Look Before Leap: Look-Ahead Planning with Uncertainty in Reinforcement Learning

Abstract

Model-based reinforcement learning (MBRL) has demonstrated superior sample efficiency compared to model-free reinforcement learning (MFRL). However, the presence of inaccurate models can introduce biases during policy learning, resulting in misleading pseudo samples. The challenge lies in obtaining accurate models due to limited diverse training data, particularly in regions with limited visits (uncertain regions). Existing approaches passively quantify uncertainty after sample generation, failing to actively collect uncertain samples that could enhance state coverage and improve model accuracy. Moreover, MBRL often faces difficulties in making accurate multi-step predictions, thereby impacting overall performance. To address these limitations, we propose a novel framework that combines MBRL and MFRL while actively considering uncertainty on both fronts. In the MBRL component, we introduce a k-step lookahead planning approach that incorporates uncertainty to guide action selection. This process involves a trade-off analysis between model uncertainty and value function error, effectively enhancing policy performance. In the MFRL component, we leverage an uncertainty-driven exploratory policy to actively collect diverse training samples, resulting in improved model accuracy and overall performance of the RL agent. Our approach offers flexibility and applicability to tasks with varying state/action spaces and reward structures. We validate its effectiveness through experiments on challenging robotic manipulation tasks and Atari games, surpassing state-of-the-art methods with fewer interactions, thereby leading to significant performance improvements.

1 Introduction

Model-based Reinforcement Learning (MBRL) methods have emerged as powerful approaches that exhibit superior sample efficiency compared to model-free methods, both in theory and practice (Luo et al. 2018; Janner et al. 2019). MBRL leverages an approximate dynamics model of the environment to aid in policy learning (Moerland et al. 2023). However, acquiring a precise model, which plays a critical role in MBRL, presents a challenge. An inaccurate model can introduce biases during policy learning, ultimately leading to the generation of misleading pseudo samples. This prevalent issue is commonly referred to as the model bias problem (Deisenroth and Rasmussen 2011).

Attaining an accurate model presents challenges due to the continuous updates of the policy, causing a shift in the distribution over visited states. The modified distribution often lacks of enough diverse training data for effective approximation, which can result in inaccurate model, particularly in regions with limited visits (uncertain regions). To mitigate this challenge, existing methods employ uncertainty-aware dynamic model techniques, such as utilizing a bootstrapped model ensemble (Chua et al. 2018) or Bayesian RL (Ghavamzadeh et al. 2015; Deisenroth and Rasmussen 2011; McAllister and Rasmussen 2016). However, these approaches passively quantify uncertainty only after the samples have been generated, failing to actively collect uncertain samples that could offer a broader coverage of states and enhance the model’s accuracy.

Moreover, an additional challenge in MBRL is that the learned models often involve predicting multiple steps ahead, such as Model Predictive Control (Camacho and Alba 2013). However, the learning objective of the model primarily focuses on minimizing one-step prediction errors. Consequently, an objective mismatch (Lambert et al. 2020) arises between model learning and model usage. While policy optimization relies on accurate multi-step trajectories (model usage), the model is primarily trained to prioritize the accuracy of immediate predictions (model learning). Although it is possible to generate multi-step predictions using one-step models by iteratively feeding the predictions back into the learned model, this approach can lead to deviations from the true dynamics. Accumulated errors (Machado et al. 2018), stemming from the model’s lack of optimization for long-range predictions, can significantly impact the accuracy of multi-step predictions and compromise overall performance.

To overcome the limitations mentioned above, our paper presents a novel framework that combines the MBRL and MFRL while considering uncertainty on both fronts. In the MBRL component, we introduce a k-step lookahead planning mechanism that accounts for model uncertainty to guide action selection at each step. This planning process operates under an approximate uncertainty-aware model and an approximate value function. We not only present a bound for multiple-step prediction that explicitly factors in both the model uncertainty and the value function approximation error, but our analysis also uncovers an inherent trade-off between these two aspects. Furthermore, our empirical experiments demonstrate leveraging this trade-off can en-
hance policy performance in our framework. To delve into the specifics, during action selection at each step, we simulate k-step roll-outs and sample a number of ‘fantasy’ samples utilizing the uncertainty-aware model. This simulated sampling procedure enables us to construct a tree structure, facilitating a joint optimization that optimizes all actions within the k-step horizon. We then select the first-planned action as the action to be executed in real environment in each step.

