Theoretically Guaranteed Policy Improvement Distilled from Model-Based Planning

Chuming Li[1,2*], Ruonan Jia[1,3*], Jiawei Yao[4], Jie Liu[1], Yinmin Zhang[1,2], Yazhe Niu[1], Yaodong Yang[5], Yu Liu[1], Wanli Ouyang[1]

[1] Shanghai Artificial Intelligence Laboratory [2] University of Sydney
[3] Tsinghua University [4] University of Washington [5] Peking University
{lichuming.lcm, jiaruonan97, liuyuisanai}@gmail.com, {liujie, niuyazhe}@pjlab.org.cn,
{yinmin.zhang, wanli.ouyang}@sydney.edu.au, jwyao@uw.edu, yaodong.yang@pku.edu.cn

Abstract

Model-based reinforcement learning (RL) has demonstrated remarkable successes on a range of continuous control tasks due to its high sample efficiency. To save the computation cost of conducting planning online, recent practices tend to distill optimized action sequences into an RL policy during the training phase. Although the distillation can incorporate both the foresight of planning and the exploration ability of RL policies, the theoretical understanding of these methods is yet unclear. In this paper, we extend the policy improvement step of Soft Actor-Critic (SAC) by developing an approach to distill from model-based planning to the policy. We then demonstrate that such an approach of policy improvement has a theoretical guarantee of monotonic improvement and convergence to the maximum value defined in SAC. We discuss effective design choices and implement our theory as a practical algorithm—***M**odel-based **P**lanning **D**istilled to **P**olicy (MPDP)*—that updates the policy jointly over multiple future time steps. Extensive experiments show that MPDP achieves better sample efficiency and asymptotic performance than both model-free and model-based planning algorithms on six continuous control benchmark tasks in MuJoCo.

Introduction

Model-based Reinforcement Learning (RL) has achieved great success on continuous control tasks (Levine and Abbeel 2014; Heess et al. 2015; Buckman et al. 2018; Kurutach et al. 2018; Zhang et al. 2022). Model-based RL algorithms learn the true dynamics by fitting a model (usually a neural network) to the repeated interactions with the environment and use the model to generate imaginary data or perform online planning, which provides better sample efficiency than model-free RL (Mnih et al. 2013; Schulman et al. 2017; Fujimoto, Hoof, and Meger 2018; Hu et al. 2021).

A typical kind of model-based RL algorithm performs online planning to optimize the future action sequence over a long time horizon, i.e., model-based planning (Levine and Abbeel 2014; Chua et al. 2018; Wang and Ba 2019; Rybkin et al. 2021). However, model-based planning has two weaknesses. First, it can hardly be applied in real-time, because it needs to solve an optimization problem on each time step and cannot remember the solution for reuse in the future similar states (Wang and Ba 2019). Second, it only optimizes the maximum of the reward sum over the future states, rather than the trade-off between exploration and exploitation, which limits the ability to discover diverse states and better policies (Levine and Abbeel 2014). To reduce the time consumption during the application and incorporate the foresight of planning and the exploration ability of RL, some recent works distill the result of model-based planning into an RL policy (Levine and Abbeel 2014; Wang and Ba 2019). Specifically, POPLIN uses the cross entropy method (CEM) (Botev et al. 2013) to optimize the action planning and uses behavior cloning to distill the planning result into the policy network. However, some essential theoretical properties of such kind of distillation are not well-understood, i.e., **(1)** whether the distilled policy achieves a higher value than the old policy; **(2)** whether the distilled policy has a guarantee of convergence to the optimal policy; **(3)** whether the distilled policy incorporates the foresight of planning and achieves a higher value than the original model-free policy update.

In this paper, we theoretically analyze the problems mentioned above. We choose Soft Actor-Critic (SAC) (Haarnoja et al. 2018) as the RL component of our analysis due to its state-of-the-art performance in both model-free and model-based paradigms. Originally, the policy improvement of SAC is a one-step optimization. We first define a planning problem by extending the one-step optimization of SAC under the model-based paradigm to a multi-step optimization problem of action planning. For each state $\boldsymbol{s}_t$, the optimal planning solution returns a policy $\pi^H_{\boldsymbol{s}_t}$ defined on a horizon of states $\boldsymbol{s}_{t:t+H-1}$ starting from $\boldsymbol{s}_t$. Then, we propose a simple approach to distill the solution of the above multi-step optimization to the policy, which is an extended form of the policy improvement of SAC. This approach reserves the returned policy $\pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_t)$ for the first state $\boldsymbol{s}_t$ and discards the returned policy $\pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_{t+1:H-1})$ for the future states.

Afterwards, we derive the theoretical result that the extended policy improvement is promising to achieve a higher return and lead the policy to converge to the optimal policy. Thus the extension incorporates the farsight planning and has the potential to improve remarkably upon origi-

*These authors contributed equally.

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Algorithms	Ensemble Dynamics	Multiple Horizon	Regularization	Planning Theorem
SAC(Haarnoja et al. 2018)	✗	✗	✗	✗
MBPO(Janner et al. 2019)	✔	✗	✗	✗
POPLIN(Wang and Ba 2019)	✔	✔	✗	✗
M2AC(Pan et al. 2020)	✔	✔	✗	✗
MPDP(our work)	✔	✔	✔	✔

Table 1: Key features of different model-free and model-based algorithms.

nal one-step policy improvement. Furthermore, to develop a practical algorithm, we discuss the solver of the defined multi-step optimization and design regularization to reduce the model error. Based on the above theory and discussion, we propose a new model-based RL algorithm, ***M**odel-based **P**lanning **D**istilled to **P**olicy (MPDP)*. Compared to POPLIN, which uses behavior cloning for distillation and realizes the stochastic exploration via the CEM sampling, MPDP utilizes a distillation approach with theoretically guaranteed improvement and inherits the stochastic exploration of SAC, thus has a naturally strong ability to explore better policies. For illustrating the effectiveness of MPDP, a thorough component comparison of relevant algorithms is given in Table 1.

Summary of Contributions: (1) We propose a model-based extended policy improvement method, which utilizes model-based planning to distill RL policy and model regularization to reduce the impact of model errors. **(2)** We demonstrate that our method has a theoretical guarantee of monotonic improvement and convergence. And we theoretically analyze how the planning horizon affects policy improvement. **(3)** Experimental results empirically show that MPDP achieves better sample efficiency and asymptotic performance than state-of-the-art model-free and model-based planning algorithms on the MuJoCo (Todorov, Erez, and Tassa 2012) benchmark.

Related Work

Model-based Reinforcement Learning. Model-based reinforcement learning methods show a promising prospect for real-world decision-making problems due to their data efficiency. However, learning an accurate model is challenging, especially in complex environments. Many papers (Chua et al. 2018; Kurutach et al. 2018; Janner et al. 2019; Wang et al. 2023) commonly use ensemble probabilistic networks to construct uncertainty-aware environment models.

The previously proposed model-based methods (Feinberg et al. 2018; Buckman et al. 2018; Jia et al. 2021; Voelcker et al. 2022) allow the model rollout to a fixed depth, and value estimations are split into a model-based reward and a model-free value. To guarantee the monotonic improvement, the recent work (Luo et al. 2019) builds a lower bound of the expected reward and then maximizes the lower bound jointly over the policy and the model. Furthermore, model-based policy optimization (Janner et al. 2019) utilizes short model-generated rollouts to do policy improvement and evaluation, and also provides a guarantee of monotonic improvement.

Current model-based RL mainly focuses on better model usage. For example, M2AC (Pan et al. 2020) implements a masking mechanism based on the model's uncertainty to decide whether its prediction should be used or not. Another line of works (Levine and Abbeel 2014; Heess et al. 2015) aims to exploit the differentiability of the learned model in model-based RL. Model-augmented actor-critic (Clavera, Fu, and Abbeel 2020) uses the path-wise derivative of the learned model and policy across future time steps. Moreover, our work estimates value function by utilizing the model error as regularization.

