Iterative Regularized Policy Optimization with Imperfect Demonstrations

Xudong Gong [1 2] **Dawei Feng** [1 2] **Kele Xu** [1 2] **Yuanzhao Zhai** [1 2]
Chengkang Yao [3] **Weijia Wang** [3] **Bo Ding** [1 2] **Huaimin Wang** [1 2]

Abstract

Imitation learning heavily relies on the quality of provided demonstrations. In scenarios where demonstrations are imperfect and rare, a prevalent approach for refining policies is through online fine-tuning with reinforcement learning, in which a Kullback–Leibler (KL) regularization is often employed to stabilize the learning process. However, our investigation reveals that on the one hand, imperfect demonstrations can bias the online learning process, the KL regularization will further constrain the improvement of online policy exploration. To address the above issues, we propose Iterative Regularized Policy Optimization (IRPO), a framework that involves iterative offline imitation learning and online reinforcement exploration. Specifically, the policy learned online is used to serve as the demonstrator for successive learning iterations, with a demonstration boosting to consistently enhance the quality of demonstrations. Experimental validations conducted across widely used benchmarks and a novel fixed-wing UAV control task consistently demonstrate the effectiveness of IRPO in improving both the demonstration quality and the policy performance. Our code is available at https://github.com/GongXudong/IRPO.

1. Introduction

Imitation Learning (IL) (Zheng et al., 2022) is a paradigm in which an agent acquires task proficiency by imitating the behavior of a demonstrator or expert. Unlike reinforcement learning (RL) (Sutton & Barto, 2018), which relies on trial and error, IL offers distinct advantages in sample efficiency, making it a prevalent choice for applications in games (Silver et al., 2016; Vinyals et al., 2019) and control systems (Shukla et al., 2020; Oh et al., 2023). However, the efficacy of IL crucially depends on the availability of a substantial number of high-quality demonstrations (Kim et al., 2013; Wu et al., 2019). In practical scenarios, only a limited supply of imperfect demonstrations is often accessible, due to factors such as inexperience, fatigue, or unnecessary operations by demonstrators (Sasaki & Yamashina, 2020; Hedlund-Botti & Gombolay, 2023). Consequently, policies learned through IL tend to be constrained by the quality of these imperfect demonstrations (Brown et al., 2020). The challenge at hand lies in devising methodologies to extract superior policies from such imperfect demonstrations, which constitutes a pivotal obstacle in IL research (Sasaki & Yamashina, 2020; Xu et al., 2022).

A promising approach to address the aforementioned challenge involves further refining the policy acquired through offline IL with online learning, such as RL (Kim et al., 2013; Wu et al., 2019; Huang et al., 2022). In this offline-to-online approach, the primary objective during offline learning is to maximize the likelihood of actions (Pomerleau, 1991) or the occupancy measure (Ho & Ermon, 2016). However, a notable distinction arises when transitioning from offline to online learning, as the objective of online learning (Sutton & Barto, 2018) shifts towards maximizing cumulative rewards in the task environment. This difference in objectives introduces instability into the transition process (Vinyals et al., 2019; Baker et al., 2022). To mitigate this instability, distance-based regularizations, often employing KL divergence (Rohatgi & Saleh, 2015; Vieillard et al., 2020), are commonly incorporated into the optimization objective of online learning (Vinyals et al., 2019; Baker et al., 2022; Ramrakhya et al., 2023). The regularization term plays a crucial role in stabilizing the training process by constraining differences between two policies (Rudner et al., 2021).

Despite its success, the offline-to-online learning approach encounters two significant challenges when dealing with imperfect demonstrations. **Over-constrained Exploration due to KL Regularization:** One notable issue arises in the form of over-constrained exploration during online learning, induced by the KL regularization (Over-constrained exploration). This regularization method imposes limitations on

[1]College of Computer, National University of Defense Technology, Changsha, Hunan, China [2]State Key Laboratory of Complex & Critical Software Environment, Changsha, Hunan, China [3]Flight Automatic Control Research Institute, AVIC, Xian, Shanxi, China. Correspondence to: Dawei Feng <davyfeng.c@qq.com>.

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

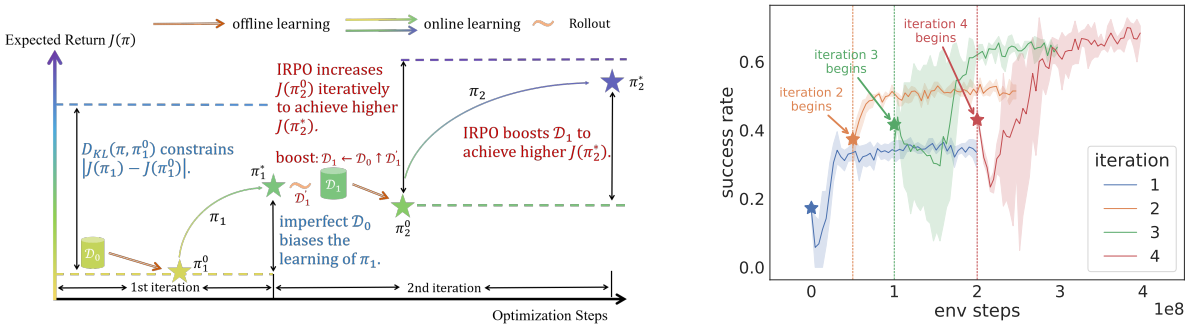


Figure 1. **Left:** The KL regularization, denoted as $D_{KL}(\pi_1, \pi_1^0)$, introduces the over-constrained exploration problem, limiting the upper bound of $\|J(\pi_1) - J(\pi_1^0)\|$. Imperfect $\mathcal{D}_0$ induces the primacy bias problem, constraining the enhancement of $J(\pi_1)$. IRPO makes π_1^* to serve as the demonstrator for successive learning iterations to roll out $\mathcal{D}_1'$, with a data boosting $\mathcal{D}_1 \leftarrow \mathcal{D}_0 \uparrow \mathcal{D}_1'$ to consistently enhance the quality of demonstrations. High quality $\mathcal{D}_1$ mitigates both the two above problems. **Right:** Performance of IRPO on the fixed-wing attitude control task. IRPO converges the policy to a higher performance with each iteration.

the exploration of the online policy, resulting in a restricted improvement in the expected return of the policy compared to the policy derived offline from imperfect demonstrations.

Primacy Bias Problem in the Online Learning Process: Another challenge manifests as the primacy bias problem in the online learning process. This bias, recognized as a tendency to over-fit to early or low-quality experiences, has been identified as a detriment to the overall learning process in RL (Nikishin et al., 2022). The bias introduces a constraint on the enhancement of the policy's expected return. The presence of these two problems is substantiated through theoretical analysis and experiment validation.

To address the aforementioned issues, we introduce the **I**terative **R**egularized **P**olicy **O**ptimization (IRPO) method in this paper. IRPO adopts a dual-pronged strategy, employing iterative training and demonstration boosting to enhance demonstration quality. Specifically, iterative training capitalizes on the policy learned online, utilizing it as the demonstrator for the subsequent training iteration. Simultaneously, demonstration boosting utilizes demonstrations generated by the online-learned policy to consistently enhance demonstration quality. The utilization of high-quality demonstrations is pivotal in resolving both of the aforementioned issues. Fig. 1 presents the overview of IRPO. We conduct extensive experiments across diverse complex tasks, including articulated-body control (Tunyasuvunakool et al., 2020), robotic arm control (Gallouédec et al., 2021), and fixed-wing Unmanned Aerial Vehicle (UAV) attitude control. Our results illustrate that IRPO efficiently learns well-performing policies from imperfect demonstrations. Our contributions are succinctly summarized as follows:

- We provide an exhaustive examination of the primary challenges that arise during offline IL to online RL with imperfect demonstrations. Our contribution includes a theoretical analysis focused on the over-constrained exploration problem and an empirical investigation on both issues through carefully designed experiments.
- We propose the IRPO, a framework comprised of two integral components: iterative training and demonstration boosting. These components synergistically address both the over-constrained exploration and primacy bias problems concurrently.
- We systematically assess the efficacy of IRPO across diverse and complex tasks, encompassing articulated-body control, robotic arm control, and fixed-wing UAV control. Our results demonstrate that IRPO exhibits a consistent ability to enhance policy and demonstration quality over successive iterations.

2. Related Work

Learn from imperfect demonstrations: IL proves effective in learning policies from demonstrations; however, the efficacy of the acquired policy is intricately tied to the quality of the provided demonstrations. Consequently, various approaches have been proposed to adapt IL to imperfect demonstrations. Some of these approaches necessitate additional information pertaining to demonstrations, such as annotations on non-optimality (Grollman & Billard, 2012; Wu et al., 2019; Wang et al., 2021b), or comparisons between distinct demonstrations (Ibarz et al., 2018; Brown et al., 2019). In contrast, others involve the fine-tuning of policies through online interactions with environments (Kim et al., 2013; Hester et al., 2018; Jing et al., 2020; Tsurumine & Matsubara, 2022; Ball et al., 2023). Given that the former category of approaches places an increased burden on the demonstrator, our study concentrates on the latter category.

Fine-tune policy by online learning with regularization: During the transition from offline to online learning, the shift in objective from imitating the demonstrator to maximizing

cumulative rewards from environments introduces training instability. To mitigate this, AlphaStar (Vinyals et al., 2019), VPT (Baker et al., 2022), and PIRLNav (Ramrakhya et al., 2023) incorporate KL regularization into the online learning objective. This integration aims to balance the original objective of RL, which is optimality, with the stability objective of KL regularization (Li et al., 2023). To accentuate the original RL objective during training (Schmitt et al., 2018), some works adjust the strength of regularization or the reference policy to alleviate the imposed constraint. Schmitt et al. (2018); Agarwal et al. (2022); Shenfeld et al. (2023) employ a strategy of annealing the strength of KL regularization during training. In a different approach, Li et al. (2023) gradually evolve the reference policy with the learning policy, enhancing the performance of the reference policy and, consequently, indirectly relaxing the constraint strength. While these methods prove effective with high-quality demonstrations, they overlook the impact of imperfect demonstrations on the learning process, as highlighted in (Nikishin et al., 2022). Furthermore, our analysis also reveals that the KL regularization constrains the policy's exploration, resulting in a limited improvement in policy performance compared to the imperfect demonstrator.

Cognitive bias in RL: The primacy bias, a well-explored cognitive bias in human learning, refers to individuals often forming generalizations based on initial facts and paying less attention to subsequent ones (Marshall & Werder, 1972). Nikishin et al. (2022) extends this concept to the learning process of RL. The study reveals that over-fitting to early or low-quality experiences can detrimentally impact the overall RL learning process. Notably, Nikishin et al. (2022) observes that demonstrations from primed policies prove sufficient to train an improved policy, presenting a practical solution: periodic reinitialization of some policy parameters while preserving the replay buffer. This observation prompts our consideration that fine-tuning policies learned offline with imperfect demonstrations may be susceptible to the primacy bias problem. Moreover, leveraging the primed policy can provide high-quality demonstrations, serving as a potential remedy to mitigate the impact of the primacy bias.