In the MFRL component, we leverage an uncertainty-driven exploratory policy (Burda et al. 2018) to actively collect diverse training samples from the environment. This approach mitigates the detrimental effects of insufficient exploration on both policies and forward dynamics models. By gathering more informative data through this exploratory policy, we enhance the learning accuracy of our models, which in turn improves the overall performance of our RL agent. Through the integration of these components, our framework leverages uncertainty on both the planning side (MBRL) and the exploration side (MFRL). This comprehensive approach addresses the limitations of traditional MBRL methods by actively considering and incorporating uncertainty throughout the decision-making process.

In addition to addressing the limitations of previous works, our approach offers several notable advantages. Unlike some prior methods, we do not assume a perfect forward dynamics model or make any specific assumptions about the state or action space. Instead, we employ an approximate forward dynamics model that can be applied to arbitrary state and action spaces. This inherent flexibility makes our approach more adaptable and suitable for a wide range of tasks.

To validate the effectiveness of our approach, we conducted extensive experiments on challenging control tasks in both the MuJoCo simulator and Atari games. The results unequivocally demonstrate that our approach outperforms SOTA with less data. Moreover, the results show that our method is scalable on a wide range of RL tasks with high/low-dimensional states, discrete/continuous actions, dense/sparse rewards.

2 Related Work

2.1 Uncertainty-aware MBRL

One of the key challenges in MBRL is effectively handling uncertainty associated with our model predictions. By doing so, we can identify instances where our predictions may be less reliable when planning based on our model. In statistics, there are two main approaches to uncertainty estimation: frequentist and Bayesian methods. The frequentist approach, as demonstrated by (Chua et al. 2018) and (Fröhlich, Theis, and Hasenauer 2014), employs techniques such as statistical bootstrapping for model estimation. (Sekar et al. 2020) estimates state uncertainty with latent disagreement and generates model training data with exploration policy. Bayesian RL methods have been extensively surveyed by (Ghavamzadeh et al. 2015). Non-parametric Bayesian methods, including Gaussian Processes, have shown great success in modeling estimation, as exemplified by PILCO (Deisenroth and Rasmussen 2011). However, GP’s suffer from computational scalability issues when dealing with high-dimensional state spaces. In recent years, there has been growing interest in Bayesian methods for approximating dynamics using neural networks. Techniques like variational dropout (McAllister and Rasmussen 2016) and variational inference (Depeweg et al. 2016) have been applied to neural network-based Bayesian modeling of dynamics. These approaches aim to address the limitations of GPs and provide efficient uncertainty estimation in high-dimensional state spaces.

2.2 Uncertainty-driven Exploration in MFRL

Uncertainty-driven exploration considers environmental uncertainty and can be classified into two categories. The first category is intrinsic motivation methods, which augment the extrinsic reward from the environment with an exploration bonus, known as intrinsic reward. This additional reward incentivizes the agent to explore states that are more uncertain. Several techniques exist for estimating intrinsic rewards. One approach is count-based exploration, as explored in works such as (Bellemare et al. 2016; Ostrovski et al. 2017; Zhao and Tresp 2019). These methods estimate the number of times a state has been visited, denoted as \( n(s) \), and use it to compute the intrinsic reward. Common formulations include \( 1/n(s) \) or \( 1/\sqrt{n(s)} \). Another approach is prediction-based exploration, such as the RND algorithm introduced by (Burda et al. 2018). Prediction-based methods estimate state novelty based on prediction error. The underlying assumption is that if similar states have
been encountered frequently in the past, the prediction error for a given state should be lower. The second category is uncertainty-oriented methods, as discussed in works by (Osband, Van Roy, and Wen 2016; Azizzadenesheli, Brunskill, and Anandkumar 2018; Moerland, Broekens, and Jonker 2017). These methods typically model uncertainty using the Bayesian posterior of the value function to capture epistemic uncertainty. Additionally, they may also consider aleatoric uncertainty through distributional value functions. By doing so, the agent is encouraged to explore regions with high epistemic uncertainty while avoiding areas with high aleatoric uncertainty.