Model-based Planning. Many recent papers on deep model-based RL (Chua et al. 2018; Ebert et al. 2018; Tassa, Erez, and Todorov 2012) optimize the future action trajectories over a given horizon starting from the current state, which is usually referred as model-based planning. Model predictive control (Tassa, Erez, and Todorov 2012) is a common control approach for model-based planning. It frequently solves the action planning over a limited horizon and conducts the first action on the environment. Random Shooting optimizes the action sequence among the randomly generated candidates to maximize the expected reward under the learned dynamic model, and PETS (Chua et al. 2018) uses the cross entropy method (Botev et al. 2013) to improve the efficiency of the random search. However, shooting methods usually rely on the local search in the action space and are not effective on high-dimension environments. To solve this problem, the latest work (Rybkin et al. 2021) utilizes the collocation-based planning in a learned latent space. In contrast, we extend the policy improvement step of SAC to distill from model-based planning to the policy, which reduces the cost in the deployment phase.

In addition, some recent works distill the result from model-based policy planning into an RL policy. POPLIN (Wang and Ba 2019) formulates action planning at each time step as an optimization problem w.r.t. the parameters of the policy network, and uses behavior cloning to distill the resulted action into the policy network. GPS (Levine and Koltun 2013; Levine and Abbeel 2014) uses KL divergence to minimize the distance between the policy and the planning result. However, the essential theoretical properties of such distillation are not well-understood. Instead, we propose an algorithm to improve the policy with the solution of model-based planning over multiple time steps, and give the theoretical guarantee of its improvement and convergence.

Actor-Critic Methods. Actor-critic algorithms are typically derived from policy iteration, which alternates between policy evaluation and policy improvement. Deep deterministic policy gradient (Lillicrap et al. 2016) is a common model-free actor-critic method, however, the critic is usually overestimated to predict Q value, which leads to the worse policy. Moreover, twin delayed deep deterministic policy (Fujimoto, Hoof, and Meger 2018) mainly utilizes the clipped double Q learning to alleviate the above overestimation. SAC (Haarnoja et al. 2018; Zhou et al. 2022) is the SOTA algorithm of policy learning under the model-based paradigm. In the framework of SAC, the actor aims to maximize expected reward with entropy and the critic evaluates the expected cumulative reward with entropy. Due to the splendid performance of SAC, we choose it as the RL instance to prove the theoretical properties, by distilling the planning into an RL policy.

Preliminaries

Notation

We consider continuous control tasks which can be formulated as infinite-horizon Markov Decision Processes (MDP) $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where the state space $\mathcal{S}$ and the action space $\mathcal{A}$ are both continuous. State transition $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^+$ and $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ are the dynamics of the environment and the reward function, respectively. γ is the discount factor. Additionally, we define $\pi(\boldsymbol{a}|\boldsymbol{s}) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^+$ as the RL policy on the state $\boldsymbol{s}$, with $Q(\boldsymbol{s}, \boldsymbol{a})$ and $V(\boldsymbol{s})$ as the corresponding value functions.

Soft Actor-Critic

Soft Actor-Critic(SAC) (Haarnoja et al. 2018) develops a maximum entropy objective to incentivize the policy to explore more widely, which is the discounted sum of both the reward and the entropy, formalized as:

$$J_{\boldsymbol{s_t}}(\pi) = \mathbb{E}_{\boldsymbol{a_t} \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t \cdot [r(\boldsymbol{s}_t, \boldsymbol{a}_t) - \alpha \cdot log\pi(\boldsymbol{a}_t|\boldsymbol{s}_t)] \right]. \tag{1}$$

The coefficient α balances the importance of the reward and entropy, and hence controls the exploration of the policy. we omit α in the rest of this paper for simplicity. The policy evaluation of SAC is based on the maximum entropy objective, i.e., the value function Q and V also contain the discounted sum of the entropy over the subsequent states. The Bellman backup operator $\mathcal{T}^\pi$ of SAC is given by:

$$\mathcal{T}^\pi Q(\boldsymbol{s}_t, \boldsymbol{a}_t) = r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \cdot V(\boldsymbol{s}_{t+1}), \tag{2}$$

$$V(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a_t} \sim \pi} \left[Q(\boldsymbol{s}_t, \boldsymbol{a}_t) - log\pi(\boldsymbol{a}_t|\boldsymbol{s}_t) \right]. \tag{3}$$

In the policy improvement step of SAC, the new policy optimizes the $V(\boldsymbol{s}_t)$ on each state $\boldsymbol{s}_t$:

$$\pi_{new}(\cdot|\boldsymbol{s}_t) = \arg\max_{\pi} \mathbb{E}_{\boldsymbol{a_t} \sim \pi} \left[Q^{\pi_{old}}(\boldsymbol{s}_t, \boldsymbol{a}_t) - log\pi(\boldsymbol{a}_t|\boldsymbol{s}_t) \right]. \tag{4}$$

We reformulate the objective as:

$$\pi_{new}(\cdot|\boldsymbol{s}_t) = \arg\max_{\pi} \mathbb{E}_{\boldsymbol{a_t} \sim \pi} \left[\, r(\boldsymbol{s}_t, \boldsymbol{a}_t) - log\pi(\boldsymbol{a}_t|\boldsymbol{s}_t) + \gamma \cdot V^{\pi_{old}}(\boldsymbol{s}_{t+1}) \, \right]. \tag{5}$$

This objective leads the new policy to optimize the modified reward $r(\boldsymbol{s}_t, \boldsymbol{a}_t) - log\pi(\boldsymbol{a}_t|\boldsymbol{s}_t)$ only on the current state $\boldsymbol{s}_t$ w.r.t. $\boldsymbol{a}_t$, with the subsequent states following the old policy π_{old}, which is myopic under the model-based paradigm, because the dynamics of the environment can be approximated by the environment model, which enables the joint optimization of actions over multiple future time-steps.

Environment Model

A common setting used in model-based RL is model ensemble (Chua et al. 2018; Kurutach et al. 2018; Janner et al. 2019; Luo et al. 2019; Pan et al. 2020), where an ensemble of models learn the distribution of the transitions from historical interactions. Typically, the models are parametric function approximators $p_{1:K}(\cdot|\boldsymbol{s}, \boldsymbol{a})$ and are trained via maximum likelihood: $\sum_{i=1}^{K} \mathbb{E}\left[log(p_i(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t))\right]$.

Distillation from Planning into Policy

In this section, we propose an approach to distilling the solution of model-based planning into the policy, which is a multi-step extension of the original policy improvement of SAC. We will first derive this extension. Then, we will verify its theoretical properties and advantages. Finally, based on our theory, we will develop a practical reinforcement learning algorithm by discussing the essential design choices in the next section.

Multi-step Optimization

The policy improvement of SAC optimizes the trade-off between the expected cumulative reward and entropy only with regard to the action distribution on the current time-step $\boldsymbol{s}_t$, with the future states $\boldsymbol{s}_{t+1:\infty}$ following the old policy π_{old}, formalized in Equation (5). Under the model-based paradigm, we assume that the true dynamics of the environment is accessible. Because we can always obtain a dynamic model with a lower generalization error (Kurutach et al. 2018; Janner et al. 2019), as the training proceeds. This assumption enables us to quantify the expected future state and the according reward and entropy with regard to the future action sequence over a given horizon H, and derive a more foresighted optimization form than the original SAC. Specifically, we extend the one-step optimization in Equation (5) to a multi-step optimization problem of the action planning over H steps based on the environment model, with the objective $J^H_{\boldsymbol{s}_t}(\pi)$ on the state $\boldsymbol{s}_t$ defined as:

$$J^H_{\boldsymbol{s}_t}(\pi) = \mathbb{E}_{\boldsymbol{a_t} \sim \pi} \left[\sum_{i=0}^{H-1} \gamma^i \cdot r^\pi(\boldsymbol{s}_{t+i}, \boldsymbol{a}_{t+i}) + V^{\pi_{old}}(\boldsymbol{s}_{t+H}) \right], \tag{6}$$

$$r^\pi(\boldsymbol{s}_{t+i}, \boldsymbol{a}_{t+i}) = r(\boldsymbol{s}_{t+i}, \boldsymbol{a}_{t+i}) - log\pi(\boldsymbol{a}_{t+i}|\boldsymbol{s}_{t+i}). \tag{7}$$

Here H is the planning horizon, π is the policy only defined on $\boldsymbol{s}_t$ and its subsequent $H-1$ steps. $r^{\pi}(\boldsymbol{s},\boldsymbol{a})$ is the sum of the reward and the logarithmic likelihood, which inherits the maximum entropy objective of SAC. Specifically, when $H=1$, this objective degenerates to that of SAC.