3. Background and Notation

Reinforcement learning can be described by the Markov Decision Process (MDP) (Sutton & Barto, 2018). We consider the Episodic RL, which is modeled by finite-horizon MDP: $M = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma, H \rangle$, where $\mathcal{S}, \mathcal{A}, \gamma \in (0, 1], H$ are the state set, action set, discount factor, and the time horizon length respectively; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition probabilities, where $\Delta(\mathcal{X})$ denotes the probability distribution over a set $\mathcal{X}$; $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. RL aims to find a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ that can maximize the expected return, calculated by either the expected sum of discounted rewards $J_{RL}(\pi) = \mathbb{E}\big[\sum_{t=0}^{H-1} \gamma^t r(s_t, a_t)|\pi, \mathcal{T}\big]$ or the expected average of rewards $J_{RL}(\pi) = \mathbb{E}\big[\frac{1}{H}\sum_{t=0}^{H-1} r(s_t, a_t)|\pi, \mathcal{T}\big]$. If the policy is parameterized by θ, then $\theta^* = \arg\max_\theta J_{RL}(\pi_\theta)$.

When fine-tuning a pre-trained policy with RL, it frequently results in performance drops without regularization (Wang et al., 2023b). In order to stabilize the fine-tuning process, the regularization between the training policy and the pre-trained policy π^0, of which KL is the most frequently utilized, is utilized to extend the objective of RL (Vieillard et al., 2020):

$$J_{o2o}(\pi) = \mathbb{E}\big[\sum_{t=0}^{H-1} \gamma^t \big(r(s_t, a_t) - \lambda log(\frac{\pi(a_t|s_t)}{\pi^0(a_t|s_t)})\big)\big]. \quad (1)$$

Imitation learning is a data-driven, sample-efficient method for learning policies by imitating demonstrators (Belkhale et al., 2023). IL assumes access to a dataset $\mathcal{D}_E = (\tau_1, \ldots, \tau_N)$ of N demonstrations. $\tau_i = \{(s_1, a_1), \ldots, (s_{T_i}, a_{T_i})\}$ is a sequence of length T_i of state-action pairs sampled by the demonstrator $\pi_E(\cdot|s_t)$ through environment dynamics $\mathcal{T}(\cdot|s_t, a_t)$. The objective of IL is to learn a policy $\pi_\theta : \mathcal{S} \to \Delta(\mathcal{A})$ parameterized by θ from $\mathcal{D}_E$. Behavioral cloning (Pomerleau, 1991) is a widely applied IL method, which learns the imitation policy by optimizing a supervised loss to maximize the likelihood of demonstrator actions (Sasaki & Yamashina, 2020):

$$\mathcal{L}(\theta) = -\mathbb{E}_{(s,a)\sim\mathcal{D}_E}[\log \pi_\theta(a|s)]. \quad (2)$$

which optimizes the following objective under finite state-action pairs from demonstrator (Belkhale et al., 2023):

$$\begin{aligned}\mathcal{L}(\theta) &= \mathbb{E}_{s\sim d_{\pi_E}(\cdot)}\big[D_{KL}\big(\pi_E(\cdot|s), \pi(\cdot|s)\big)\big] \\ &= -\mathbb{E}_{s\sim d_{\pi_E}(\cdot), a\sim\pi_E(\cdot|s)}\big[\log \pi_\theta(a|s)\big] + C,\end{aligned} \quad (3)$$

where $d^t_\pi(s') = \int_{s,a} d^{t-1}_\pi(s)\pi(a|s)\mathcal{T}(s'|s, a)\, ds\, da$ and $d_\pi = \frac{1}{H}\sum_{t=0}^{H-1} d^t_\pi$ denote the distribution of states at time step t if execute π from time step 0 to $t-1$ and the average distribution of states over H time steps, C is the entropy of the demonstrator state-action distribution, which is constant with respect to θ.

4. Methodology

Here, we first show that the KL regularization over-constrains the exploration of the policy. Then, we provide a detailed exposition of the IRPO.

4.1. The Over-Constrained Exploration Problem

In imitation learning, action divergence serves as a metric quantifying the discrepancy between the learned policy and

the demonstrator, denoted as $D_f(\pi(\cdot|s), \pi_E(\cdot|s))$. Biases in algorithms or demonstrations, such as mismatched action representations or inadequate demonstration quantity, can lead to discrepancies in action divergence (Belkhale et al., 2023). Integrating action divergence into optimization objectives helps align the action distribution of the learned policy with that of the demonstrator. Belkhale et al. (2023) delve into the relationship between the policy's action divergence and state distribution divergence, proposing:

Lemma 4.1 ((Belkhale et al., 2023)). *Given a policy π, a demonstrator π_E, and an environment horizon length H, the difference in the state visitation distribution between π and π_E can be bounded by:*

$$D_{KL}(d_\pi, d_{\pi_E}) \leq \frac{1}{H}\sum_{t=0}^{H-1}(H-t)D_{KL}^{s\sim d_\pi^t}\big(\pi(\cdot|s), \pi_E(\cdot|s)\big). \quad (4)$$

Lemma 4.1 implies that the KL divergence between the state visitation distributions of two policies is constrained by the expected KL divergence between their action distributions over the state distribution of the first policy. In the specific context of finite-horizon MDP with a reward function that depends only on states, we further elucidate the connection between the difference in the policy's expected return and the action divergence in Theorem 4.2.

Theorem 4.2. *Given a finite-horizon MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma, H\rangle$, in which the reward function depends only on states $r : \mathcal{S} \to \mathbb{R}$, and a demonstrator π_E, for any policy π, the difference in the average of expected return with π_E is bounded by the KL divergence between π and π_E:*

$$|J(\pi) - J(\pi_E)| \leq \max_s |r(s)|\sqrt{2HD_{KL}^{s\sim d_\pi}\big(\pi(\cdot|s), \pi_E(\cdot|s)\big)}. \quad (5)$$

The proof for Theorem 4.2 relies on Pinsker's inequality (Fedotov et al., 2003) and Lemma 4.1 (provided in detail in Appendix A). While the incorporation of KL regularization between policies π and π_E in policy optimization aims to confine the policy's exploration for training stabilization, Theorem 4.2 reveals that it simultaneously restricts the relative improvement of $J(\pi)$ over $J(\pi_E)$ in the context of finite-horizon MDP with a reward function dependent only on states. When the demonstrator π_E is imperfect, $J(\pi)$ is bounded by Eq. 5, preventing it from reaching the optimal $J(\pi^*)$. To substantiate the practical implications of Theorem 4.2, we conduct numerous experiments across various tasks. Detailed results can be found in Section 5.3.

4.2. The Proposed IRPO Method

The analysis presented in Section 4.1 highlights two primary factors contributing to the over-constrained exploration problem: (1) The low expected return of the demonstrator, $J(\pi_E)$. (2) The KL-regularization-determined maximum improvement in policy's expected return over the demonstrator, $\max_s |r(s)|\sqrt{2HD_{KL}^{s\sim d_\pi}\big(\pi(\cdot|s), \pi_E(\cdot|s)\big)}$.

While the stabilizing effect of KL regularization during training is acknowledged, relaxing it does not consistently yield high-performance policies, particularly in challenging tasks. The results corresponding to this observation are detailed in Section 5.3. Consequently, our attention turns to mitigating the over-constrained exploration problem by addressing the imperfect demonstrator π_E. It is crucial to note that in the KL regularization term $D_{KL}(\pi(\cdot|s), \pi_E(\cdot|s))$, the second term, π_E, represents the policy learned offline from imperfect demonstrations, not the actual demonstrator that generates the demonstrations. Thus, the root cause of the over-constrained exploration problem is attributed to imperfect demonstrations. Meanwhile, the primacy bias problem also stems from learning from imperfect demonstrations.

To this end, IRPO is specifically designed to alleviate both the over-constrained exploration problem and the primacy bias problem by enhancing demonstration quality. It achieves this by utilizing the policy learned online as the demonstrator for the subsequent training iteration and consistently improving demonstration quality with data rolled out by the aforementioned demonstrator. IRPO is detailed in Algorithm 1.

Algorithm 1 Iterative Regularized Policy Optimization (IRPO) method

Require: demonstrations $\mathcal{D}_E$, number of iteration: K
Ensure: $\pi^*_{1\ldots K}$
1: $\mathcal{D}_0 \leftarrow \mathcal{D}_E$
2: **for all** $k \in 1 \ldots K$ **do**
3: train π_k^0 by Eq. 6 on $\mathcal{D}_{k-1}$
4: $\pi_k \leftarrow \pi_k^0$
5: optimize π_k by Eq.7 with the imitation policy as π_k^0 to get the π_k^*
6: sample with π_k^* to get $\mathcal{D}'_k$
7: update demonstrations $\mathcal{D}_k \leftarrow \mathcal{D}_{k-1} \uparrow_f \mathcal{D}'_k$
8: **end for**

For clarity, we distinguish between different generations of policies in IRPO as follows: in the k^{th} iteration, π_k^0 refers to the imitation policy learned by IL from $\mathcal{D}_{k-1}$, π_k refers to the policy being optimized for online learning, and π_k^* refers to the policy obtained after the online learning optimization is completed. Additionally, $\mathcal{D}_k$ refers to the demonstrations optimized in the k^{th} iteration, and $\uparrow_f$ is the demonstration update operator, which will be introduced in the following. IRPO learns offline policy π_k^0 and online policy π_k^* with:

$$\pi_k^0 \leftarrow \arg\min_\pi -\mathbb{E}_{(s,a)\sim\mathcal{D}_{k-1}}[\log \pi(a|s)], \quad (6)$$

$$\pi_k^* \leftarrow \arg\max_\pi \mathbb{E}\Big[\sum_{t=0}^{H-1}\gamma^t\big(r(s_t,a_t) - \lambda_k log(\frac{\pi(a_t|s_t)}{\pi_k^0(a_t|s_t)})\big)\Big], \quad (7)$$

where λ_k is the strength of KL regularization used in the k^{th} iteration. In the following, we analyze the two components of IRPO in detail.

Iterative training is specifically crafted to enhance the expected return of the imitation policy, denoted as $J(\pi_k^0)$, aiming to alleviate the over-constrained exploration problem. If π_k^* is directly employed as the reference policy for KL regularization in the subsequent iteration, π_{k+1}^0, then the alteration in the policy's expected return can be characterized by Theorem. 4.3.

Theorem 4.3. *Define π_k as the policy obtained the k^{th} iteration by optimizing the objective:*

$$\pi_k \leftarrow \arg\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{H-1} \gamma^t \left(r(s_t,a_t) - \lambda \log\left(\frac{\pi(a_t|s_t)}{\pi_{k-1}(a_t|s_t)}\right)\right)\right],$$

π_0 is the pre-trained policy. Let $v_{max}^{\lambda} := \frac{r_{max}+\lambda \ln|\mathcal{A}|}{1-\gamma}$, ϵ_j is the approximation error of value function at j^{th} iteration, then:

$$|J(\pi_k) - J(\pi^*)| \leq \frac{4}{1-\gamma}\left\|\frac{1}{k+1}\sum_{j=0}^{k}\epsilon_j\right\| + \frac{8}{1-\gamma}\frac{v_{max}^{\lambda}}{k+1}, k \in N.$$

The proof of Theorem. 4.3 is detailed in Appendix. B. Theorem 4.3 demonstrates that the difference in expected return between the learned policy and the optimal policy is constrained, and this difference diminishes as the iteration progresses. In other words, Theorem. 4.3 signifies that the policy's expected return can be progressively improved with iterative training.