2.3 Combing MBRL and MFRL

MBRL has posed challenges in complex environments, especially when the agent has limited access to a simulator. To tackle this issue, researchers have been exploring the integration of MBRL and MFRL techniques to improve performance (Hong, Pajarijiam, and Peters 2019). For instance, (Nagabandi et al. 2018) addressed this challenge by training a neural network-based global model and leveraging MPC to initialize the policy for model-free fine-tuning. This combined approach demonstrated significantly enhanced sample efficiency compared to purely model-free methods. Another approach, presented by (Chebotar et al. 2017), combined local model-based algorithms like Linear Quadratic Regulator (LQR) with a model-free framework called path integral policy improvement. However, the use of local models such as LQR is limited to simple environments and requires re-fitting at each iteration, rendering them unsuitable for high-dimensional state spaces like images. A different method, proposed by (Gu et al. 2016) called QProp, introduced a critic trained over an amortized deterministic policy using DDPG as a control variate.

3 Preliminaries and Notations

3.1 Markov Decision Process

A Markov Decision Process (MDP) is represented by a tuple $(S, A, R, P, \mu, \gamma)$ (Sutton and Barto 2018), where $S$ is the set of states, $A$ is the set of actions, $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function, $P : S \times A \times S \rightarrow [0,1]$ is the transition probability function, where $P(s'|s, a)$ is the transition probability from state $s$ to state $s'$ with taking action $a$, $\mu : S \rightarrow [0,1]$ is the initial state distribution and $\gamma$ is the discount factor for future reward. A policy $\pi : S \rightarrow P(A)$ is a mapping from states to a probability distribution over actions and $\pi(a|s)$ is the probability of taking action $a$ in state $s$. We write a policy $\pi$ as $\pi_\phi$ to emphasize its dependence on the parameter $\phi$. A common goal of a MDP is to select a policy $\pi_\phi$ which maximizes the discounted cumulative reward. It is denoted as

$$\max_\phi \ J_R^{\pi_\phi} = \mathbb{E}_{\tau \sim \pi_\phi} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \right],$$

where $\tau = (s_0, a_0, s_1, a_1, \ldots)$ denotes a trajectory, and $\tau \sim \pi_\phi$ means that the distribution over trajectories is following policy $\pi_\phi$. For a trajectory starting from state $s$, the value function is $V(s) = \mathbb{E}_{\tau \sim \pi_\phi} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) | s_0 = s \right]$. The action-value function of state $s$ and action $a$ is $Q(s, a) = \mathbb{E}_{\tau \sim \pi_\phi} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) | s_0 = s, a_0 = a \right]$ and the advantage function is

$$A(s, a) = Q(s, a) - V(s). \tag{2}$$

3.2 Policy Gradient Methods

Policy gradient (Sutton et al. 2000) is a method for finding an optimal policy of a MDP problem. It first calculates gradient of the objective $\nabla J_R^{\pi_\phi} = \mathbb{E}_\tau [\nabla_\phi \log \pi_\phi (a_t | s_t) A_t]$, where $\pi_\phi$ is the current policy under parameter $\phi$ and $A_t$ is the advantage function Eq. (2) at time step $t$. Thereafter, $\phi$ is updated as $\phi = \phi + \eta \nabla J_R^{\pi_\phi}$, where $\eta$ is the learning rate.

Proximal Policy Optimization (PPO) (Schulman et al. 2017) approximates the objective by a first-order surrogate optimization problem to reduce the complexity of TRPO (Schulman et al. 2015), defined as

$$\max_\phi L_{CLIP}^{\pi_\phi}(\phi) = \mathbb{E}_t [\min(r_t(\phi)A_t, \text{clip}(r_t(\phi), 1 - \epsilon, 1 + \epsilon)A_t)],$$

where $r_t(\phi) = \frac{\pi_\phi(a_t | s_t)}{\pi_{\text{old}}(a_t | s_t)}$, $A_t$ is the advantage function, $\text{clip}(\cdot)$ is the clip function and $r_t(\phi)$ is clipped between $[1 - \epsilon, 1 + \epsilon]$.

