	26%±17%	331.1±230.6	3.8±1.1	27%±12%
	AR-ICRL	**237.6±35.7**	**2.29±0.81**	**24%±9%**	1138.5±152.9	**1.19±0.43**	**22%±12%**	**451.2±122.8**	**1.9±0.5**	**18%±2%**

Table 5. Evaluation results in three environments with small-scale noises. Each value is reported as the mean ± standard deviation for 50 episodes and 4 seeds.

Env	Method	Small Scale Fully Random Noise			Small Scale Partially Random Noise			Small Scale Attack Noise		
		Feasible Reward	Episode Cost	Constraint Violation Rate	Feasible Reward	Episode Cost	Constraint Violation Rate	Feasible Reward	Episode Cost	Constraint Violation Rate
Blocked Half-Cheetah	MEICRL	3191.1±393.5	2.51±4.35	1%±1%	3684.2±642.1	2.22±0.89	11%±2%	3180.2±1468.1	30.1±19.5	29%±26%
	BC2L	1682.4±523.9	14.80±25.71	4%±7%	1273.8±498.4	38.30±35.14	29%±13%	199.8±236.0	848.5±171.2	98%±4%
	VICRL	2768.6±993.7	1.19±2.06	1%±1%	2798.6±1143.3	14.42±9.95	16%±6%	2479.1±1534.5	268.0±169.8	41%±21%
	IRCO	3298.0±215.8	2.17±3.07	1%±1%	3718.0±164.7	0.87±0.19	10%±1%	3926.6±368.0	32.1±19.5	12%±4%
	AR-ICRL	**3358.2±343.4**	**0.00±0.00**	**0%±0%**	**4049.5±260.2**	**0.81±0.19**	**8%±2%**	**4369.7±390.5**	**16.8±14.7**	**5±3%**
Blocked Ant	MEICRL	3831.2±1339.5	2.08±0.61	1%±1%	4890.7±1653.6	8.74±5.58	6%±4%	3890.2±1749.1	11.3±11.8	10%±7%
	BC2L	**9068.0±1573.7**	1.25±2.07	2%±3%	**11708.1±2213.5**	7.74±6.69	6%±6%	10987.5±2195.6	12.5±14.8	9%±10%
	VICRL	7813.3±1368.1	1.22±1.31	2%±2%	10892.0±3113.8	10.70±6.34	9%±7%	**11976.0±3139.4**	18.1±28.5	9%±13%
	IRCO	5258.8±1129.4	0.00±0.00	0%±0%	6579.7±442.9	9.91±17.21	6%±11%	5538.5±656.9	22.1±16.8	14%±12%
	AR-ICRL	5967.9±744.5	**0.00±0.00**	**0%±0%**	6892.2±669.3	**0.00±0.00**	**0%±0%**	6056.1±1338.7	**0.0±0.0**	**0%±0%**
Crippled Walker	MEICRL	1056.0±268.5	1.24±0.78	19%±10%	2321.7±203.6	0.0±0.0	0%±0%	2294.5±298.2	0.11±0.05	6%±4%
	BC2L	1178.3±628.7	3.06±4.54	16%±14%	2059.5±121.2	0.0±0.0	0%±0%	1398.7±759.6	2.01±3.08	16%±13%
	VICRL	1467.2±196.9	1.25±0.14	15%±2%	2210.1±424.7	0.0±0.0	0%±0%	2053.1±386.5	0.02±0.03	1%±1%
	IRCO	800.4±40.2	2.48±2.42	12%±10%	1893.8±231.6	0.0±0.0	0%±0%	1243.0±190.5	0.00±0.00	0%±0%
	AR-ICRL	**1865.8±184.2**	**0.00±0.00**	**0%±0%**	**2483.0±243.4**	**0.0±0.0**	**0%±0%**	**2439.4±228.6**	**0.00±0.00**	**0%±0%**

Figure 8, Figure 9, and Figure 10 visualize the learned constraints by MEICRL and AR-ICRL in the Blocked Ant, Blocked Half-cheetah, and Crippled Walker respectively. Specifically, the red curve in the left is the predicted constraint distribution by utilizing synthetic samples, and the blue points are the recorded information (e.g., x-position or thigh angle) of the policy with the predicted costs. The right figure is a histogram that shows the number of states with a specific feature value (e.g., x-position) during testing. By observing the results, we find that compared to MEICRL, AR-ICRL can enable safer control and keep a relative distance from the constraint locations. This can be attributed to the fact that the policy must resist the attacks from the opponent policy when training.