While maintaining the online policy π_k^* as the reference policy π_{k+1}^0 aids in mitigating the over-constrained exploration problem, it does not alter the demonstrations, leaving the primacy bias problem unresolved. The value of this iterative approach lies in the theoretical potential for continuous improvement in the policy's expected return. In practice, it is more effective to leverage the policy learned online to assist subsequent training iterations by rolling out demonstrations, as described in the following.

Demonstration boosting is designed to enhance the demonstration quality. The imperfections in demonstrations introduce primacy bias to the online learning process through π_k^0. Nikishin et al. (2022) assert that primacy bias is not a failure to collect proper data per se but rather a failure to learn from it. Therefore, we involve the policy learned online in the next training iteration by rolling out demonstrations. Specifically, after obtaining π_k^* from the k^{th} iteration, we roll out demonstrations $\mathcal{D}_k'$ with π_k^* and update demonstrations as $\mathcal{D}_k \leftarrow \mathcal{D}_{k-1} \uparrow_f \mathcal{D}_k'$. For demonstration $\mathcal{D}$ and a trajectory τ, the demonstration update operator $\uparrow_f$ is defined:

$$\mathcal{D} \uparrow_f \tau = \begin{cases} \mathcal{D} \setminus \tau' \cup \{\tau\}, & \text{if } \exists \tau', \text{s.t.} f(\tau') < f(\tau) \\ \mathcal{D}, & \text{else} \end{cases},$$

where $f(\tau)$ is the indicator function, which can be the sum of reward $R(\tau) = \sum_{t=0}^{H-1} r_t$ collected by τ, the smoothness of τ (Mysore et al., 2021), the length of τ, the difficulty of goal finished by τ, etc. When the right-hand side of the operator is a set of demonstrations $\mathcal{D}' = \{\tau_1, \tau_2, \cdots, \tau_m\}$, the update operator is defined as $\mathcal{D} \uparrow \mathcal{D}' = ((\mathcal{D} \uparrow \tau_1) \uparrow \tau_2 \cdots) \uparrow \tau_m$.

The demonstration update operator guarantees that the average quality described by f over $\mathcal{D} \uparrow_f \mathcal{D}'$ is no less than that of $\mathcal{D}$. The quality improvement of demonstrations helps to mitigate both the over-constrained exploration problem and the primacy bias problem. Therefore, the ideal indicator function should express the same optimization objective as the reward function. This alignment ensures that the demonstrations and the policy are optimized in the same direction during iterative training, thus enabling IRPO to achieve its best performance. Nevertheless, our experiments reveal that IRPO exhibits robustness to the design of the indicator function, with even simple indicator functions yielding impressive results, as detailed in Section 5.6 and Appendix F.

5. Experiments

5.1. Experimental Setups

We conduct experiments encompassing three distinct control tasks. **Articulated-body control:** Halfcheetah and Hopper tasks on the MuJoCo physics engine with D4RL (Fu et al., 2020) datasets. **Robotic Arm Control:** Modified Reach task (Gallouédec et al., 2021) on the Bullet physics engine with demonstrations generated by a PID controller. **Fixed-wing UAV Attitude Control:** Attitude control task in a self-designed fixed-wing UAV environment with demonstrations generated by a PID controller and human play data (Wang et al., 2023a). All four tasks are finite-horizon MDPs with reward functions that fully or substantially satisfy the conditions outlined in Theorem 4.2. The demonstrations utilized in these experiments are imperfect. Further details are available in Appendix C.

It is crucial to acknowledge that the extent of fluctuation during the transition from offline IL to online RL is contingent upon various factors, encompassing environmental dynamics, demonstrations, offline and online learning algorithms, among others. In our specific experimental settings: Halfcheetah exemplifies single-goal easy-transition tasks. Hopper represents single-goal hard-transition tasks. Reach portrays multi-goal easy-transition tasks. Attitude control characterizes multi-goal hard-transition tasks. For supporting experimental evidence delineating the distinctions among these four tasks, please refer to Appendix C.3.6. The inclusion of this diverse set of tasks in our experiments is deliberate, aiming to demonstrate the broad applicability

Table 1. Comparisons on different tasks. The expected return of policy is shown for Halfcheetah, Hopper, and Reach, while the success rate is for the attitude control task. All results of IRPO come from the second training iteration. The mean and variance are shown over 5 random seeds. Optimal values are highlighted in bold, and sub-optimal values are underlined.

Method	Venue	Operation on KL	Halfcheetah	Hopper	Reach	Attitude control
BC	Neural computation (1991)	-	4967.05±36.88	1673.42±175.23	-42.68±1.26	0.17±0.01
AlphaStar VPT PIRLNAV	Nature (2019) NeurIPS (2022) CVPR (2023)	fixed	6690.12±597.12	2641.55±207.13	-10.39±1.92	<u>0.38±0.02</u>
Reincarnating RL TGRL	NeurIPS (2022) ICML (2023)	annealed	6850.23±679.10	2586.6±276.5	-9.34±2.54	0.30±0.02
PROTO	ArXiv (2023)	EMA	<u>6954.97±606.67</u>	<u>2656.11±222.7</u>	<u>-7.36±2.07</u>	0.32±0.01
IRPO (Ours)	-	iterative	**7678.4±60.81**	**3044.7±48.65**	**-5.32±0.43**	**0.54±0.01**
Average improvement over 3 baselines (%)			10.40	14.63	38.72	42.11

and versatility of the IRPO.

5.2. Main Results

We evaluate IRPO against the following baselines: (1) **IL+RL with fixed regularization:** This method involves fine-tuning the policy learned offline by IL with KL-regularized RL in the online learning stage. Commonly employed in approaches like AlphaStar, VPT, PIRLNAV, etc. (2) **IL+RL with annealed regularization:** This method incorporates an annealed KL regularization during training to gradually shift the focus of policy optimization towards the original RL objective (such as Reincarnating RL (Agarwal et al., 2022), TGRL (Shenfeld et al., 2023)). (3) **IL+RL with exponential moving average (EMA) regularization:** This method employs a fixed KL regularization while applying an exponential moving average to the reference policy by the current training policy, i.e., $\pi_k^0 \leftarrow \alpha\pi_k^0 + (1-\alpha)\pi_k, \alpha \in [0,1]$ (such as PROTO (Li et al., 2023)). Implementation details are provided in Appendix D.

Table 1 presents the results on the four tasks. Notably, IRPO outperforms all three baseline algorithms across all four tasks. This suggests that IRPO exhibits effectiveness in learning superior policies from imperfect demonstrations and exhibits applicability across a diverse range of tasks, regardless of their complexity in transitioning from offline to online. We also present an analysis on the convergence rate of IRPO against the aforementioned baselines in Appendix G.3.2. The underlying reasons for this superiority are elaborated in the subsequent sub-sections.

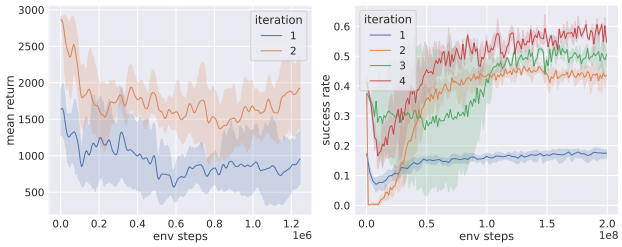
5.3. KL Regularization Over-Constrains the Policy's Exploration

To support Theorem 4.2, we train policies on the four tasks with different KL regularization. Stability in the training process is observed with stronger KL regularizations, as depicted in the results illustrated in Fig. 8 (Appendix G.1) and Fig. 12 (Appendix G.3). It is noteworthy that while

Table 2. $KL(\pi, \pi_0)$ and $J(\pi)$ from training with different strengths of KL regularization. The mean and variance are shown over 5 random seeds. Optimal values are highlighted in bold, and sub-optimal values are underlined.

Task	λ	$KL(\pi, \pi_0)$	$J(\pi)$
Halfcheetah	$+\infty$	0.0±0.0	4967.04±36.88
	10^0	0.06±0.01	5708.93±80.10
	10^{-1}	1.28±0.13	6690.12±597.12
	10^{-2}	6.76±0.88	<u>7392.39±897.89</u>
	10^{-3}	28.07±24.49	**7411.44±836.34**
Hopper	$+\infty$	0.0±0.0	1673.42±175.23
	10^0	0.09±0.02	<u>2619.40±209.39</u>
	10^{-1}	1.61±0.10	**2641.55±207.13**
	10^{-2}	8.35±1.31	2046.55±853.14
	10^{-3}	40.13±18.72	1626.12±770.63
Reach	$+\infty$	0.0±0.0	-42.68±1.25
	10^0	0.21±0.10	-41.61±0.73
	10^{-1}	4.74±0.68	-10.39±1.92
	10^{-2}	9.47±0.41	**-2.89±0.13**
	10^{-3}	41.62±5.31	<u>-3.00±0.19</u>
Attitude Control	$+\infty$	0.0±0.0	0.169±0.004
	10^0	0.08±0.01	0.164±0.004
	10^{-1}	1.88±0.44	0.225±0.031
	10^{-2}	9.53±0.84	<u>0.362±0.018</u>
	10^{-3}	34.91±2.62	**0.383±0.016**
	10^{-4}	136.06±27.82	0.268±0.069

KL regularization enhances training stability, it simultaneously imposes constraints on policy improvement. Table 2 provides a detailed examination of KL values and policy returns on the four tasks under different regularizations (We also present a graphical representation of the table's content for intuitive understanding in Appendix E). The first two columns illustrate the relationship between regularization strength and KL values. Stronger regularization leads to policies closely resembling the imitation policy, characterized by small KL values. Conversely, weaker regularization results in policies significantly deviating from the imitation



(a) Hopper (b) Attitude control task

Figure 2. The return of policy trained by IL+RL without KL regularization under varying levels of demonstration quality. Demonstrations from later iterations are of superior quality compared to those from earlier iterations, as detailed in Section 5.5.

policy, evident in large KL values. Consequently, the KL regularization primarily constrains the exploration space relative to the imitation policy.

The final two columns illustrate the relationship between KL values and the policy's return. For small KL values, an increase in the KL value corresponds to an increase in the policy's return. This observation suggests that the KL value acts as an upper bound on the policy's return, and judiciously increasing the KL value can relax this upper bound, aligning with Theorem 4.2. However, as the KL value becomes excessively large, the policy's return diminishes, accompanied by an escalation in return variance. We hypothesize that, in such cases, the limiting factor for policy optimization is no longer the relaxed upper bound but rather the expanded optimization space introduced by the larger KL value, making the search for superior policies near the imitation policy more challenging.

5.4. Imperfect Demonstrations Induce the Primacy Bias for Online Learning

To demonstrate the emergence of the primacy bias problem during the online learning stage, we train imitation policies using IL with demonstrations of varying qualities. Subsequently, we initialize the policy with the imitation policy and fine-tune it without KL regularization. Experiments are conducted on Hopper and attitude control, representing challenging tasks in the transition from offline to online learning. The demonstrations are sourced from different iterations of IRPO, with later iterations producing higher-quality demonstrations compared to earlier iterations, as detailed in Section 5.5.