4 Lookahead Planning with Uncertainty

Traditional MBRL faces the challenge of learning accurate models due to limited and non-diverse training data. In our work, we introduce a novel framework that combines MBRL and MFRL while addressing uncertainty on both fronts. In MBRL, our framework incorporates k-step lookahead planning with uncertainty, which introduces a trade-off between model uncertainty and value function approximation error. In MFRL, we leverage uncertainty-driven exploration to gather more diverse and informative samples. By actively exploring uncertain states, we enhance the learning accuracy of our dynamic models, ultimately improving the overall performance of the RL agent. In Section 4.1, we provide a detailed explanation of our method’s overall framework, including the joint learning of policy, value function, and uncertainty-aware dynamic models. Additionally, in Section 4.2, we delve into the proposed k-step uncertainty lookahead planning approach within MBRL.

4.1 The Framework

As shown in Alg. 1 and Fig. 1, our algorithmic framework demonstrates how our approach combines MBRL and MFRL while considering uncertainty estimates from both sources. We simultaneously train the control policy $\pi_\phi$, value function $V_\sigma$, and uncertainty-aware dynamic model $f_\theta$ using the same data. Our framework comprises three main components. First, we gather interactions $(s, a, s', r')$, where $r'$ is extrinsic (real) reward, by exploring the real environment using an exploratory policy $\pi_\phi$. These real interactions are initially used to calculate the value function $V_\sigma$.
and uncertainty-aware dynamic model \( f_\theta \) (lines 1-3). Next, in each step, we perform simulated k-step lookahead planning using the model \( f_\theta \). We then take the first-planned action and interact with the real environment, appending the gathered interactions to the training dataset (lines 6-17). Finally, we update the control policy \( \pi_\phi \), value function \( V_\sigma \), and dynamics model \( f_\theta \) using batches sampled from the dataset (lines 18-21). The last two iterative processes continue until convergence.

Moreover, within each step (lines 6-17), the framework operates in two layers. In the outer layer (lines 15-17), the agent performs uncertainty-driven MFRL, interacting with the real environment and calculating the intrinsic reward \( r^i \). Each action for MFRL is obtained from the k-step lookahead planning in MBRL, instead of relying on the policy network. In the inner layer (lines 7-14), the agent operates within a simulation using the uncertainty-aware dynamic model \( f_\theta \). We generate virtual k-step samples using the model and evaluate the trajectory. We then plan all actions within the k-step lookahead and select the first-planned action as the action for MFRL. The details of the inner layer operation will be discussed in Sec. 4.2.

In our MBRL approach, we employ Bayesian neural networks to approximate the forward dynamics model and account for model uncertainty. In MFRL, we incorporate intrinsic-motivation exploration, like RND (Burda et al. 2018), combined with PPO to gather more informative data. While our paper focuses on learning uncertainty-aware dynamic models using Bayesian RL and learning exploratory policies with intrinsic-motivation methods, our approach can also be applied using other uncertainty techniques, such as bootstrapped model ensemble methods in MBRL and uncertainty-oriented exploration methods in MFRL.

4.2 K-step Lookahead Planning with Uncertainty in MBRL

In our proposed MBRL framework, lookahead planning is effective when we have access to the dynamic model of the environment. To obtain the uncertainty of the dynamic model, we employ Bayesian neural network \( f_\theta \) to learn the transition function \( P(s_{t+1}|s_t, a_t) \) and the reward function \( R(s_t, a_t, s_{t+1}) \). Using this learned model, we perform trajectory sampling and evaluation through simulation. To select actions at each step, our approach utilizes a k-step joint optimization strategy based on the evaluation of simulated trajectories.