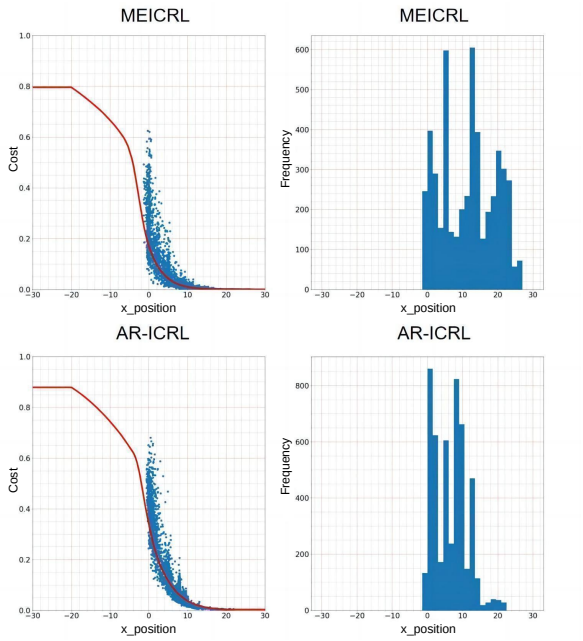

Figure 8. Visualization results in Blocked Ant environment with MEICRL and AR-ICRL.

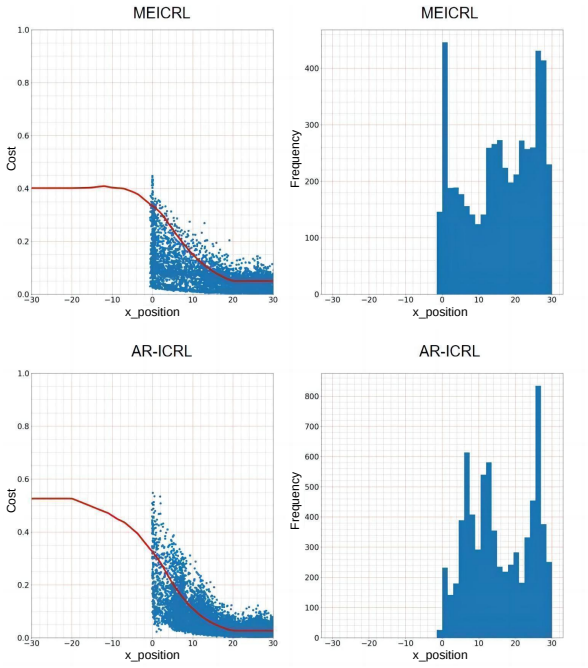

Figure 9. Visualization results in Blocked Half-Cheetah environment with MEICRL and AR-ICRL.

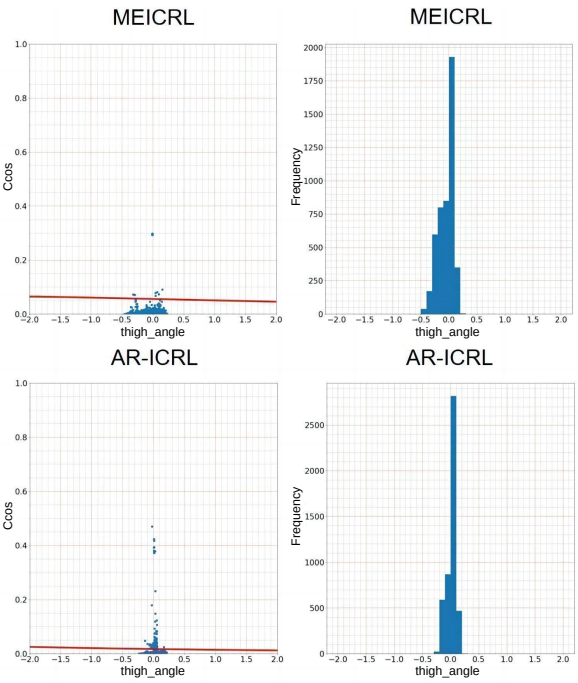

Figure 10. Visualization results in Cirppled Walker environment with MEICRL and AR-ICRL.