Fig. 2 illustrates that demonstrations of higher quality correspond to superior offline and online learning policies. The leftmost points in both charts of Fig. 2 depict the performance of offline learning, indicating that higher-quality demonstrations correspond to better performance of the imitation policy. Given that the only distinction between online learning iterations is the policy initialization, the results from both the Hopper and attitude control experiments indicate that initializing the online learning policy with a higher-performing imitation policy leads to superior online learning performance. This observation implies that higher-quality demonstrations mitigate the primacy bias problem during online learning.

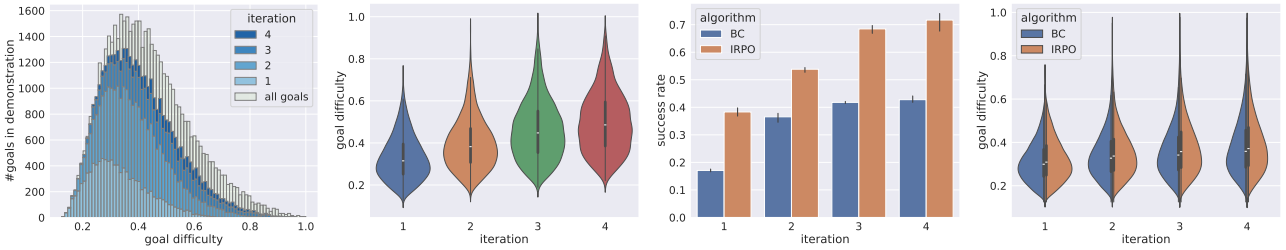
5.5. IRPO Improves Policy's Expected Return Progressively

To illustrate the effectiveness of IRPO, we present an analysis of the quantity and quality of demonstrations, as well as the policy performance during iterations of attitude control tasks in Fig.3. Fig. 3a shows that demonstrations progressively cover an increasing number of goals as IRPO iterates. Fig. 3b shows that demonstrations progressively encompass more challenging goals as IRPO iterates. Fig. 3c shows that both offline and online-learned policies exhibit an increasing capability to achieve more goals. Additionally, online learning contributes to the policy's capability to achieve more goals within a single iteration. Fig. 3d shows that both offline and online-learned policies exhibit an improved capability to achieve increasingly challenging goals. Additionally, online learning contributes to the policy's capability to achieve more difficult goals within a single iteration.

To summarize, Fig. 3a and 3b suggest that IRPO can progressively improve demonstration quality as training iterates. Higher demonstration quality helps to mitigate: (1) The over-constrained exploration problem. The data illustrated in the blue charts of Fig. 3c and 3d indicates that a higher quality of demonstrations contributes to obtaining a higher-performance offline learning policy. According to Theorem 4.2, improving $J(\pi_k^0)$ leads to an enhancement in $J(\pi_k^*)$. This relationship is also evident in the orange charts of Fig. 3c and 3d, displaying a similar upward trend with the blue ones. (2) The primacy bias problem. Fig. 2b illustrates that when online learning fine-tunes the policy without KL regularization, the better the offline learning policy performs, the better the online learning policy performs. This observation indicates that high-quality demonstrations contribute to mitigating the primacy bias problem. Experiments on Reach show similar conclusions. The relevant results are listed in Appendix G.2.

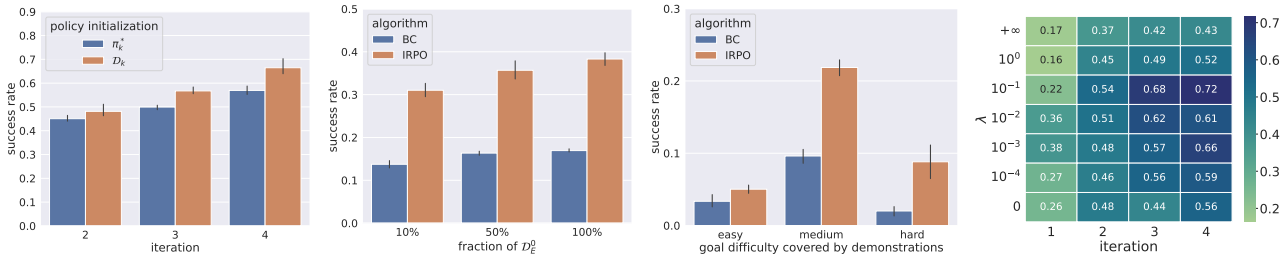
5.6. Ablation Study

In this section, we answer the following questions about ablation study: (1) Does iteration with rolling out data perform better than iteration with remaining policy? (2) How many demonstrations should be rolled out for the next iteration? (3) Which part of goal space should be focused on rolling out demonstrations for multi-goal problems? (4) How to set the strength of KL regularization for different



(a) histogram of goals on goal difficulty (b) distribution of goal difficulty of new-added trajectories (c) number of goals finished by BC and IRPO (d) distribution of difficulty of goals finished by BC and IRPO

Figure 3. Changes in the quantity and quality of demonstrations and performance of policies during iterations. The columns labeled "all goals" in the leftmost graph refer to the distribution of all goals discretized with the method detailed in C.3.5 over goal difficulty. The goal difficulty is detailed in Appendix C.3.4. Results come from experiments on attitude control over 5 random seeds.



(a) Success rate of policies with different remains for the next iteration. (b) Success rate of policies with different quantity of demonstrations. (c) Success rate of policies with demonstrations from different part of goal distribution. (d) Success rate of policies with different strength of KL regularizations in different iterations.

Figure 4. Ablation studies of IRPO. Results come from experiments on attitude control over 5 random seeds.

iterations? (5) How does the indicator function influence the performance? (6) How well does IRPO perform on easily accessible imperfect demonstrations, such as human play data?

Data is more valuable than policy for the next training iteration. To better understand how IRPO raises the upper bound of policy optimization through iterative training or what should remain for the next iteration, we design the following experiments: (1) Remain π_k^* for the next iteration. Specifically, we initialize π_{k+1} with π_k^*, then fine-tune π_{k+1} by RL with KL regularization $D_{KL}(\pi_{k+1}, \pi_k^*)$. (2) Remain $\mathcal{D}_k$ for the next iteration. The training process is identical to that of IRPO. As Fig. 4a shows, iteration with rolling out data contributes to achieving higher policy returns. Although π_{k+1}^0 may exhibit lower performance than the remaining policy π_k^*, as indicated by Fig. 3c, it is crucial to recognize that π_{k+1}^0 is derived from higher-quality demonstrations. Consequently, it is less affected by the primacy bias problem, contributing to improved policy performance through online learning. This implies that the data rolled out by the policy learned online contributes more to enhancing policy returns than the policy itself.

Roll out as many demonstrations as possible. To assess the impact of the quantity of demonstrations on IRPO, we train policies using 10% and 50% of $\mathcal{D}_E$ and compare them with the policy trained using all $\mathcal{D}_E$. Figure 4b illustrates that increasing the quantity of demonstrations leads to better policy performance. This implies that, when employing IRPO, it is beneficial to sample as many demonstrations as possible in each iteration.

Roll out demonstrations in areas with dense goal distribution. To assess the impact of the distribution of goals in demonstrations on IRPO, we train policies with demonstrations on the three goal sets defined in Appendix. C.3.4. The results indicate that the policy trained with the medium goal set performs better than the easy and hard ones. Considering the fact that the majority of goals are distributed around medium difficulty (Fig. 3a), we conclude that demonstrations in areas with a dense goal distribution contribute to better performance in IRPO.

Relax the KL regularization when demonstration quality improves. To determine the appropriate strength of regularization in the iteration of IRPO, we train policies with various regularization strengths. Fig. 4d shows that for the first iteration, where demonstration quality is relatively low, a smaller regularization strength of 10^{-3} results in the best performance. Conversely, for the last three iterations, where demonstration quality is relatively high, a larger regulariza-

Table 3. Comparison between different indicator functions on the attitude control task. $smooth(\tau)$ indicates the smoothness of trajectory τ (Mysore et al., 2021), and $length(\tau)$ indicates the length of trajectory τ. The negative dense reward function described in Appendix C.3.3 is employed for all experiments. Results come from experiments over 5 random seeds.

$f(\tau)$	Demonstration (quantity ↑)	Demonstration (trajectory length ↓)	Demonstration (trajectory smooth ↓)	Policy (success rate ↑)	Policy (trajectory length ↓)	Policy (trajectory smooth ↓)
no $f(\tau)$	24924	193.61±128.29	7.23±5.53	0.43±0.01	285.32±131.95	17.12±16.61
$f(\tau) = -smooth(\tau)$	24924	132.07±60.37	**5.76±6.35**	0.42±0.05	**221.27±122.67**	**12.74±10.58**
$f(\tau) = -length(\tau)$	24924	**124.64±53.07**	7.42±6.38	**0.54±0.01**	224.88±120.59	40.94±36.61
reference: demonstrations and policy in 1st iteration	10184	281.83±149.48	2.11±2.21	0.38±0.02	223.14±131.06	11.01±11.74

Table 4. Performance of IRPO on human play data. Results come from experiments over 5 random seeds. Note: demonstrations in the 1st iteration come from human and PID, while the demonstrations used in the 2nd iteration are optimized by IRPO.

Data Source	Iteration	Demonstration (quantity ↑)	Demonstration (trajectory length ↓)	Policy (success rate ↑)
Human Play	1	613	143.71±23.91	0.29±0.02
Human Play	2	21014	133.96±51.57	0.36±0.03
PID	1	10184	281.83±149.48	0.38±0.02
PID	2	24924	124.64±53.07	0.54±0.01

tion strength of 10^{-1} results in the best performance. This suggests that the strength of KL regularization should be increased as the quality of demonstrations improves.

IRPO demonstrates robustness to the setting of indicator functions. To assess the impact of indicator function on IRPO, we train policies with different indicator functions and show results with Table 3. It is evident from the results that both not using an indicator function and using simple indicator functions, while differing in the extent to which they enhance the performance of the policy, still lead to a certain degree of improvement (We provide a detailed analysis of the experiments related to the indicator function in Appendix F). This suggests that IRPO can effectively enhance the quantity and quality of the demonstrations with a simple indicator function or even without an indicator function.

IRPO performs well on human play data. To showcase the performance of IRPO on easily accessible imperfect demonstrations, we collect human play data (with the detailed collection method documented in Appendix G.3.1), and compare IRPO's performance on human play data with that on PID-generated demonstrations. The results are presented in Table 4. As shown, despite the relatively small quantity of human play data, which constitute only about 6% of the PID demonstrations, they exhibit higher quality. The trajectories from human play are approximately 51% of the length of those from the PID controller. Despite this, IRPO is able to nearly optimize the demonstrations from human play to a level indistinguishable from that of the PID demonstrations by the second iteration. Specifically, the demonstration from human play is only 15.7% less than that from the PID, with trajectory lengths being only 7.5% longer. In terms of policy performance, the IRPO algorithm also demonstrates a consistent improvement in performance when trained on human play demonstrations.

6. Conclusion and Limitations

The paper introduces the Iterative Regularized Policy Optimization (IRPO) method for addressing challenges in IL arising from imperfect demonstrations. Theoretical analysis and experimental verification expose the limitations of conventional online fine-tuning with KL regularization when demonstrations are imperfect. Two major issues, namely over-constrained exploration and primacy bias, are identified as intrinsic issues associated with imperfect demonstrations. IRPO employs iterative training and demonstration boosting to enhance demonstration quality. Iterative training progressively improves the policy's expected return, while demonstration boosting involves rolling out demonstrations with the online policy to mitigate both over-constrained exploration and primacy bias problems. Theoretical support for IRPO's effectiveness is provided through Theorems and Lemmas, complemented by empirical evidence from experiments on diverse tasks.