**Trajectory Sampling and Evaluation** To improve trajectory sampling in our framework, we utilize the model-free control policy \( \pi_\phi \) to generate trajectories. We simulate the next k steps, imagining how the environment would unfold \((s_{t+1}, a_{t+1}, s_{t+2}, a_{t+2}, \ldots, s_{t+k+1})\). Here, \((s_{t+1}, s_{t+2}, \ldots, s_{t+k})\) are obtained from the learned uncertainty-aware dynamic model, while \((a_{t+1}, a_{t+2}, \ldots, a_{t+k-1}, a_{t+k})\) are obtained from the model-free control policy \( \pi_\phi \). In each step, we assume that we can draw ‘fantasy’ instances of the next state \( s'_j \), where \( j = 1, 2, \ldots, \infty \), from the Bayesian posterior of the dynamic model. The state instances \( s'_j \) are considered ‘fantasy’ because they are simulated using the uncertainty-aware dynamic model rather than the real environment. In each step, all the \( s_j \) instances \((s'_j, j = 1, 2, \ldots, \infty)\) collectively form the one-step Bayesian posterior.

As depicted in Fig. 1, we construct a discrete scenario tree to visualize this approach. Each branch in a node corresponds to a specific fantasized state drawn from the dynamic model posterior. By considering multiple branches in each node, we explore a diverse range of possible states, which aids the agent in learning a more robust policy. The final action selection is determined by optimizing the evaluation of expected return across all the sampled fantasy trajectories.

We evaluate the expected return of the generated trajectories by calculating the average of the sum of k-step simulated rewards and the value function \( V_\sigma(s_{t+k}) \). This allows us to estimate the expected return considering all the sampled fantasy states and approximated value function which provides a trade-off measurement between model uncertainty and value function error. Our analysis in Sec. 4.2 and empirical experiments in Sec. 5.3 demonstrate the effectiveness of leveraging this trade-off to enhance policy performance in deep RL.

**Action Selection** In the tree structure depicted in Fig. 1, the k-step lookahead planning involves recursively maximizing the Q-function to determine actions and integrating over k steps, as the next states depend on the previous ones. However, the nested expectations become analytically in-
Algorithm 1: Look before leap

Input: a PPO control policy \( \pi_{\theta} \), a value function \( V_{\sigma} \), a Bayesian dynamic model \( f_{\theta} \), number of lookahead step \( k \), number of fantasy samples in each simulated step \( m \), policy and model update frequency \( N \)

1. Run policy \( \pi_{\theta} \) to collect real interactions \( \{(s, a, s', r^c)\} \), \( r^c \) is extrinsic reward
2. Calculate value function \( V_{\sigma} \) for extrinsic reward
3. Learn uncertainty-aware Bayesian dynamic model \( f_{\theta} \)
4. Repeat until converge:
5. Repeat \( N \) times:
6. Current state \( s \)
7. Simulated k-step lookahead planning through \( f_{\theta} \) to choose action:
   8. I: Trajectory sampling, repeat k times:
      9. Get action from policy \( \pi_{\theta} \)
   10. For each action, sample \( m \) fantasy next states with the Bayesian dynamic model \( f_{\theta} \)
   11. II: Trajectory evaluation
      12. Evaluate simulated trajectories with k-step rollouts and extrinsic reward value function \( V_{\sigma} \)
   13. III: Action selection
      14. Compute action according to joint-optimization Eq. 5
      15. Execute the first planned action \( a \), observe real next state \( s' \) and extrinsic reward \( r^c \)
   16. Calculate intrinsic reward \( r^i \)
   17. Append \( \{(s, a, s', r^c, r^i)\} \) to dataset \( D \)
   18. Calculate intrinsic reward advantages \( A^i_t \) and extrinsic reward advantages \( A^c_t \) with \( D \)
   19. Update PPO policy using the combined advantages \( A_t = A^i_t + A^c_t \) with Eq. 3
   20. Update value function \( V_{\sigma} \) for extrinsic reward with \( D \)
   21. Update uncertainty-aware Bayesian dynamic model \( f_{\theta} \) with \( D \)