Extended Policy Improvement

The improvement property of distillation from planning into an RL policy has not been well discussed. Another work(Clavera, Fu, and Abbeel 2020) proves that the solution of action planning achieves a higher value, but it does not develop a distillation approach to obtain a policy π_{new} with provably higher value $V^{\pi_{new}}(\boldsymbol{s}_t)$, i.e., a policy with higher cumulative rewards. In this section, we propose a distillation approach, also an extended form of the original policy improvement step in SAC, based on the multi-step optimization. We will show that the proposed extended policy improvement provably achieves a new policy with a higher value than the old policy with respect to the maximum entropy target Equation (1) defined in SAC.

Distillation. We use $\pi^H_{\boldsymbol{s}_t}$ to denote the optimal solution of $J^H_{\boldsymbol{s}_t}(\pi)$. After the policy improvement, we define the new policy $\pi_{new}(\cdot|\boldsymbol{s}_t)$ as $\pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_t)$, i.e., although $\pi^H_{\boldsymbol{s}_t}$ is define on H steps of states $\boldsymbol{s}_{t:t+H-1}$, we only adopt the policy $\pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_t)$ of the current state $\boldsymbol{s}_t$ and discard the policy $\pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_{t+1:t+H-1})$ over the following states.

Improvement. We present the improvement property of this distillation in Lemma 1. Please note that Lemma 1 is a more general multi-step extension of the Lemma 2[1] in SAC (Haarnoja et al. 2018). Our result reveals that, if we optimize the policy jointly over a horizon starting from each state $\boldsymbol{s}_t$ and only adopt the optimal policy on the first state $\boldsymbol{s}_t$, the resulting new policy has a monotonic improvement. Specifically, when $H=1$, Lemma 1 degenerates to the Lemma 2 in SAC (see Appendix A. for more details).

Lemma 1. *Let $\pi^H_{\boldsymbol{s}_t}$ be the optimizer of the optimization objective of Equation* (6)*. When the new policy $\pi_{new}(\cdot|\boldsymbol{s}_t) = \pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_t)$, $V^{\pi_{new}}(\boldsymbol{s}_t) \geq V^{\pi_{old}}(\boldsymbol{s}_t)$ for all $\boldsymbol{s}_t \in S$.*

Policy Convergence

The monotonic increasing property of our extended form is crucial, because it facilitates the derivation of the proposition that this form will provably converge to the optimal maximum entropy policy defined in SAC. We present the result in Theorem 2.

Theorem 2. *Let π_0 be any initial policy. Assuming $|\mathcal{A}| < \infty$, if the policy evaluation in Equation* (2) *and the policy improvement with the objective in Equation* (6) *are alternatively carried out, π_0 converges to a policy π_*, with $V^{\pi_*}(\boldsymbol{s}_t) \geq V^{\pi}(\boldsymbol{s}_t)$ for any $\boldsymbol{s}_t \in S$.*

The Effect of Planning Horizon

We have shown that the proposed extension of policy improvement, based on optimization of the action planning over multiple time steps, can always lead to a higher value via the developed distillation, which is guaranteed to converge to the optimal policy. In this section, we will discuss another problem: does the extended form of policy improvement incorporate the farsight of planning and benefit SAC? Or more generally, does a larger planning horizon H always result in a better value?

Unfortunately, there exist some special cases where a larger H leads to a smaller value due to a bad initial policy π_{old}. We construct a counter-example in Appendix C. Nevertheless, although a larger H is not equivalent to a higher value, we can still show the potential advantage of increasing H in two aspects.

(1) A larger horizon results in a higher optimization objective defined in Equation (6), as formalized in Lemma 3.

Lemma 3. *Let $\pi^H_{\boldsymbol{s}_t}$ and $\pi^{H+1}_{\boldsymbol{s}_t}$ be the optimal solution of $J^H_{\boldsymbol{s}_t}(\pi)$ and $J^{H+1}_{\boldsymbol{s}_t}(\pi)$. Then $J^{H+1}_{\boldsymbol{s}_t}(\pi^{H+1}_{\boldsymbol{s}_t}) \geq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$ for all $H \geq 1$ and $\boldsymbol{s}_t \in S$.*

(2) Although the resulting policy does not have a value monotonically increasing with H, we can prove that π_{new} converges to the optimal policy as H increases, which is formalized in Theorem 4.

Theorem 4. *Let $\pi^H_{\boldsymbol{s}_t}$ be the optimal solution of $J^H_{\boldsymbol{s}_t}(\pi)$, and $\pi_{new}(\cdot|\boldsymbol{s}_t) = \pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_t)$. π_* denotes the optimal policy. As H increases, $V^{\pi_{new}}$ and $J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$ converge to V^{π_*} for all $\boldsymbol{s}_t \in S$. Specifically, $V^{\pi_{new}} \geq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) \geq V^{\pi_*}(\boldsymbol{s}_t) - \frac{\gamma^H \cdot r^{max}}{1-\gamma}$ with r^{max} the maximum of $r^{\pi}(\boldsymbol{s},\boldsymbol{a})$ over all π and $(\boldsymbol{s},\boldsymbol{a}) \in |\mathcal{S}| \times |\mathcal{A}|$.*

Starting from Theorem 4, it can be naturally derived that, we can always find a larger $\hat{H}$ than H, which results in a policy with a larger value. We formalize this conclusion as Theorem 5.

Theorem 5. *Let $\pi^H_{\boldsymbol{s}_t}$ be the optimal solution of $J^H_{\boldsymbol{s}_t}(\pi)$, and $\pi^H_{new}(\cdot|\boldsymbol{s}_t) = \pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_t)$. There exists another $\hat{H} > H$, with $V^{\pi^{\hat{H}}_{new}} \geq V^{\pi^H_{new}}$ for all $\boldsymbol{s}_t \in S$, assuming $|\mathcal{S}| < \infty$.*

Proof. According to Theorem 4, we can always find a $\hat{H}$ with $V^{\pi_*} - V^{\pi^{\hat{H}}_{new}} \leq V^{\pi_*} - V^{\pi^H_{new}}$ on all states, which means $V^{\pi^{\hat{H}}_{new}} \geq V^{\pi^H_{new}}$. □

Implementation

According to the above theory, the proposed extended policy improvement via planning over multiple time steps can also guarantee value improvement and convergence to the optimal policy. And the increase of planning horizon has the potential to get a better new policy. In this section, we discuss some essential design choices for distilling the model-based planning into SAC (Haarnoja et al. 2018). We further propose a practical algorithm, ***M**odel-based **P**lanning **D**istilled to **P**olicy (MPDP)*, under the model-based paradigm. There are two essential issues in the design of MPDP, **(1)** how to solve the objective in Equation (6), and **(2)** how to reduce the bias introduced by the generalization error of the environment model.

[1] https://arxiv.org/pdf/1801.01290.pdf

Solver

Solving the proposed objective defined by Equation (6) is a model-based planning problem, which has been widely discussed in many prior works (Rybkin et al. 2021; Chua et al. 2018; Wang and Ba 2019). We roughly divide the current solvers into two categories, sample-based methods and gradient-based methods.

Sample-based methods typically include random shooting and cross-entropy method (CEM) (Botev et al. 2013). However, sample-based methods are usually inefficient in complex high-dimensional tasks. Gradient-based methods include gradient optimization and collocation method (Rybkin et al. 2021), which optimize with reward to the action sequence and backpropagate the gradient to all actions in the sequence. Both gradient optimization and collocation methods suit our formulation due to their accessibility of the gradient. We can develop a practical algorithm based on both of them. We observe that they perform comparably on the MuJoCo benchmark in our early-stage experiments.

With the above discussion, we choose gradient optimization as our solver, because it naturally suits the framework of SAC and achieves comparable performance without introducing extra hyperparameters and computational cost compared to the collocation method.