Some limitations should be addressed in future work. Firstly, the design of the indicator function in the demonstration boosting mechanism relies on our understanding of the task. Although our experiments demonstrate that even without or with very simple indicator functions, the iterative training of IRPO is beneficial, it remains an open question whether a more carefully designed indicator function could lead to faster convergence and better-performing policies. Secondly, while we believe that IRPO is a general framework for RL training based on imperfect demonstrations, our experimental validation has been limited to on-policy RL. It is still unclear whether IRPO is applicable to off-policy RL settings.

Acknowledgements

This work was supported by the National Key R&D Program of China (No. 2021ZD0112904).

Impact Statement

IRPO, integrating imitation learning and reinforcement learning, adeptly tackles challenges presented by imperfect data, extending the applicability of both paradigms. Furthermore, IRPO emerges as an efficient learning methodology, demonstrating proficiency in acquiring high-quality policies for complex problems. It is essential to underscore that IRPO enhances data quality iteratively throughout its training process, underscoring the paramount importance of data in the machine learning domain. Nevertheless, it is important to note that the data sampled by IRPO may not align perfectly with human demonstrations, introducing a potential risk of misuse.

References

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., and Bellemare, M. Reincarnating reinforcement learning: Reusing prior computation to accelerate progress. *Advances in Neural Information Processing Systems*, 35: 28955–28971, 2022.

Baker, B., Akkaya, I., Zhokov, P., Huizinga, J., Tang, J., Ecoffet, A., Houghton, B., Sampedro, R., and Clune, J. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35:24639–24654, 2022.

Ball, P. J., Smith, L., Kostrikov, I., and Levine, S. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, 2023.

Belkhale, S., Cui, Y., and Sadigh, D. Data quality in imitation learning. *Advances in neural information processing systems*, 2023.

Brown, D., Goo, W., Nagarajan, P., and Niekum, S. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pp. 783–792. PMLR, 2019.

Brown, D. S., Goo, W., and Niekum, S. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pp. 330–359. PMLR, 2020.

Fedotov, A. A., Harremoës, P., and Topsoe, F. Refinements of pinsker's inequality. *IEEE Transactions on Information Theory*, 49(6):1491–1498, 2003.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Gallouédec, Q., Cazin, N., Dellandréa, E., and Chen, L. panda-gym: Open-source goal-conditioned environments for robotic learning. *4th Robot Learning Workshop: Self-Supervised and Lifelong Learning at NeurIPS*, 2021.

Gleave, A., Taufeeque, M., Rocamonde, J., Jenner, E., Wang, S. H., Toyer, S., Ernestus, M., Belrose, N., Emmons, S., and Russell, S. imitation: Clean imitation learning implementations. arXiv:2211.11972v1 [cs.LG], 2022. URL https://arxiv.org/abs/2211.11972.

Grollman, D. H. and Billard, A. G. Robot learning from failed demonstrations. *International Journal of Social Robotics*, 4:331–342, 2012.

Hedlund-Botti, E. and Gombolay, M. C. Investigating learning from demonstration in imperfect and real world scenarios. In *Companion of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 769–771, 2023.

Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Horgan, D., Quan, J., Sendonaris, A., Osband, I., et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Ho, J. and Ermon, S. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Huang, Z., Wu, J., and Lv, C. Efficient deep reinforcement learning with imitative expert priors for autonomous driving. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward learning from human preferences and demonstrations in atari. *Advances in neural information processing systems*, 31, 2018.

Jing, M., Ma, X., Huang, W., Sun, F., Yang, C., Fang, B., and Liu, H. Reinforcement learning from imperfect demonstrations under soft expert guidance. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5109–5116, 2020.

Kim, B., Farahmand, A.-m., Pineau, J., and Precup, D. Learning from limited demonstrations. *Advances in Neural Information Processing Systems*, 26, 2013.

Li, J., Hu, X., Xu, H., Liu, J., Zhan, X., and Zhang, Y.-Q. Proto: Iterative policy regularized offline-to-online reinforcement learning. *arXiv preprint arXiv:2305.15669*, 2023.

Marshall, P. H. and Werder, P. R. The effects of the elimination of rehearsal on primacy and recency. *Journal of Verbal Learning and Verbal Behavior*, 11(5):649–653, 1972.

Mysore, S., Mabsout, B., Mancuso, R., and Saenko, K. Regularizing action policies for smooth control with reinforcement learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1810–1816. IEEE, 2021.

Nikishin, E., Schwarzer, M., D'Oro, P., Bacon, P.-L., and Courville, A. The primacy bias in deep reinforcement learning. In *International conference on machine learning*, pp. 16828–16847. PMLR, 2022.

Oh, H., Sasaki, H., Michael, B., and Matsubara, T. Bayesian disturbance injection: Robust imitation learning of flexible policies for robot manipulation. *Neural Networks*, 158:42–58, 2023.

Pomerleau, D. A. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., and Dormann, N. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html.

Ramrakhya, R., Batra, D., Wijmans, E., and Das, A. Pirlnav: Pretraining with imitation and rl finetuning for objectnav. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17896–17906, 2023.

Rohatgi, V. K. and Saleh, A. M. E. *An introduction to probability and statistics*. John Wiley & Sons, 2015.

Rudner, T. G., Lu, C., Osborne, M. A., Gal, Y., and Teh, Y. On pathologies in kl-regularized reinforcement learning from expert demonstrations. *Advances in Neural Information Processing Systems*, 34:28376–28389, 2021.

Sasaki, F. and Yamashina, R. Behavioral cloning from noisy demonstrations. In *International Conference on Learning Representations*, 2020.

Schmitt, S., Hudson, J. J., Zidek, A., Osindero, S., Doersch, C., Czarnecki, W. M., Leibo, J. Z., Kuttler, H., Zisserman, A., Simonyan, K., et al. Kickstarting deep reinforcement learning. *arXiv preprint arXiv:1803.03835*, 2018.

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

Shenfeld, I., Hong, Z.-W., Tamar, A., and Agrawal, P. Tgrl: An algorithm for teacher guided reinforcement learning. In *International Conference on Machine Learning*, pp. 31077–31093. PMLR, 2023.

Shukla, D., Keshmiri, S., and Beckage, N. Imitation learning for neural network autopilot in fixed-wing unmanned aerial systems. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1508–1517. IEEE, 2020.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Tsurumine, Y. and Matsubara, T. Goal-aware generative adversarial imitation learning from imperfect demonstration for robotic cloth manipulation. *Robotics and Autonomous Systems*, 158:104264, 2022.

Tunyasuvunakool, S., Muldal, A., Doron, Y., Liu, S., Bohez, S., Merel, J., Erez, T., Lillicrap, T., Heess, N., and Tassa, Y. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

Vieillard, N., Kozuno, T., Scherrer, B., Pietquin, O., Munos, R., and Geist, M. Leverage the average: an analysis of kl regularization in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:12163–12174, 2020.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wang, C., Fan, L., Sun, J., Zhang, R., Fei-Fei, L., Xu, D., Zhu, Y., and Anandkumar, A. Mimicplay: Long-horizon imitation learning by watching human play. In *Conference on Robot Learning*, pp. 201–221. PMLR, 2023a.

Wang, S., Yang, Q., Gao, J., Lin, M. G., CHEN, H., Wu, L., Jia, N., Song, S., and Huang, G. Train once, get a family: State-adaptive balances for offline-to-online reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b.

Wang, X., Song, J., Qi, P., Peng, P., Tang, Z., Zhang, W., Li, W., Pi, X., He, J., Gao, C., et al. Scc: An efficient deep reinforcement learning agent mastering the game of starcraft ii. In *International conference on machine learning*, pp. 10905–10915. PMLR, 2021a.

Wang, Y., Xu, C., Du, B., and Lee, H. Learning to weight imperfect demonstrations. In *International Conference on Machine Learning*, pp. 10961–10970. PMLR, 2021b.

Wu, Y.-H., Charoenphakdee, N., Bao, H., Tangkaratt, V., and Sugiyama, M. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pp. 6818–6827. PMLR, 2019.

Xu, H., Zhan, X., Yin, H., and Qin, H. Discriminator-weighted offline imitation learning from suboptimal demonstrations. In *International Conference on Machine Learning*, pp. 24725–24742. PMLR, 2022.

Zheng, B., Verma, S., Zhou, J., Tsang, I. W., and Chen, F. Imitation learning: Progress, taxonomies and challenges. *IEEE Transactions on Neural Networks and Learning Systems*, (99):1–16, 2022.

A. Proof of Theorem 4.2

Proof.

$$
\begin{aligned}
&|J(\pi) - J(\pi_E)| \\
&= \left| \sum_s d_\pi(s) r(s) - \sum_s d_{\pi_E}(s) r(s) \right| \\
&\leq \max_s |r(s)| \sum_s |d_\pi(s) - d_{\pi_E}(s)| \\
&= \max_s |r(s)| \cdot \|d_\pi - d_{\pi_E}\|_1 \\
&\leq \max_s |r(s)| \sqrt{2 D_{KL}(d_\pi, d_{\pi_E})} \\
&\leq \max_s |r(s)| \sqrt{2 \frac{1}{H} \sum_{t=0}^{H-1} (H-t) D_{KL}^{s \sim d_\pi^t} \big(\pi(\cdot|s), \pi_E(\cdot|s)\big)} \\
&\leq \max_s |r(s)| \sqrt{2 \frac{1}{H} \sum_{t=0}^{H-1} H D_{KL}^{s \sim d_\pi^t} \big(\pi(\cdot|s), \pi_E(\cdot|s)\big)} \\
&= \max_s |r(s)| \sqrt{2H D_{KL}^{s \sim d_\pi} (\pi(\cdot|s), \pi_E(\cdot|s))}
\end{aligned}
$$

□

In the above proof, Pinsker's inequality is utilized for the second inequality, and Lemma 4.1 is applied for the third. It is noteworthy that the provided proof is applicable to cases where the expected return is computed either as the expected sum of rewards or the expected sum of discounted rewards, with only variations in constants.

B. Proof of Theorem 4.3

We begin by presenting Lemma B.1 (Li et al., 2023), which provides the performance bound of iterative policy regularization.

Lemma B.1 ((Li et al., 2023)). *Define Q_k as the action-value of policy π_k obtained the k^{th} iteration by optimizing the Objective:*

$$
\pi_k \leftarrow \arg\max_\pi \mathbb{E}\left[\sum_{t=0}^{H-1} \gamma^t \left(r(s_t, a_t) - \lambda \log \left(\frac{\pi(a_t|s_t)}{\pi_{k-1}(a_t|s_t)} \right) \right) \right], \tag{8}
$$

π_0 as the pretrained policy and Q_0 the corresponding action-value. Let $v_{max}^\lambda := \frac{r_{max} + \lambda \ln |\mathcal{A}|}{1-\gamma}$, $v_{max} := v_{max}^0$, ϵ_j is the approximation error of value function at j^{th} iteration, then:

$$
\|Q^* - Q_k\|_\infty \leq \frac{2}{1-\gamma} \left\| \frac{1}{k+1} \sum_{j=0}^{k} \epsilon_j \right\| + \frac{4}{1-\gamma} \frac{v_{max}^\lambda}{k+1}, k \in N.
$$

Then we give the proof of Theorem 4.3.