tractable because each step has a distinct Bayesian posterior based on different states and actions. Moreover, the number of instances in the recursive Bayesian posterior grows exponentially with \( k \), posing computational challenges in considering all possible roll-outs within \( k \) steps. Therefore, we need to rely on numerical integration. To address this, we introduce variables \( m_{t+1} \), \( m_{t+2} \), \ldots, \( m_{t+k} \) to represent the number of fantasy samples from the posterior at each step. With this, we can approximate the planning objective as follows:

\[
Q(s_t, a_t) = \mathbb{E}_{\tau \sim \pi}[\frac{1}{m_{t+1}} \sum_{j_{t+1}=1}^{m_{t+1}} R(s_t, a_t, s_{t+1}^{j_{t+1}}) + \frac{\gamma}{m_{t+2}} \sum_{j_{t+2}=1}^{m_{t+2}} R(s_{t+1}^{j_{t+1}}, a_{t+1}^{j_{t+1}}, s_{t+2}^{j_{t+2}}) + \ldots + \gamma^k V(s_{t+k}^{j_{t+k}}) | [s_t, a_t], \]
\]

where \( V(s) \) is the value function for extrinsic rewards. To overcome the challenge of solving the nested optimization problem, we adopt a different approach by formulating a joint optimization problem of higher dimension. By doing so, we are able to determine the action \( a_t \) at each step, taking into account the virtual consideration of the next \( k \) steps. This alternative approach offers computational efficiency and allows us to effectively address the intractable nested expectations inherent in the k-step problem.

\[
a_{t+1} = \arg \max_{a_{t+1}} \mathbb{E}_{\tau \sim \pi} \left[ \frac{1}{m_{t+1}} \sum_{j_{t+1}=1}^{m_{t+1}} R(s_t, a_t, s_{t+1}^{j_{t+1}}) + \frac{\gamma}{m_{t+2}} \sum_{j_{t+2}=1}^{m_{t+2}} R(s_{t+1}^{j_{t+1}}, a_{t+1}^{j_{t+1}}, s_{t+2}^{j_{t+2}}) + \ldots + \gamma^k V(s_{t+k}^{j_{t+k}}) | [s_t, a_t] \right],
\]

where \( a_{t+1} = a_{t+1}^{j_{t+1}}, j_{t+1} = 1, 2, \ldots, m_{t+1}, a_{t+2} = a_{t+2}^{j_{t+2}}, j_{t+2} = 1, 2, \ldots, m_{t+2}, \) and so on.

Why Lookahead MBRL relies on an approximate dynamics model learned from data collected in the environment. One way to leverage this model is by searching for an action sequence that maximizes the cumulative reward through trajectory optimization. However, a limitation of this approach is the accumulation of errors in the model, which hampers its effectiveness when planning over longer horizons.

To tackle this challenge and enable efficient planning across multiple steps, one approach is to integrate a terminal value function into the planning trajectory. Specifically, given a value function \( V_{\sigma} \), we derive a policy \( \pi_{\theta}^{k,V} \) by maximizing the k-step lookahead objective:

\[
\pi_{\theta}^{k,V}(\sigma_0) = \arg \max_{\sigma_0} \mathbb{E}_{f_{\theta}} \left[ \sum_{t=0}^{k-1} \gamma^t r(s_t, a_t) + \gamma^k V(s_k) \right].
\]

The performance of the overall policy depends on the quality of both the dynamics model \( f_{\theta} \) and the value function \( V_{\sigma} \). To illustrate the benefits of combining model-based k-step rollouts with the value function, we conduct an analysis and provide performance bounds for the k-step lookahead policy \( \pi_{\theta}^{k,V} \) compared to its all-step horizon counterpart, which doesn’t utilize the value function \( V_{\sigma} \), as well as the one-step greedy policy derived from the value function, inspired by (Sikchi, Zhou, and Held 2022).