Algorithm 1: Farsighted Policy Improvement

Require: state batch B, policy networks $\pi_{0:H_{max}-1}$, dynamic models $p_{1:K}$, threshold u_T, coefficient α and β

for $\boldsymbol{s}$ **in** B **do**
  $\boldsymbol{s}_0 = \boldsymbol{s}$, $J = 0$
  for $t = 0 : H_{max} - 1$ **do**
    Sample $\boldsymbol{a}_t \sim \pi_t$
    Predict $\boldsymbol{s}_{t+1} \sim p_{1:K}(\boldsymbol{s}_{t+1}, \boldsymbol{a}_t)$
    if $u(\boldsymbol{s}_t, \boldsymbol{a}_t) \geq u_T$ or $\boldsymbol{s}_{t+1}$ is a terminal state **then**
      $J = J + \gamma^{t+1} \cdot V(\boldsymbol{s}_{t+1})$
      break
    end if
    $J = J + \gamma^t \cdot [r(\boldsymbol{s}_t, \boldsymbol{a}_t) - \alpha \cdot log\pi(\boldsymbol{a}_t|\boldsymbol{s}_t) - \beta \cdot u(\boldsymbol{s}_t, \boldsymbol{a}_t)]$
  end for
end for
Update $\pi_{0:H_{max}-1}$ with the mean of $\nabla_{\boldsymbol{a}_{0:H_{max}-1}} J$

Model Regularization

The bias resulting from the environment model's generalization error raises two issues for consideration. First, although increasing the planning horizon has the potential of resulting in a higher value theoretically, we must consider the trade-off between the bias of $Q^{\pi_{old}}$ and the environment model. A larger H introduces more model bias but reduces the bias of $Q^{\pi_{old}}$. Second, we need to avoid the update of the policy towards the area where the model has high generalization error, because this will result in a sub-optimal solution and the gradients of the environment model at those unseen state-action pairs (s, a) are unsupervised and not numerically stable, i.e., applying the environment model iteratively for many time steps may lead to gradient explosion (Rybkin et al. 2021).

Both the two issues need the estimation of the model error, which has been well discussed in prior works. In this paper, we use One-vs-Rest (OvR) (Pan et al. 2020), a simple method to estimate model errors. OvR learns multiple dynamic models and uses the KL divergence between models as an estimator of model error, which is formalized as:

$$u(\boldsymbol{s}, \boldsymbol{a}) = \sum_{i=1}^{K} D_{KL}[p_i(\cdot|\boldsymbol{s}, \boldsymbol{a}) \| p_{-i}(\cdot|\boldsymbol{s}, \boldsymbol{a})]. \tag{8}$$

Here $p_i(\cdot|s, a)$ is the predicted distribution of the one model and $p_{-i}(\cdot|s, a)$ is the mean of the rest models' prediction.

Based on OvR, we develop two methods separately for the above two issues. First, we use adaptive horizons for trajectories starting from different states. The planning stops when a trajectory generates a state-action pair which has a model error larger than a pre-defined threshold. Secondly, we develop an additional regularization of model error, which adds the model error estimated by OvR on our objective Equation (6). This regularization directs the final solution to the area where the environment model is more believable and reduces both the numerical instability and the model error. Specifically, we add the estimation $u(\boldsymbol{s}, \boldsymbol{a})$ on the original reward $r^{\pi}(s, a)$ as a regularization, and re-formalize Equation (6) as:

$$J^{H,u}_{\boldsymbol{s}_t}(\pi) = \mathbb{E}_{\boldsymbol{a}\sim\pi}\left[\sum_{i=0}^{H-1} \gamma^i \cdot r^{\pi,u}(\boldsymbol{s}_{t+i}, \boldsymbol{a}_{t+i}) + V^{\pi_{old}}(\boldsymbol{s}_{t+H})\right], \tag{9}$$

$$\begin{aligned} r^{\pi,u}(\boldsymbol{s}_{t+i}, \boldsymbol{a}_{t+i}) = & r(\boldsymbol{s}_{t+i}, \boldsymbol{a}_{t+i}) - log\pi(\boldsymbol{a}_{t+i}|\boldsymbol{s}_{t+i}) \\ & - \beta \cdot u(\boldsymbol{s}_{t+i}, \boldsymbol{a}_{t+i}). \end{aligned} \tag{10}$$

Algorithm 2: Model-based Planning Distilled to Policy

Initialize data buffer $D = \emptyset$, dynamic models $p_{1:K}$, policy networks $\pi_{0:H_{max}-1}$, value networks Q and V
repeat
  Collect data from real environment with policy π_0: $D \leftarrow D \cup (s, a, r, s')$
  Train ensemble models $p_{1:K}$ on D
  Sample a batch B from D
  Update Q and V with B as in SAC
  Update $\pi_{0:H_{max}-1}$ by Algorithm 1.
until Convergence

Model-based Planning Distilled to Policy

We conclude our extended policy improvement in Algorithm 1. The algorithm processes a batch of states at each iteration and the model rollouts states until the task terminates, that is to say, the pair of $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ has a larger model error than the threshold u_T, or the rollout reaches the max horizon H_{max}. And we maintain the policy networks $\pi_{0:H_{max}-1}$ at H time steps. The policy networks generate the actions for each step and are updated jointly in our extended improvement step. After the model rollouts, the policy networks $\pi_{0:H_{max}-1}$ are updated with the gradients to

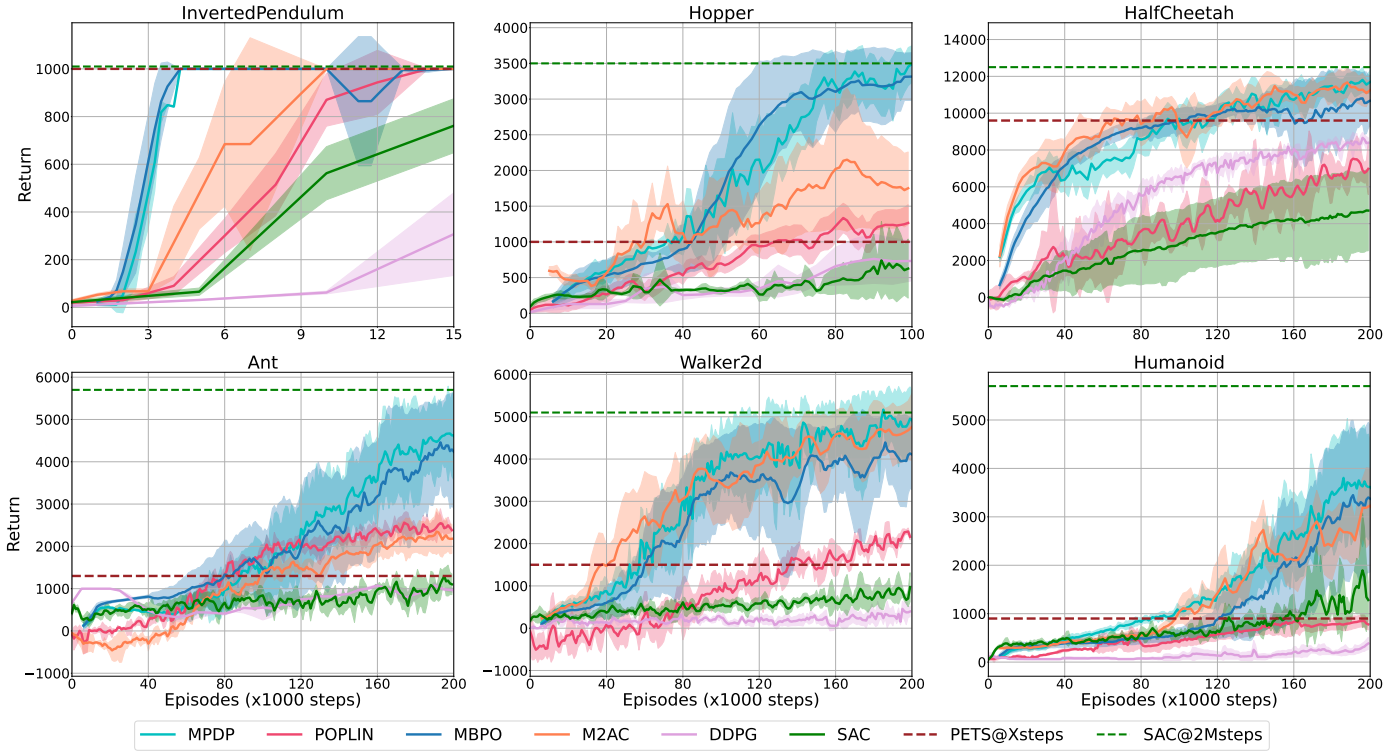


Figure 1: Performance curves for our method (MPDP) and baselines on MuJoCo continuous control benchmarks. Solid lines depict the mean of four trials and shaded regions correspond to standard deviation among trials. The dashed lines indicate the asymptotic performance of PETS at the corresponding training steps (15k steps for InvertedPendulum, 100k steps for Hopper, and 200k steps for the other tasks) and SAC at 2M steps.

the action sequence. The complete algorithm is described in Algorithm 2. The method alternates among using the policy π_0 on the first step to interact with the environment, training an ensemble of models, and updating the policy with policy evaluation and our extended policy improvement.