Proof. Suppose the optimal policy π^* and the corresponding action-value Q^*, the initial state s_0, then the optimal return is $J(\pi^*) = Q^*(s_0, a^*)$, where $a^* = \pi^*(s_0) = \arg\max_a Q^*(s_0, a)$. For the policy π_k trained with Eq. 8 in the k^{th} iteration and a prediction error Δ, we discuss the difference in $J(\pi_k)$ and $J(\pi^*)$:

(1) $\forall a \in \mathcal{A} \setminus a^*$, if $Q_k(s_0, a) = Q^*(s_0, a) + \Delta \leq Q^*(s_0, a^*) - \Delta = Q_k(s_0, a^*)$, then π_k will select a^* and obtains the return $Q^*(s_0, a^*)$, which means $J(\pi_k) = J(\pi^*)$;

(2) $\exists a \in \mathcal{A} \setminus a^*$, if $Q_k(s_0, a) = Q^*(s_0, a) + \Delta \geq Q^*(s_0, a^*) - \Delta = Q_k(s_0, a^*)$, then π_k will select a and obtains the return $Q^*(s_0, a)$, and $|J(\pi_k) - J(\pi^*)| = |Q^*(s_0, a) - Q*(s_0, a^*)| \leq 2\Delta$.

The prediction error is bounded: $\Delta \leq \|Q^* - Q_k\|_\infty$. With Lemma B.1, we have:

$$|J(\pi_k) - J(\pi^*)| \leq \frac{4}{1-\gamma}\left\|\frac{1}{k+1}\sum_{j=0}^{k}\epsilon_j\right\| + \frac{8}{1-\gamma}\frac{v_{max}^{\lambda}}{k+1}, k \in N.$$

□

C. Experimental Tasks

C.1. Halfcheetah and Hopper

We employ Halfcheetah-medium-v2 and Hopper-medium-v2, as defined by D4rl. Both tasks fall under finite MDPs with a maximum step limitation of 1000.

C.1.1. Reward Function

The reward function of Halfcheetah-medium-v2 is defined as $reward = forward\_reward - ctrl\_cost$, where $forward\_reward = forward\_reward\_weight * (x\text{-}coordinate\ before\ action - x\text{-}coordinate\ after\ action)/dt$ and $ctrl\_cost = ctrl\_cost\_weight * sum(action^2)$.

The reward function of Hopper-medium-v2 is defined as $reward = healthy\_reward + forward\_reward - ctrl\_cost$, where $healthy\_reward$ is a scalar defined by the hopper's state and $forward\_reward$ and $ctrl\_cost$ is the same as Halfcheetah-medium-v2.

The reward functions of Halfcheetah and Hopper depend mainly on states and partially on the gain of action to penalize the agent if it takes too large actions. As the penalties are usually at least one order of magnitude smaller than rewards based on states, the reward functions of Halfcheetah and Hopper substantially satisfy the applicability of Theorem 4.2. We experiment on Halfcheetah and Hopper to demonstrate that Theorem 4.2 empirically applies to these reward functions.

C.1.2. Demonstrations

The corresponding datasets of Halfcheetah-medium-v2 and Hopper-medium-v2 are imperfect as they are sampled by: first training a policy online using Soft Actor-Critic (SAC), early-stopping the training, and collecting 1M samples from this partially-trained policy (Fu et al., 2020). The demonstrator policy's performance is about one-third of that of a well-trained SAC policy (Fu et al., 2020).

C.2. The Reach Task

We use the end-effector displacement control mode, in which the action corresponds to the displacement of the end-effector, provided by panda-gym.

C.2.1. Reward Function

The reward function of Reach is defined as $r = 0$ if reaches the goal, else $r = -1$. It's a kind of sparse reward that encourages the agent to reach the goal as quickly as possible. It depends only on states, satisfying the applicability of Theorem 4.2.

C.2.2. Termination Condition

The definition of the Reach task is the same as panda-gym except that: (1) a more precise termination condition is used: the parameter *goal_range* is changed from 0.3 to 0.01. (2) a larger goal space is used: the parameter *distance_threshold* is changed from 0.05 to 0.3. Besides, the task has a max step limitation of 50. This version of Reach is more difficult for learning policy.

Table 5. Parameters used to discretize the goal space

(a) Reach

	min	max	Δ
x	-0.15	0.15	0.02
y	-0.15	0.15	0.02
z	0	0.3	0.02
#goals: 4096			

(b) Attitude control

	min	max	Δ
v	100	300	10
μ	-85	85	5
χ	-170	170	5
#goals: 50715			

C.2.3. DEMONSTRATIONS

The goal space is discretized with parameters listed in Table 5a. Given the desired position (d_x, d_y, d_z) and the current position (x, y, z) of the end-effector, the action (a_x, a_y, a_z) is calculated by the following PID controller

$$\begin{cases} a_x = & \eta_x(d_x - x) \\ a_y = & \eta_y(d_y - y) \\ a_z = & \eta_z(d_z - z) \end{cases} \tag{9}$$

where η_x, η_y, η_z are the coefficient of the proportional corresponding to x, y, z. In experiment, we set η_x, η_y, η_z all to 1.5.

The above PID controller samples 2621 trajectories from 4096 goals with an average length of 43.03. These 2621 successful trajectories form the $\mathcal{D}_E$. $\mathcal{D}_E$ are imperfect: firstly, the quantity of demonstrations is limited with only about 64% goals successfully sampled; secondly, the quality of demonstrations is low as a well-trained policy can usually finish the goal within 5 steps.

C.3. The Fixed-Wing UAV Attitude Control Environment

The fixed-wing UAV's attitude control task is to target its velocity vector to a target velocity vector.

C.3.1. STATE AND ACTION SPACE

The state consists of pitch angle θ, roll angle ϕ, yaw angle ψ, flight path azimuth angle χ, flight path elevator angle μ, altitude h, roll angular velocity p, true airspeed v, and goal (v_g, μ_g, χ_g). The action consists of ail, ele, rud, pla, which denotes the actuator position of the aileron, elevator, rudder, and power level actuator.

C.3.2. TRANSITION

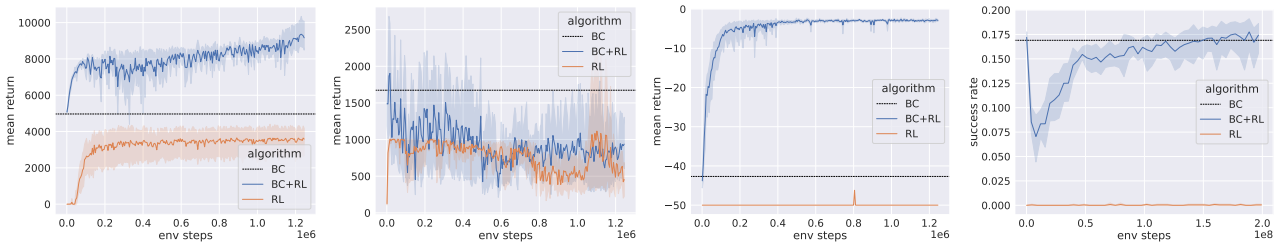
The action (ail, ele, rud, pla) is sent to the Flight Dynamics Model (FDM) to get the next state with the F-16 model. The episode terminates when triggers one of the following two conditions: (1) if v, μ, χ is close to (v_g, μ_g, χ_g) within error of e_v, e_μ, e_χ; (2) if does not trigger the first condition for T_{max} steps.

C.3.3. REWARD FUNCTION

The reward function is designed as

$$r_{g,t} = -(w_v \frac{\|\vec{v}_t - \vec{v}_g\|_v}{\sigma_v} + w_d \frac{\|\vec{v}_t - \vec{v}_g\|_d}{\sigma_d}), \tag{10}$$

where $w_v \in [0, 1], w_d \in [0, 1], w_v + w_d = 1.0$ are weight factors for velocity and direction, σ_v, σ_d are scaling factors for velocity and direction, $\|.\|_v$ calculates the difference in modulus of two velocity vectors, and $\|.\|_d$ calculates the difference in direction of two velocity vectors. The above reward function depends only on states, satisfying the applicability of Theorem 4.2.



(a) Halfcheetah (b) Hopper (c) Reach (d) Attitude control

Figure 5. Comparison among BC, RL, and BC+RL on the four tasks.

Table 6. Characteristics of the four experimental tasks.

Task	Task goal	Transition fluctuation
Halfcheetah	single-goal	small
Hopper	single-goal	large
Reach	multi-goal	small
Attitude control	multi-goal	large

C.3.4. Goal Difficulty

In order to evaluate the quality of demonstrations in the following sections, we introduce the goal difficulty

$$d_v(g, v_0) = \alpha_v + (1 - \alpha_v)\frac{|v_g - v_0|}{|v_{max} - v_{min}|}, \tag{11}$$

where $\alpha_v \in [0, 1)$ is a base value of difficulty for v. And the same is for $d_\mu(g, \mu_0)$ and $d_\chi(g, \chi_0)$. Consequently, the difficulty of the goal is defined as $d(g, v_0, \mu_0, \chi_0) = d_v(g, v_0) \cdot d_\mu(g, \mu_0) \cdot d_\chi(g, \chi_0)$, which describes the magnitude of changes in the UAV's state variables.

Based on Eq. 11, we sort all goals based on their difficulty and define the following three goal sets: the easy goal set, comprising the 100 simplest goals; the medium goal set, consisting of goals with difficulty values ranked between 3000 and 3100; and the difficult goal set, comprising goals with difficulty values ranked between 7000 and 7100.

C.3.5. Demonstrations

A PID controller is used to sample trajectories. For convenience, the goal space is discretized with parameters listed in Table 5b. Of the 50715 discretized goals, 10184 trajectories are successfully sampled with an average length of 282.01. These 10184 successful trajectories form the $\mathcal{D}_E$. $\mathcal{D}_E$ are imperfect: firstly, the quantity of demonstrations is limited with only about 20% goals successfully sampled; secondly, the quality of demonstrations is low as a well-trained policy can usually finish the goal within 150 steps.

C.3.6. Comparison about Difficulty in Transition from Offline to Online on the Four Tasks

To show the differences of the four tasks, we compare BC, RL, and BC+RL (train an imitation policy with BC first, then fine-tune this policy with RL without regularization) and show the results in Fig. 5. Note that the degree of fluctuation in the transition from offline to online is determined by various factors, including environment dynamics, demonstrations, offline learning objectives, online learning objectives, and so on. In our settings, the four tasks have different training process. For the two single-goal problems, the return of policy can be directly improved by RL on Halfcheetah but suffers a drop and can never be recovered to the original level on Hopper. This suggests that Hopper is more challenging than Halfcheetah in our settings when transitioning from offline to online. For the two multi-goal problems, the return of policy can be directly improved by RL on Reach but suffers a drop and can be recovered to the original level through a long time of training on the attitude control task. This suggests that attitude control is more challenging than Reach in our settings when transitioning from offline to online.