Theorem 1 (k-step lookahead policy) Suppose \( f_{\theta} \) is an approximate Bayesian dynamics model with uncertainty variation for all states bounded by \( \epsilon_\theta \). Let \( V_{\sigma} \) be an approximate value function for extrinsic rewards satisfying max \( s \in S \), \( | V^*_{\sigma}(s) - V_{\sigma}(s) | \leq \epsilon_\sigma \), where \( V^*_{\sigma}(s) \) is the optimal value function. Let the reward function \( R(s, a) \) be bounded by \( [0, R_{\max}] \), and \( V_{\sigma} \) be bounded by \( [0, V_{\max}] \). Let \( \epsilon_\sigma \) represent the distance from the suboptimality to the global optimum incurred in k-step lookahead optimization, such that
$J^{π^∗} - J^{π^{k,V}} ≤ ϵ_p$, where $J^{π^∗}$ is the global optimal return for the k-step optimization and $J^{π^{k,V}}$ is the results of the suboptimal k-step optimization. Then, the performance of the k-step lookahead policy $π^{k,V}$ can be bounded as follows (proof provided in the appendix):

$$J^{π^∗} - J^{π^{k,V}} ≤ \frac{2}{1 - γ^k} [R_{max} \sum_{t=0}^{k-1} γ^t ϵ_f + γ^k k ϵ_f V_{max} + \frac{ϵ_p}{2} + γ^k ϵ_v]$$

Intuited by Theorem 1, we can view the all-step horizon policy as a special case of $π^{k,V}$, where $V_φ(s) = 0$. When the value function has significant approximation error, the worst-case optimality gap can be large. This suggests that improving the accuracy of the value function leads to a smaller optimality gap. Comparing the k-step lookahead policy to the one-step greedy policy (k=1), we observe that by carefully selecting an appropriate $k$, we can enhance policy performance in our framework.

5 Experiments
In this section, we will explore the following questions:

- Can incorporating k-step lookahead planning in MBRL with uncertainty yield significant benefits?
- Does combining uncertainty in both MBRL and MFRL improve the quality of the dynamic model and enhance policy performance?

5.1 Tasks
Our method is highly versatile and can be applied to a wide range of RL tasks, regardless of the dimensionality of their states, the discreteness or continuity of their action spaces, and the sparsity or density of their rewards. To demonstrate its efficacy, we conducted empirical studies on control tasks using both the MuJoCo physics engine (Todorov, Erez, and Tassa 2012) and the Arcade Learning Environment (ALE) (Bellemare et al. 2013).

The MuJoCo control tasks feature low-dimensional states but a continuous action space, while the ALE tasks involve high-dimensional image-based states and a discrete action space. We evaluated our method on the MuJoCo control tasks with both dense rewards (such as Walker2D, HalfCheetah, and Swimmer) and sparse rewards (such as Hand Manipulate Block). Additionally, we tested our approach on a suite of Atari games, including challenging games with dense rewards (such as Beam Rider, Atlantis, and Freeway) and games with sparse rewards (such as Montezuma’s Revenge, Gravitar, and Venture).

5.2 Baselines
To assess the effectiveness of our k-step lookahead planning with uncertainty in MBRL, we compared our method (K-UMB) against four baselines. The first baseline, MB, represents the standard MBRL approach (Wang et al. 2019; Kaiser et al. 2019). The second baseline, one-UMB, utilizes uncertain MBRL with Bayesian RL technique (McAllister and Rasmussen 2016) and employs a one-step greedy policy. The third baseline, ALL-UMB, applies uncertain MBRL with an all-step horizon policy. Lastly, the fourth baseline, ALL-UMB(op), involves uncertain MBRL with an all-step optimization policy that performs joint optimization for all horizon steps. In practice, given the challenge of optimizing for tasks with a large horizon, we adopted a strategic approach. Instead of considering the complete all-step horizon, we employ a more efficient method by utilizing a $10^k$ step horizon.

To further evaluate the benefits of combining uncertainty in both MBRL (i.e., k-step lookahead uncertainty with Bayesian RL (McAllister and Rasmussen 2016)) and MFRL (i.e., RND (Burda et al. 2018)), we compared our method (UMF+KUMB) against three baselines: MB+MF (no uncertainty), KUMB+MF (k-step uncertainty in MBRL), and MB+UMF (uncertainty-driven exploration in MFRL).