Experiment

Our experiment goal is to investigate the following questions: **(1)** How the sample efficiency and the asymptotic performance of MPDP compared to state-of-the-art(SOTA) model-based planning algorithms? **(2)** How the proposed extended policy improvement and the design choices affect the performance?

Comparison

Baseline. In this section, we focus on understanding how well MPDP performs compared to SOTA model-based planning algorithms. We choose PETS (Chua et al. 2018), which uses CEM to perform model-based action planning; and POPLIN (Wang and Ba 2019), which extends CEM from action space to the domain of policy network parameters and distills the planning results into the policy with behavior cloning. Additionally, we compare our proposed approach to the SOTA model-free methods and model-based methods without planning. For model-free algorithms, we compare to SAC (Haarnoja et al. 2018) and DDPG (Lillicrap et al. 2016), which are the two competitive policy learning algorithms. For model-based RL, we choose MBPO (Janner et al. 2019) and M2AC (Pan et al. 2020), which are the previous SOTA model-based baselines. MPDP, PETS, POPLIN, MBPO and M2AC share the same model architecture. The implementation details of our method are in Appendix B.

Results. The performance curves on all six environments of MuJoCo are shown in Figure 1. It demonstrates that MPDP significantly outperforms the SOTA model-based planning algorithms (PETS and POPLIN) on both sample efficiency and asymptotic performance. For example, on the highly dimensional Ant task, MPDP's performance at 140k steps is equivalent to that of POPLIN at 200k steps.

Further, the results in Figure 1 reveal that MPDP achieves much higher convergence speed than the SOTA of model-free algorithms (SAC and DDPG) on the all tasks and obtains comparable asymptotic performance, which also validates that incorporating our extended policy improvement benefits a lot. We also observe that MPDP achieves better performance than the SOTA model-based algorithms, MBPO and M2AC on some complex tasks like Humanoid, and is comparable to them on the rest of tasks.

Ablation Study

In this section, we conduct a series of ablation studies on MPDP to investigate the effect of the designed adaptive horizon and regularization on the model error. We choose the Hopper task in the MuJoCo for the experiments.

Horizon. To verify that our method can really adapt the horizon to the model error, i.e. the adaptive horizon does not fall into a very small range and increases as the model generalizes better, we profile the average horizon of MPDP during the training on Hopper with different error threshold u_T in Figure 2. As shown in the curves, the horizon grows from 2 to 12 as the training proceeds, where the model becomes more accurate in Figure 3. It also proves that MPDP does not degenerate to SAC.

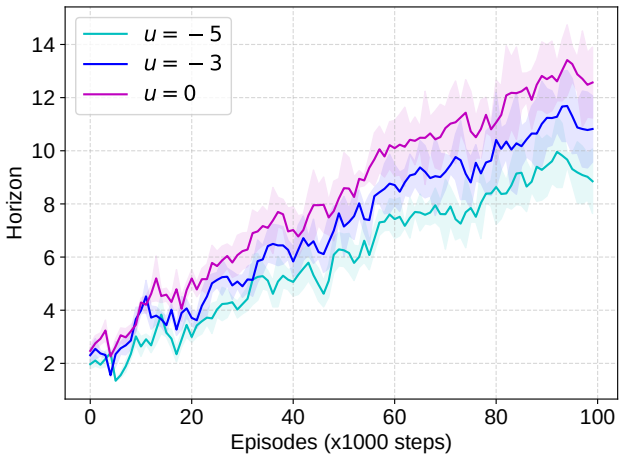


Figure 2: This figure demonstrates the length of the adaptive horizon of MPDP. The solid lines denote the average horizon length evaluated on each training batch. As the interactions accumulate, the model generalizes better and our method rapidly adapts to longer horizons.

Model Error. We validate that the regularization based on OvR does push the policy to explore areas with low dynamic model error. We vary β at Equation (10) with $\{0.2, 0.5, 0.7\}$ and evaluate the model error as shown in Figure 3. The result demonstrates that the model error decreases with β, which verifies the effectiveness of the designed regularization. We also plot the final performance of corresponding β in Figure 4. However, we find that a too large regularization harms the asymptotic performance due to the excessive restriction on the exploration area of the policy. Figure 4 also implies that a larger regularization brings more stable results.

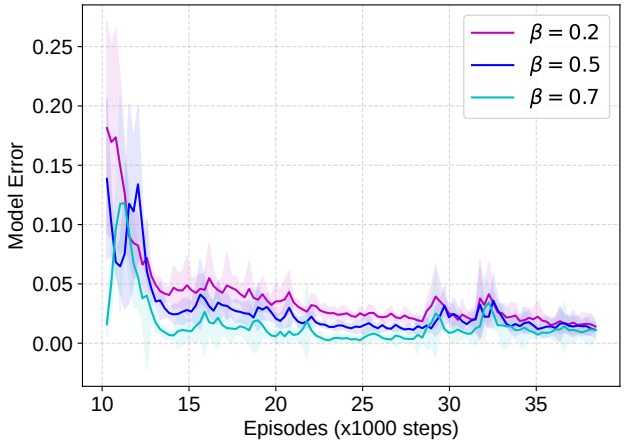


Figure 3: This figure shows the model error curves of MPDP with β varying from 0.2 to 0.7, measured by the average L_2 norm of the predicted states on every 250 interactions. The model error decreases with β, which verifies that optimizing under our regularization effectively restricts behavior policy in the areas with low model error.

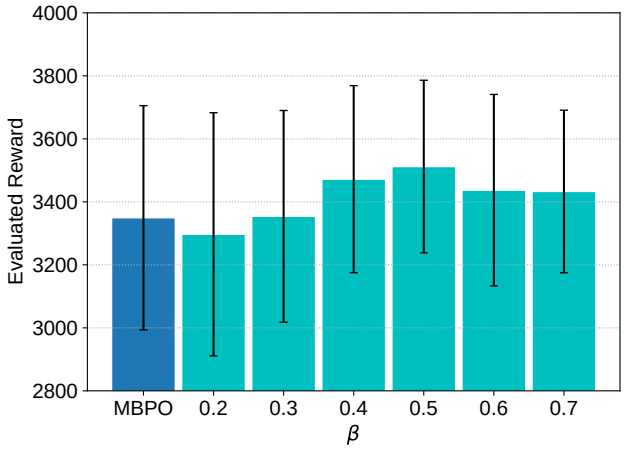


Figure 4: This figure displays the performance of MPDP with β varying from 0.2 to 0.7 along with MBPO on the Hopper task, evaluated over 4 trials. As β increases, the performance increases at first then decreases due to the too strong restriction on the exploration.

Conclusion and Future Work

In this paper, we investigate the theoretical guarantee of distillation from model-based planning into an RL policy. We first extend the one-step optimization of SAC to a multi-step optimization formulation. Then, we develop a distillation approach based on the solution of the proposed multi-step optimization. It provably has the guarantee of monotonic improvement and convergence to the optimal policy. We further theoretically verify its potential to incorporate the foresight planning. Based on the theory, we discuss several design choices to instantiate a practical algorithm MPDP. Experimental results confirm that MPDP outperforms the state-of-the-art model-based planning algorithms in both sample efficiency and asymptotic performance on a range of continuous control tasks in MuJoCo.