Table. 6 summarizes the characteristics of the four tasks in our settings: (1) Halfcheetah represents single-goal and easy-

Table 7. Parameters used in BC

(a) Halfcheetah & Hopper

Parameter	Value
l2_weight	10^{-4}
ent_weight	10^{-2}
batch_size	256
epochs	100

(b) Reach

Parameter	Value
l2_weight	10^{-4}
ent_weight	10^{-2}
batch_size	256
epochs	10

(c) Attitude control

Parameter	Value
l2_weight	0
ent_weight	10^{-2}
batch_size	4096
epochs	300

Table 8. Parameters used in PPO

(a) Halfcheetah & Hopper

Parameter	Value
ent_coef	10^{-4}
gamma	0.98
gae_lambda	0.92
lr	10^{-4}
batch_size	64
train_steps	1.25×10^6
rollout_process_num	1
n_steps	256
n_epochs	5
use_sde	True
normalize_advantage	True

(b) Reach

Parameter	Value
ent_coef	10^{-4}
gamma	0.98
gae_lambda	0.92
lr	10^{-4}
batch_size	64
train_steps	1.25×10^6
rollout_process_num	1
n_steps	256
n_epochs	5
use_sde	True
normalize_advantage	True

(c) Attitude control

Parameter	Value
ent_coef	10^{-2}
gamma	0.995
gae_lambda	0.95
lr	10^{-4}
batch_size	4096
train_steps	5×10^8
rollout_process_num	64
n_steps	2048
n_epochs	5
use_sde	True
normalize_advantage	True

transition tasks that are easy to optimize without a drop or can be easily recovered from a drop when transitioning from offline to online learning; (2) Hopper represents single-goal and hard-transition tasks that suffer a drop and are hard to recover from a drop when transitioning from offline to online learning; (3) Reach represents multi-goal and easy-transition tasks with a similar optimization process as Halfcheetah; (4) Attitude control represents multi-goal and hard-transition tasks with a similar optimization process as Hopper. We experiment with these four different tasks to show the broad applicability of IRPO.

D. Implementation Details

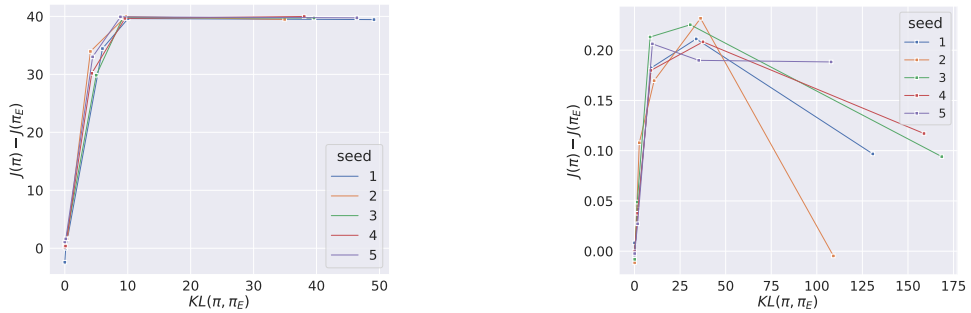
Behavioral Cloning (BC) (Pomerleau, 1991) is employed for offline learning and Proximal Policy Optimization (PPO) (Schulman et al., 2017) for online learning. The Imitation framework (Gleave et al., 2022) is utilized to implement the BC algorithm, with parameters detailed in Table 7, while the Stable Baselines3 framework (Raffin et al., 2021) is used for PPO with parameters listed in Table 8. A fully connected network of size 256×256 is employed for Halfcheetah, Hopper, and Reach tasks. For attitude control, a 128×128 fully connected network is used for the first and second training iterations, and a larger architecture of $256 \times 256 \times 128 \times 128 \times 64$ for the third and fourth iterations. The Tanh activation function is applied throughout all training processes.

As offline imitation learning only learns a policy network, we add a warm-up for the value network at the beginning of online learning (Wang et al., 2021a; Ramrakhya et al., 2023). When the learning transitions from offline to online, we first freeze the policy parameter and train the value network with online samples until it converges, then proceed with the normal RL training. For the parameter λ, we use 10^{-1} for all the iterations for Halfcheetah, Hopper, and Reach, 10^{-3} for the first iteration of Attitude control, and 10^{-1} for the last three iterations.

E. An Intuitive Explanation of Theorem 4.2

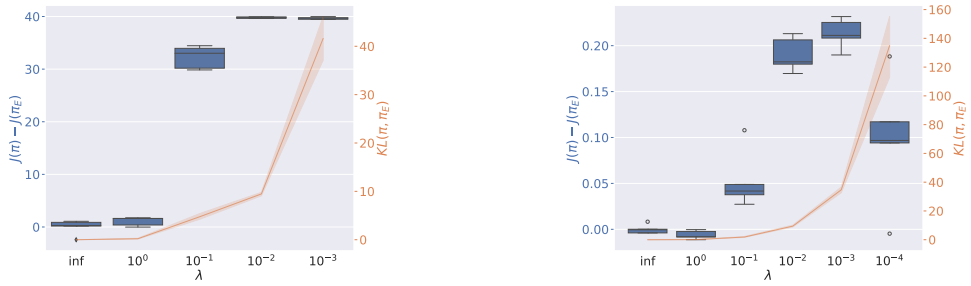
In this section, we present two sets of figures to better illustrate the conclusions related to Theorem 4.2.

In our scenario, as the expert policy π_E is imperfect, we aim for $J(\pi)$ to outperform $J(\pi_E)$ as much as possible. Theorem 4.2



(a) Reach (b) Attitude Control

Figure 6. Relationship between $KL(\pi, \pi_0)$, and $J(\pi) - J(\pi_E)$.



(a) Reach (b) Attitude Control

Figure 7. Relationship between λ, $KL(\pi, \pi_0)$, and $J(\pi) - J(\pi_E)$. Results come from experiments over 5 random seeds.

indicates that the KL divergence $D_{KL}(\pi, \pi_E)$ will constrain the improvement in $J(\pi)$ over $J(\pi_E)$, with a maximum improvement of $\max_s |r(s)| \sqrt{2H D_{KL}^{s \sim d_\pi}(\pi(\cdot|s), \pi_E(\cdot|s))}$.

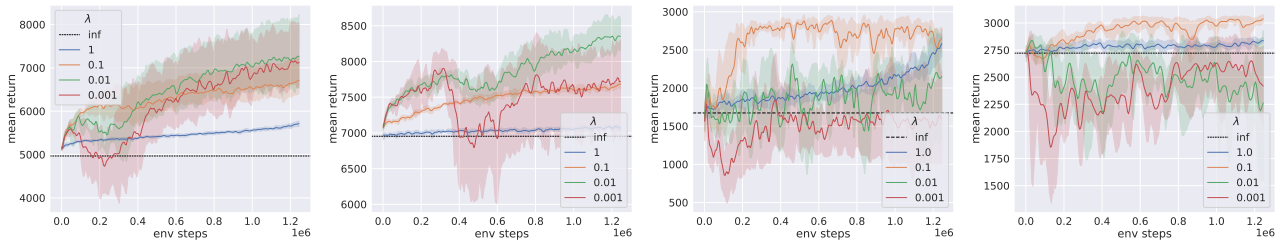
Firstly, we employ line graphs to directly depict the relationship between $J(\pi) - J(\pi_E)$ and $D_{KL}(\pi, \pi_E)$. Fig. 6a presents the results on Reach. It is evident that as the $D_{KL}(\pi, \pi_E)$ increases, the improvement in $J(\pi)$ relative to $J(\pi_E)$ becomes more pronounced. Secondly, we utilize a combined graph, as depicted in Fig. 7a, where the KL regularization strength λ is used as the abscissa. The box graph illustrates the relationship between λ and $J(\pi) - J(\pi_E)$, while the line graph shows the relationship between λ and $D_{KL}(\pi, \pi_E)$. The result shows that as λ decreases, $D_{KL}(\pi, \pi_E)$ increases, and the improvement in $J(\pi)$ relative to $J(\pi_E)$ also increases correspondingly. In summary, the above experimental results corroborate the validity of the relationship expressed in Theorem 4.2 between $J(\pi) - J(\pi_E)$ and $D_{KL}(\pi, \pi_E)$.

Furthermore, we present results on Attitude Control in Fig. 6b and 7b. There is a slight discrepancy in the results compared to the Reach task. As the KL divergence $D_{KL}(\pi, \pi_E)$ increases, the improvement in $J(\pi)$ relative to $J(\pi_E)$ initially increases and then diminishes. The experimental analysis is consistent with the last paragraph of Section 5.3.

F. Discussion on Indicator Function

The design of $f(\tau)$ is aimed at consistently enhancing the quality of demonstrations, thereby addressing the over-constrained exploration and primacy bias problems that arise from imperfect demonstrations. In our experiments, for the two single-goal tasks, Halfcheetah and Hopper, we directly replace the demonstrations without employing $f(\tau)$. For the two multi-goal tasks, Reach and Attitude Control, we use the opposite number of trajectory length as the $f(\tau)$. Below, we offer a comprehensive explanation of the points discussed above.

Firstly, **the design of $f(\tau)$ only influences multi-goal tasks**. For single-goal tasks represented by Halfcheetah and Hopper, if we optimize a policy from π to π^* such that the expected return satisfies $J(\pi^*) > J(\pi)$, then under the same conditions for generating demonstrations, trajectories produced by π^* will have a higher expected return and are thus better suited as new demonstrations. For multi-goal tasks like Reach and Attitude Control, although the expected return may satisfy



(a) Halfcheetah, iteration=1 (b) Halfcheetah, iteration=2 (c) Hopper, iteration=1 (d) Hopper, iteration=2

Figure 8. Training process with different strength of regularizations on Halfcheetah and Hopper. The curves are smoothed by the gaussian_filter1d function from scipy with simga=5. Results come from experiments over 5 random seeds.

$E_g[J(\pi^*)] > E_g[J(\pi)]$, for a specific goal $\hat{g}$, the trajectory $\tau^*_{\hat{g}}$ produced by π^* may not be better than the trajectory $\tau_{\hat{g}}$ produced by π. Therefore, the indicator function $f(\tau)$ primarily plays a role in multi-goal tasks.

Secondly, $f(\tau)$ **does not impact the quantity of demonstrations** and only affects the quality of demonstrations, as trajectories that can complete new goals should always be added to demonstrations. Therefore, even if it is challenging to design a reasonable $f(\tau)$ in conjunction with the task, IRPO can still benefit from the increased number of demonstrations without $f(\tau)$, and the iterative training will obtain policies with continuously improved performance.

Thirdly, **a simple design for $f(\tau)$ can effectively assist IRPO in iterative training**. In our experiments, since the demonstrations consist of only state-action pairs without rewards, we employ the opposite number of trajectory length as $f(\tau)$ (shorter trajectories receive higher scores). Although this is not directly aligned with the 0-1 sparse rewards and the dense rewards in the range of [-1, 0] used in our experiments, it implicitly expresses the same optimization objective: to achieve the goal as quickly as possible.

Lastly, we **conduct experiments to support the above points** on the challenging multi-goal task, Attitude Control, comparing the performance with (1) no $f(\tau)$, (2) $f(\tau) = -length(\tau)$, and (3) $f(\tau) = -smooth(\tau)$ (where a smaller value of $smooth(\tau)$ indicates a smoother trajectory, with the negative dense reward function as described in Appendix C.3.3). The results are presented in Table 3.