We also compare our work with the state-of-the-art methods, such as Plan To Predict (P2P) (Sekar et al. 2020) and Plan2Explore (Wu et al. 2022), which also learn MBRL policy considering uncertainty. We discuss the benefits of our method and show the results in the appendix. The performance shows that our method can outperform the SOTA. Moreover, we used the PPO (Schulman et al. 2017) algorithm as the base RL method for all the learning methods in our experiments. The code, along with hyper-parameters and implementation details, is available in the appendix as well. Each experiment was conducted 5 times with different random seeds, and we report the average performance with the standard deviation indicated by the shaded area in Fig. 2 and 4.

5.3 Performance
K-step Lookahead in MBRL with Uncertainty In Fig. 2, we present the results of our k-step uncertainty lookahead method applied to three different environments: Walker2D, Hand Manipulate Block, and Beam Rider. These environments were chosen to represent continuous and discrete control tasks with dense and sparse rewards. Additional results can be found in the appendix. Our k-UMB method consistently outperforms the four baseline methods across all environments. The standard MBRL method performs the worst as it does not consider the uncertainty in the dynamic model, leading to inaccurate results. The one-step greedy policy performs better than the standard MBRL method but worse than our k-UMB method. The ALL-UMB and ALL-UMB(op) methods perform worse than our k-UMB method due to the accumulation of uncertainty in the dynamic model approximation. Our results suggest that the k-step uncertainty lookahead approach strikes a balance between the one-step greedy policy and the all-step horizon/optimization policy, and the optimal performance can
be achieved by selecting a suitable value of $k$. The optimal value of $k$ varies across different environments, and we determine it through grid search. The performance of different values of $k$ is provided in the appendix. These findings highlight the effectiveness of incorporating uncertainty in MBRL and the importance of carefully selecting the horizon for optimal performance.

Combination of MBRL and MFRL with Uncertainty

In this experiment, we investigate the benefits of combining uncertainty in both MBRL and MFRL approaches. Our goal is to explore whether incorporating uncertainty-driven exploration in MFRL (e.g., using RND) to collect training data can improve the quality of the learned dynamics model, particularly for tasks with sparse rewards. As shown in Fig. 3, incorporating uncertainty exploration in MFRL leads to lower prediction errors compared to standard MFRL, indicating that uncertainty exploration in MFRL provides more informative and diversified data.

Furthermore, we examine the impact of incorporating uncertainty in both MBRL and MFRL on task performance. In Fig. 4, we demonstrate that uncertainty exploration in MFRL significantly improves performance in tasks with sparse rewards and high-dimensional complex states, such as Montezuma’s Revenge, Hand Manipulate Block and Beam Rider, and slightly improves performance in tasks with dense rewards and low-dimensional states, such as Walker2D. The model-free uncertainty exploration helps the agent gather more informative samples with positive rewards, particularly in tasks with sparse rewards and complex observations. Additionally, our results show that the $k$-step uncertainty lookahead in MBRL improves performance in all tasks, regardless of whether the MFRL incorporates uncertainty exploration or not. In summary, our findings suggest that incorporating uncertainty in both MBRL and MFRL approaches can enhance learning in a variety of environments, especially those with sparse rewards, leading to more robust and effective performance.

6 Conclusion

In this study, we propose a novel framework that integrates MBRL and MFRL, leveraging uncertainty to improve both model accuracy and policy performance. Our framework incorporates k-step lookahead planning with uncertainty, allowing us to select an appropriate value for $k$ and enhance policy performance accordingly. Additionally, the utilization of an uncertainty-driven exploratory policy in the MFRL component facilitates the collection of diverse and informative training samples. The empirical results demonstrate the effectiveness of our framework across a wide range of tasks, surpassing state-of-the-art methods while requiring fewer interactions. However, it is important to acknowledge a potential limitation of our approach, which lies in the computational cost associated with a large number of lookahead steps and ‘fantasy’ samples during k-step lookahead planning. To mitigate this limitation, we can leverage insights from RL trajectory evaluation and policy optimization, thus ensuring the computational feasibility of our framework.
References