One limitation is that the generalization ability of the horizon-adapted policy may not be strong enough because we fit the horizon to the model error for fast convergence speed. Thus, our method is efficient for task-specific but not exploration-oriented problems. We leave this to future work.

References

Botev, Z. I.; Kroese, D. P.; Rubinstein, R. Y.; and L'Ecuyer, P. 2013. The cross-entropy method for optimization. In *Handbook of statistics*, volume 31, 35–59. Elsevier.

Buckman, J.; Hafner, D.; Tucker, G.; Brevdo, E.; and Lee, H. 2018. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 8234–8244.

Chua, K.; Calandra, R.; McAllister, R.; and Levine, S. 2018. Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models. *Advances in Neural Information Processing Systems*, 31.

Clavera, I.; Fu, Y.; and Abbeel, P. 2020. Model-Augmented Actor-Critic: Backpropagating through Paths. In *8th International Conference on Learning Representations*.

Ebert, F.; Finn, C.; Dasari, S.; Xie, A.; Lee, A. X.; and Levine, S. 2018. Visual Foresight: Model-Based Deep Reinforcement Learning for Vision-Based Robotic Control. *CoRR*, abs/1812.00568.

Feinberg, V.; Wan, A.; Stoica, I.; Jordan, M. I.; Gonzalez, J. E.; and Levine, S. 2018. Model-Based Value Estimation for Efficient Model-Free Reinforcement Learning. *CoRR*, abs/1803.00101.

Fujimoto, S.; Hoof, H.; and Meger, D. 2018. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, 1587–1596. PMLR.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 1861–1870. PMLR.

Heess, N.; Wayne, G.; Silver, D.; Lillicrap, T.; Erez, T.; and Tassa, Y. 2015. Learning Continuous Control Policies by Stochastic Value Gradients. *Advances in Neural Information Processing Systems*, 28: 2944–2952.

Hu, H.; Ye, J.; Zhu, G.; Ren, Z.; and Zhang, C. 2021. Generalizable Episodic Memory for Deep Reinforcement Learning. In *International Conference on Machine Learning*, 4380–4390. PMLR.

Janner, M.; Fu, J.; Zhang, M.; and Levine, S. 2019. When to Trust Your Model: Model-Based Policy Optimization. *Advances in Neural Information Processing Systems*, 32: 12519–12530.

Jia, R.; Li, Q.; Huang, W.; Zhang, J.; and Li, X. 2021. Consistency Regularization for Ensemble Model Based Reinforcement Learning. In *Trends in Artificial Intelligence: 18th Pacific Rim International Conference on Artificial Intelligence, Proceedings, Part III 18*, 3–16. Springer.

Kurutach, T.; Clavera, I.; Duan, Y.; Tamar, A.; and Abbeel, P. 2018. Model-Ensemble Trust-Region Policy Optimization. In *International Conference on Learning Representations*.

Levine, S.; and Abbeel, P. 2014. Learning Neural Network Policies with Guided Policy Search under Unknown Dynamics. In *NIPS*, volume 27, 1071–1079. Citeseer.

Levine, S.; and Koltun, V. 2013. Guided policy search. In *International conference on machine learning*, 1–9. PMLR.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. In *ICLR (Poster)*.

Luo, Y.; Xu, H.; Li, Y.; Tian, Y.; Darrell, T.; and Ma, T. 2019. Algorithmic Framework for Model-based Deep Reinforcement Learning with Theoretical Guarantees. In *International Conference on Learning Representations*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Pan, F.; He, J.; Tu, D.; and He, Q. 2020. Trust the model when it is confident: Masked model-based actor-critic. *Advances in neural information processing systems*, 33: 10537–10546.

Rybkin, O.; Zhu, C.; Nagabandi, A.; Daniilidis, K.; Mordatch, I.; and Levine, S. 2021. Model-based reinforcement learning via latent-space collocation. In *International Conference on Machine Learning*, 9190–9201. PMLR.

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

Tassa, Y.; Erez, T.; and Todorov, E. 2012. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4906–4913. IEEE.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033. IEEE.

Voelcker, C. A.; Liao, V.; Garg, A.; and Farahmand, A.-m. 2022. Value Gradient weighted Model-Based Reinforcement Learning. In *International Conference on Learning Representations*.

Wang, T.; and Ba, J. 2019. Exploring Model-based Planning with Policy Networks. In *International Conference on Learning Representations*.

Wang, X.; Wongkamjan, W.; Jia, R.; and Huang, F. 2023. Live in the Moment: Learning Dynamics Model Adapted to Evolving Policy. *arXiv preprint arXiv:2207.12141*.

Zhang, M.; Zhang, S.; Yang, Z.; Chen, L.; Zheng, J.; Yang, C.; Li, C.; Zhou, H.; Niu, Y.; and Liu, Y. 2022. GoBigger: A Scalable Platform for Cooperative-Competitive Multi-Agent Interactive Simulation. In *The Eleventh International Conference on Learning Representations*.

Zhou, T.; Wang, L.; Chen, R.; Wang, W.; and Liu, Y. 2022. Accelerating Reinforcement Learning for Autonomous Driving using Task-Agnostic and Ego-Centric Motion Skills. *arXiv preprint arXiv:2209.12072*.

Appendix: Theoretically Guaranteed Policy Improvement Distilled from Model-Based Planning

A. Proof of Lemma and Theorem

In this section, we provide proofs for bounds presented in the main paper.

Lemma 1 (Policy Improvement). *Let $\pi^H_{\boldsymbol{s}_t}$ be the optimizer of the optimization objective of Equation* (6)*. When the new policy $\pi_{new}(\cdot|\boldsymbol{s}_t) = \pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_t)$, $V^{\pi_{new}}(\boldsymbol{s}_t) \geq V^{\pi_{old}}(\boldsymbol{s}_t)$ for all $\boldsymbol{s}_t \in S$.*

Proof. Before the proof, we need to show that

$$V^{\pi_{old}}(\boldsymbol{s}_t) \leq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}), \tag{11}$$

because $\pi^H_{\boldsymbol{s}_t}$ is the optimal solution and $V^{\pi_{old}}(\boldsymbol{s}_t) = J^H_{\boldsymbol{s}_t}(\pi_{old}) \leq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$.

Next, we will prove that

$$J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) \leq \mathbb{E}_{\boldsymbol{a}_t \sim \pi^H_{\boldsymbol{s}_t}} \left[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \cdot J^H_{\boldsymbol{s}_{t+1}}(\pi^H_{\boldsymbol{s}_{t+1}}) \right], \tag{12}$$

which follows

$$\begin{aligned}
J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) &= \mathbb{E}_{\boldsymbol{a}_{t:t+H-1} \sim \pi^H_{\boldsymbol{s}_t}} \left[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \cdots + \gamma^{H-1} \cdot r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_{t+H-1}, \boldsymbol{a}_{t+H-1}) + \gamma^H \cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H}) \right] \\
&= \mathbb{E}_{\boldsymbol{a}_{t:t+H-1} \sim \pi^H_{\boldsymbol{s}_t}, \boldsymbol{a}_{t+H} \sim \pi_{old}} \Big[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \cdots + \gamma^{H-1} \cdot r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_{t+H-1}, \boldsymbol{a}_{t+H-1}) \\
&\quad + \gamma^H \cdot r^{\pi_{old}}(\boldsymbol{s}_{t+H}, \boldsymbol{a}_{t+H}) + \gamma^{H+1} \cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H+1}) \Big] \\
&= \mathbb{E}_{\boldsymbol{a}_t \sim \pi^H_{\boldsymbol{s}_t}, \boldsymbol{a}_{t+1:t+H-1} \sim \pi^H_{\boldsymbol{s}_t}, \boldsymbol{a}_{t+H} \sim \pi_{old}} \Big[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \cdot [\, r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}) + \cdots + \gamma^{H-2} \cdot r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_{t+H-1}, \boldsymbol{a}_{t+H-1}) \\
&\quad + \gamma^{H-1} \cdot r^{\pi_{old}}(\boldsymbol{s}_{t+H}, \boldsymbol{a}_{t+H}) + \gamma^H \cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H+1}) \,] \Big] \\
&\leq \mathbb{E}_{\boldsymbol{a}_t \sim \pi^H_{\boldsymbol{s}_t}, \boldsymbol{a}_{t+1:t+H} \sim \pi^H_{\boldsymbol{s}_{t+1}}} \Big[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \cdot [\, r^{\pi^H_{\boldsymbol{s}_{t+1}}}(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}) + \cdots + \gamma^{H-1} \cdot r^{\pi^H_{\boldsymbol{s}_{t+1}}}(\boldsymbol{s}_{t+H-1}, \boldsymbol{a}_{t+H-1}) \\
&\quad + \gamma^H \cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H+1}) \,] \Big] \\
&= \mathbb{E}_{\boldsymbol{a}_t \sim \pi^H_{\boldsymbol{s}_t}} \left[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \cdot J^H_{\boldsymbol{s}_{t+1}}(\pi^H_{\boldsymbol{s}_{t+1}}) \right].
\end{aligned}$$