From the results obtained, it can be observed that: (1) The quantity of demonstrations in all three sets of experiments is the same, all increased from 10184 to 24924, confirming that $f(\tau)$ does not affect the quantity of demonstrations as we have stated. (2) When $f(\tau)$ is not used, the average length of the demonstrations is the longest, and the smoothness is almost identical to that with $f(\tau) = -length(\tau)$. This indicates that the quality of the demonstrations is the poorest among the three groups of experiments. Despite this, IRPO is still able to improve the policy's success rate through iterative training, increasing it from 0.38 to 0.43. (3) For $f(\tau) = -smooth(\tau)$, as $f(\tau)$ pursues a goal that is not consistent with the reward function, while it significantly improves the smoothness of the demonstrations, the resulting policy has a limited increase in success rate. (4) For $f(\tau) = -length(\tau)$, as $f(\tau)$ pursues a goal that is consistent with the reward function, the resulting policy has the highest success rate.

In summary, although the design of $f(\tau)$ is expected to align with the reward function, the analysis and experimental results indicate that even without using $f(\tau)$ or with a very simple $f(\tau)$, IRPO can effectively enhance the quantity and quality of the demonstrations, leading to the capability of learning well-performing policies.

G. More Results

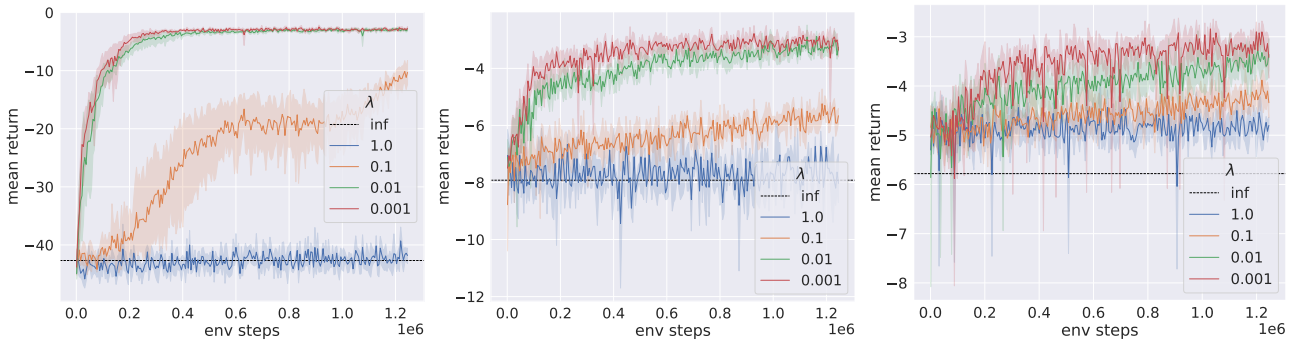
G.1. Results on MuJoCo

Fig. 8 shows the training curve of IRPO with various strengths of KL regularizations in different iterations on Halfcheetah and Hopper.

G.2. Results on Panda-Gym

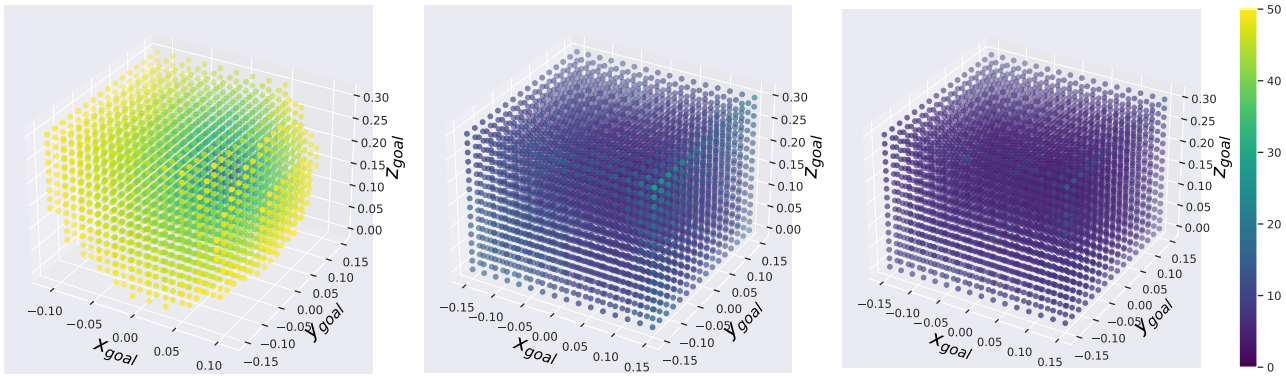
Fig. 9 shows the training curve of IRPO with various strengths of KL regularizations in different iterations on Reach.

Fig. 10 illustrates the progressive expansion of demonstration coverage over training iterations. The increasing number of



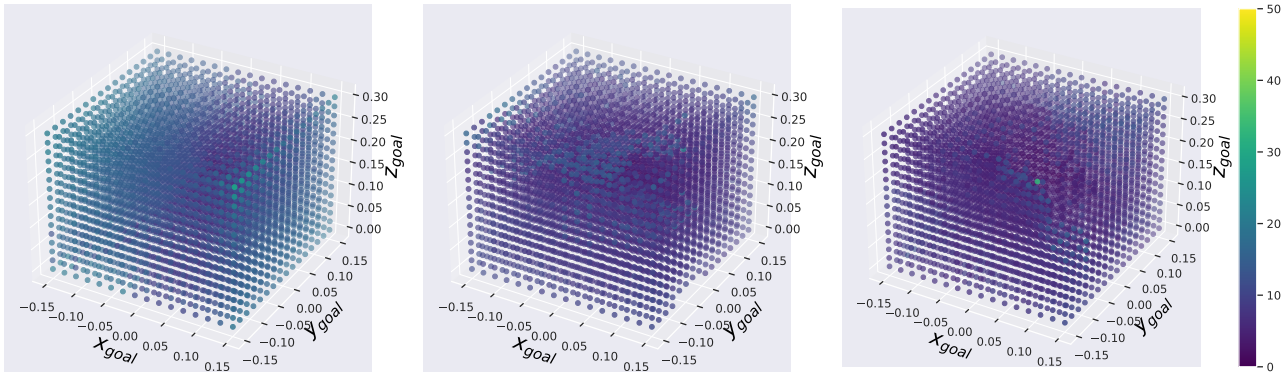
(a) Reach, iteration=1 (b) Reach, iteration=2 (c) Reach, iteration=3

Figure 9. Training process with different strength of regularizations on Reach. Results come from experiments over 5 random seeds.



(a) $\mathcal{D}_E$ (b) $\mathcal{D}_1$ (c) $\mathcal{D}_2$

Figure 10. Goal space coverage of demonstrations in Reach.



(a) π_1^*(IRPO, iteration=1) (b) π_2^*(IRPO, iteration=2) (c) π_3^*(IRPO, iteration=3)

Figure 11. Goal space coverage of policies in Reach.

points reflects the expanding range of goals covered by demonstrations as IRPO iterates. Notably, the deepening color of the points signifies shorter trajectories, indicative of improving data quality throughout the training process.

Fig. 11 shows the performance of the policy during training iterations. The policies from the initial three iterations demonstrate the ability to complete nearly all goals. However, as training progresses, the color of the points deepens, indicating that the policies achieve faster task completion, showcasing the iterative improvement facilitated by IRPO.

(a) iteration = 1 (b) iteration = 2

Figure 12. Training process with different strengths of regularizations on attitude control. The curves are smoothed by the gaussian_filter1d function from scipy with simga=5. Results come from experiments over 5 random seeds. Note that before smoothing, all curves share the same starting point, which is the performance of the pre-trained policy, also referred to as the left endpoint of the 'inf' curve.

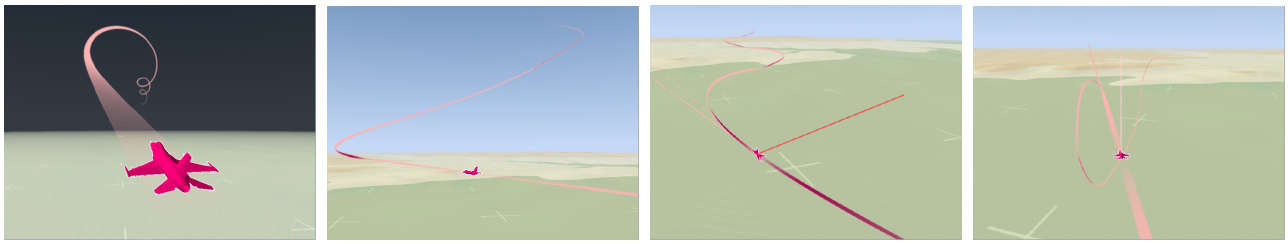


Figure 13. Four trajectories from human play data. Results come from experiments over 5 random seeds.

G.3. Results on Fixed-wing Environment

Fig. 12 shows the training curve of IRPO with various strengths of KL regularizations in different iterations on Attitude Control.

G.3.1. Experiments on Human Play Data

We provide further experiment results on human demonstrations below. We collecte demonstrations from human experts, totaling **18** trajectories. It is important to emphasize that these trajectories are not generated with the specific purpose of completing an Attitude Control task. Instead, they represent "play data" (Wang et al., 2023a), derived from experts freely interacting with the environment. Fig. 13 presents screenshots of four such trajectories.

From these 18 trajectories, we extract a total of **613** Attitude Control demonstrations, which corresponds to approximately **2.5** hours of data. We train policies with IRPO on these human play demonstrations and compare their performance with policies trained on demonstrations generated by the PID controller. The results of this comparison are presented and analyzed in Section 5.6.

G.3.2. Comparison of Convergence Rate between IRPO and Baselines

Fig. 14 shows the training curves for IRPO (two iterations) and baselines on the Attitude Control task. On the one hand, the results indicate that IRPO demonstrates a faster convergence rate when achieving similar levels of policy performance compared to all baselines. On the other hand, the results suggest that IRPO requires fewer interaction steps to achieve the same level of policy performance compared to the baselines. Additionally, even when not considering the final policy performance, IRPO demonstrates faster convergence rate than both the annealed and EMA methods. Although IRPO does exhibit a certain advantage in convergence rate, its overarching advantage is its ability to converge to a policy that outperforms all the baselines.

For the above results, we give the following analysis. The annealed and EMA methods converge relatively slower during training because they modify the KL regularization strength λ and the KL reference policy π_k^0, respectively, which is equivalent to optimizing a dynamic objective. In contrast, the fixed method, which does not change the optimization objective, converges faster. Each iteration of IRPO is essentially akin to the fixed method. Therefore, IRPO has a certain

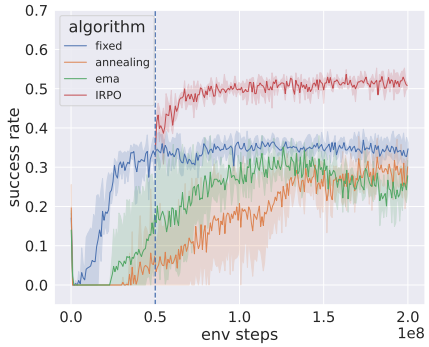


Figure 14. Training curves of IRPO and baselines on the Attitude Control. Results come from experiments over 5 random seeds.

advantage in convergence rate compared to the annealed and EMA methods. Although IRPO has a slight disadvantage in convergence rate compared to the Fixed method, it significantly improves the performance of the policy.