We finish the proof by applying Equation (11) and iteratively applying Equation (12):

$$\begin{aligned}
V^{\pi_{old}}(\boldsymbol{s}_t) &\leq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) \\
&\leq \mathbb{E}_{\boldsymbol{a}_t \sim \pi^H_{\boldsymbol{s}_t}} \left[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \cdot J^H_{\boldsymbol{s}_{t+1}}(\pi^H_{\boldsymbol{s}_{t+1}}) \right] \\
&\leq \mathbb{E}_{\boldsymbol{a}_t \sim \pi^H_{\boldsymbol{s}_t}, \boldsymbol{a}_{t+1} \sim \pi^H_{\boldsymbol{s}_{t+1}}} \left[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \cdot r^{\pi^H_{\boldsymbol{s}_{t+1}}}(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}) \right] \\
&\vdots \\
&\leq \mathbb{E}_{\boldsymbol{a}_t \sim \pi_{new}} \left[r^{\pi_{new}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + ... \right] \\
&= V^{\pi_{new}}(\boldsymbol{s}_t).
\end{aligned}$$

□

Theorem 2 (Policy Convergence). *Let π_0 be any initial policy. Assuming $|A| < \infty$, if the policy evaluation in Equation* (2) *and the policy improvement with the objective in Equation* (6) *are alternatively carried out, π_0 converges to a policy π_*, with $V^{\pi_*}(\boldsymbol{s}_t) \geq V^{\pi}(\boldsymbol{s}_t)$ for any $\boldsymbol{s}_t \in S$.*

Proof. First, let π_i be the policy at the i-th iteration. Because $V^{\pi_i}(\boldsymbol{s}_t)$ monotonically increases with i and is bounded, the sequence π_i converges to some π_*.

We will next prove that, when the old policy $\pi_{old} = \pi_*$, $V^{\pi_*}(\boldsymbol{s}_t) = J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$. First, because $\pi^H_{\boldsymbol{s}_t}$ is the optimal solution of $J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$, as shown in the proof of Lemma 1, $V^{\pi_*}(\boldsymbol{s}_t) = V^{\pi_{old}}(\boldsymbol{s}_t) \leq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$. Secondly, because π_* is the fixed point, $\pi_{new} = \pi_{old} = \pi_*$ and $V^{\pi_*}(\boldsymbol{s}_t) = V^{\pi_{new}}(\boldsymbol{s}_t) \geq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$, which completes the proof.

Finally, let π be any other policy with $\pi \neq \pi_*$. We have $V^{\pi_*}(\boldsymbol{s}_t) = J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) \geq J^H_{\boldsymbol{s}_t}(\pi)$ and expand the inequality as:

$$
\begin{aligned}
V^{\pi_*}(\boldsymbol{s}_t) \geq J^H_{\boldsymbol{s}_t}(\pi) &= \mathbb{E}_{\boldsymbol{a}_{t:t+H-1}\sim\pi}\Big[r^{\pi}(\boldsymbol{s}_t,\boldsymbol{a}_t) + \cdots + \gamma^{H-1}\cdot r^{\pi}(\boldsymbol{s}_{t+H-1},\boldsymbol{a}_{t+H-1}) + \gamma^H \cdot V^{\pi_*}(\boldsymbol{s}_{t+H})\Big] \\
&\geq \mathbb{E}_{\boldsymbol{a}_{t:t+2H-1}\sim\pi}\Big[r^{\pi}(\boldsymbol{s}_t,\boldsymbol{a}_t) + \cdots + \gamma^{2H-1}\cdot r^{\pi}(\boldsymbol{s}_{t+2H-1},\boldsymbol{a}_{t+2H-1}) + \gamma^{2H} \cdot V^{\pi_*}(\boldsymbol{s}_{t+2H})\Big] \\
&\vdots \\
&\geq \mathbb{E}_{\boldsymbol{a}_{t:\infty}\sim\pi}\Big[r^{\pi}(\boldsymbol{s}_t,\boldsymbol{a}_t) + \cdots\Big] \\
&= V^{\pi}(\boldsymbol{s}_t).
\end{aligned}
$$

□

Lemma 3 (Policy Monotone with Horizon). *Let $\pi^H_{\boldsymbol{s}_t}$ and $\pi^{H+1}_{\boldsymbol{s}_t}$ be the optimizer of $J^H_{\boldsymbol{s}_t}(\pi)$ and $J^{H+1}_{\boldsymbol{s}_t}(\pi)$. Then $J^{H+1}_{\boldsymbol{s}_t}(\pi^{H+1}_{\boldsymbol{s}_t}) \geq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$ for all $H \geq 1$ and $\boldsymbol{s}_t \in S$.*

Proof.

$$
\begin{aligned}
J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) &= \mathbb{E}_{\boldsymbol{a}_{t:t+H-1}\sim\pi^H_{\boldsymbol{s}_t}}\Big[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t,\boldsymbol{a}_t) + \cdots + \gamma^{H-1}\cdot r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_{t+H-1},\boldsymbol{a}_{t+H-1}) + \gamma^H \cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H})\Big] \\
&= \mathbb{E}_{\boldsymbol{a}_{t:t+H-1}\sim\pi^H_{\boldsymbol{s}_t},\boldsymbol{a}_{t+H}\sim\pi_{old}}\Big[r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_t,\boldsymbol{a}_t) + \cdots + \gamma^{H-1}\cdot r^{\pi^H_{\boldsymbol{s}_t}}(\boldsymbol{s}_{t+H-1},\boldsymbol{a}_{t+H-1}) \\
&\quad + \gamma^H \cdot r^{\pi_{old}}(\boldsymbol{s}_{t+H},\boldsymbol{a}_{t+H}) + \gamma^{H+1}\cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H+1})\Big] \\
&\leq \mathbb{E}_{\boldsymbol{a}_{t:t+H}\sim\pi^{H+1}_{\boldsymbol{s}_t}}\Big[r^{\pi^{H+1}_{\boldsymbol{s}_t}}(\boldsymbol{s}_t,\boldsymbol{a}_t) + \cdots + \gamma^{H}\cdot r^{\pi^{H+1}_{\boldsymbol{s}_t}}(\boldsymbol{s}_{t+H},\boldsymbol{a}_{t+H}) + \gamma^{H+1}\cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H+1})\Big] \\
&= J^{H+1}_{\boldsymbol{s}_t}(\pi^{H+1}_{\boldsymbol{s}_t}).
\end{aligned}
$$

□

Theorem 4 (Policy Convergence with Horizon). *Let $\pi^H_{\boldsymbol{s}_t}$ be the optimal solution of $J^H_{\boldsymbol{s}_t}(\pi)$, and $\pi_{new}(\cdot|\boldsymbol{s}_t) = \pi^H_{\boldsymbol{s}_t}(\cdot|\boldsymbol{s}_t)$. π_* denotes the optimal policy. As H increases, $V^{\pi_{new}}$ and $J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$ converge to V^{π_*} for all $\boldsymbol{s}_t \in S$. Specifically, $J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) \geq V^{\pi_*}(\boldsymbol{s}_t) - \frac{\gamma^H \cdot r^{max}}{1-\gamma}$ with r^{max} the maximum of $r^{\pi}(\boldsymbol{s},\boldsymbol{a})$ over all π and $(\boldsymbol{s},\boldsymbol{a}) \in |\mathcal{S}| \times |\mathcal{A}|$.*

Proof. We have show that $V^{\pi_{new}} \geq J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$ in the proof of Lemma 1, hence we only need to prove that $J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t})$ converges to V^{π_*}. We start the proof with the fact that $J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) \geq J^H_{\boldsymbol{s}_t}(\pi_*)$ and expand this inequality as:

$$
\begin{aligned}
J^H_{\boldsymbol{s}_t}(\pi^H_{\boldsymbol{s}_t}) &\geq \mathbb{E}_{\boldsymbol{a}_{t:t+H-1}\sim\pi_*}\Big[r^{\pi_*}(\boldsymbol{s}_t,\boldsymbol{a}_t) + \cdots + \gamma^{H-1}\cdot r^{\pi_*}(\boldsymbol{s}_{t+H-1},\boldsymbol{a}_{t+H-1}) + \gamma^H \cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H})\Big] \\
&= \mathbb{E}_{\boldsymbol{a}_{t:t+H-1}\sim\pi_*}\Big[r^{\pi_*}(\boldsymbol{s}_t,\boldsymbol{a}_t) + \cdots + \gamma^{H-1}\cdot r^{\pi_*}(\boldsymbol{s}_{t+H-1},\boldsymbol{a}_{t+H-1}) \\
&\quad + \gamma^H \cdot V^{\pi_*}(\boldsymbol{s}_{t+H}) + \gamma^H \cdot V^{\pi_{old}}(\boldsymbol{s}_{t+H}) - \gamma^H \cdot V^{\pi_*}(\boldsymbol{s}_{t+H})\Big] \\
&= V^{\pi_*}(\boldsymbol{s}_t) - \gamma^H \cdot \mathbb{E}_{\boldsymbol{a}_{t:t+H-1}\sim\pi_*}\Big[V^{\pi_*}(\boldsymbol{s}_{t+H}) - V^{\pi_{old}}(\boldsymbol{s}_{t+H})\Big] \\
&\geq V^{\pi_*}(\boldsymbol{s}_t) - \gamma^H \cdot \|V^{\pi_*} - V^{\pi_{old}}\|_\infty, \\
&\geq V^{\pi_*}(\boldsymbol{s}_t) - \frac{\gamma^H \cdot r^{max}}{1-\gamma}.
\end{aligned}
$$

□

B. Implementation

B.1 Experiment Setup

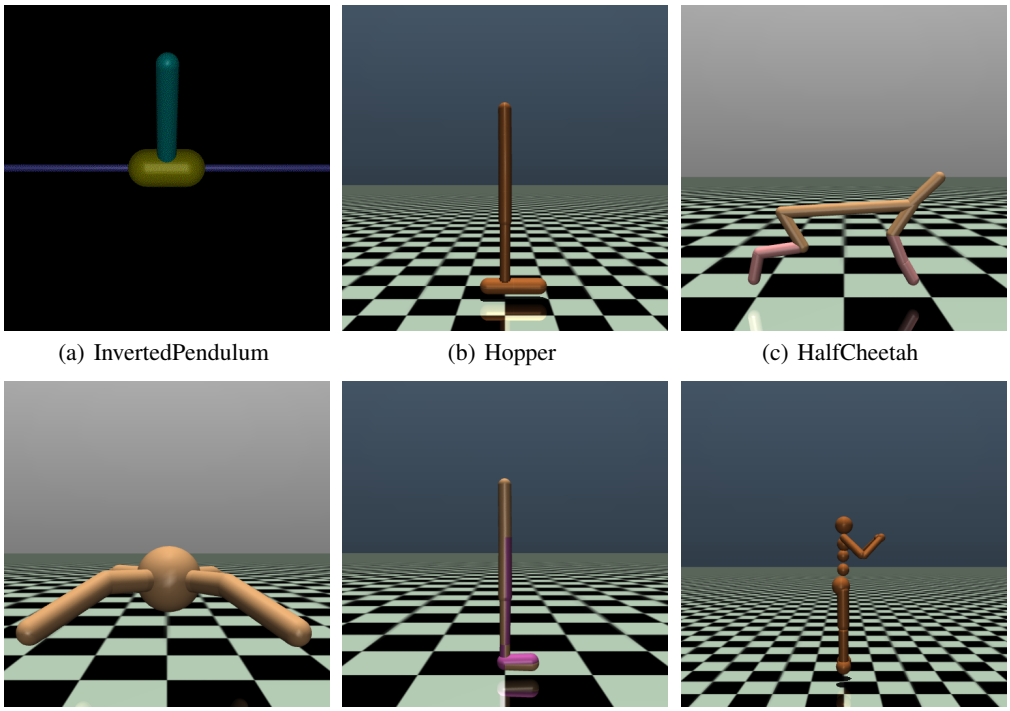
We implement MPDP based on the open-source platform DI-engine [2]. And Table 2 provides the key hyperparameters in MPDP. We follow the original implementations for all baseline algorithms with regard to the reward sum over 1000 steps. We evaluate MPDP along with the baseline algorithms on six continuous control tasks provided in MuJoCo-v2 (Todorov, Erez, and Tassa 2012).

Hyperparameter	Value
Ensemble size	7
Replay buffer size	10^6
Batch size	256
Learning rate	$3 \cdot 10^{-4}$
Threshold u_T	-5
Entropy coefficient α	0.2
Regularization coefficient β	0.5
Maximum horizon H_{max}	25
Policy updates per environment step	20
Environment steps per model training	250

Table 2: Hyperparameter setup for MPDP.

B.2 Experiment Environments

We visualize the six continuous control tasks in MuJoCo-v2 including InvertedPendulum, Hopper, HalfCheetah, Ant, Walker2d, and Humanoid, as shown in Figure 5. The first task InvertedPendulum is designed to control the pole to keep balance, and the other five tasks aim to keep the agent moving forward without falling.



(a) InvertedPendulum (b) Hopper (c) HalfCheetah

(d) Ant (e) Walker2d (f) Humanoid

Figure 5: The screenshots of MuJoCo-v2 simulation environments used in our experiments.

[2] https://github.com/opendilab/DI-engine

C. Counter Example

In Figure 6, we provide a counter example where increasing horizons (H) could result in a lower value. This setting is a simple state transition of MDP problem. We use the circle to denote the state. In the left part, we use the arrow and the number to denote the action and the corresponding reward, respectively. In the right part, we use the arrow to denote the action of π_{old} and the number to denote the value $V^{\pi_{old}}$ of each state.

Note that starting from π_{old} and performing our extended policy improvement on this environment, $H = 3$ leads to a worse value of 20 than $H = 1$ of 10000.

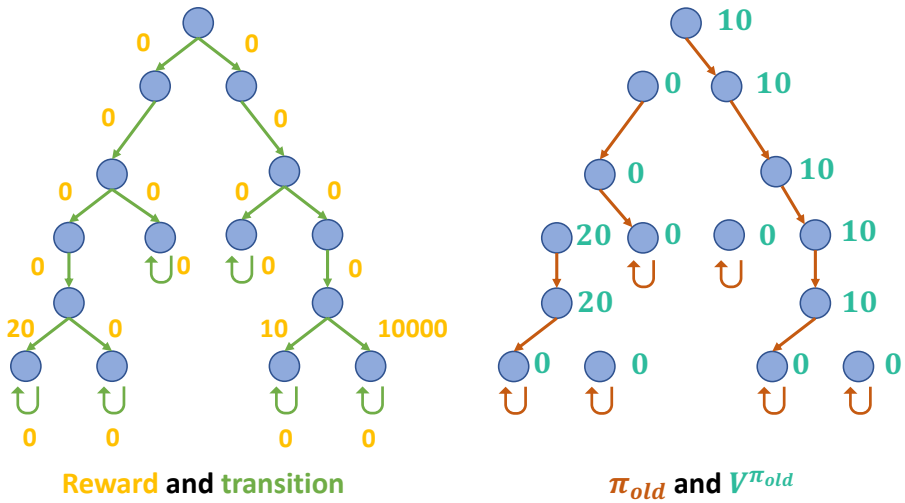


Figure 6: A counter example where $H = 3$ leads to a worse value than $H = 1$.