K-Score: Kalman Filter as a Principled Alternative to Reward Normalization in Reinforcement Learning

Zixuan Xia [*1] **Quanxi Li** [*1]

Abstract

We propose a simple yet effective alternative to reward normalization in policy gradient reinforcement learning by integrating a 1D Kalman filter for online reward estimation. Instead of relying on fixed heuristics, our method recursively estimates the latent reward mean, smoothing high-variance returns and adapting to non-stationary environments. This approach incurs minimal overhead and requires no modification to existing policy architectures. Experiments on *LunarLander* and *CartPole* demonstrate that Kalman-filtered rewards significantly accelerate convergence and reduce training variance compared to standard normalization techniques.

1. Introduction

Reinforcement Learning (RL) has emerged as a powerful paradigm for sequential decision-making, enabling agents to learn optimal behaviors through interaction with the environment. Among the broad family of RL algorithms, *policy gradient methods* stand out for their effectiveness in high-dimensional and continuous action spaces (Sutton et al., 1999), where value-based methods often struggle due to discretization challenges or instability in action selection.

Despite their theoretical appeal, policy gradient methods suffer from high variance in gradient estimates, primarily caused by the stochastic nature of both the environment and the policy. High-variance returns can lead to noisy updates, poor sample efficiency, and unstable learning, especially in early-stage training. As a result, numerous variance reduction techniques have been proposed to improve training dynamics and generalization.

[*]Equal contribution [1]Work done during Master studies at the University of Bern. Correspondence to: Zixuan Xia <zixuan.xia@students.unibe.ch>, Quanxi Li <quanxi.li@students.unibe.ch>.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

One commonly adopted strategy is *reward normalization*, where observed returns are standardized via Z-score normalization using running estimates of the mean and standard deviation (Mnih et al., 2016; Schulman et al., 2015a). This heuristic smooths fluctuations in reward signals and improves the conditioning of policy gradient updates. However, such techniques are often sensitive to hyperparameters (e.g., decay rates), require careful tuning, and crucially rely on the assumption that reward statistics remain stationary throughout training—a condition rarely satisfied in dynamic or partially observable environments.

To address these limitations, we explore a *principled and adaptive alternative* grounded in Bayesian estimation theory: Kalman filtering (Kalman, 1960). Originally developed for linear state estimation under uncertainty, Kalman filters have found extensive use in robotics, tracking, and control (Welch et al., 1995), but their potential for reward modeling in reinforcement learning remains underexploited.

In this paper, we propose to replace traditional reward normalization with a lightweight, one-dimensional Kalman filter that recursively estimates the latent mean of the return signal. Our approach models the expected reward as a latent stochastic process subject to both process and observation noise. By fusing prior knowledge with new observations in a statistically optimal manner, the Kalman filter provides a dynamic, uncertainty-aware estimate of the reward baseline. This allows the agent to better cope with noisy feedback, non-stationarity, and abrupt shifts in reward distributions.

Our method is simple to implement, computationally efficient, and architecture-agnostic. It requires no change to the policy network, making it compatible with a wide range of existing policy gradient algorithms such as REINFORCE, Actor-Critic, GAE, and PPO. Furthermore, the Kalman filter naturally adapts to the variability of the learning environment without requiring additional hyperparameter tuning beyond initial noise estimates.

We evaluate our method on two well-established control tasks from OpenAI Gym—*CartPole-v1* and *LunarLander-v2*—which respectively represent low-variance dense reward and high-variance sparse reward settings. Experimental results demonstrate that Kalman-filtered reward estimation leads to smoother training, faster convergence, and improved stability compared to conventional Z-score normalization.

Our main contributions are summarized as follows:

- We propose a novel application of Kalman filtering for reward estimation in policy gradient methods, offering a principled replacement for heuristic normalization strategies.
- We design a lightweight and efficient 1D Kalman filter module that adaptively tracks non-stationary reward signals in online learning settings.
- We empirically validate the effectiveness of our approach across multiple policy gradient algorithms and benchmark environments, showing consistent improvements in convergence speed and stability.

2. Related Work

2.1. Policy Gradient Methods

Policy gradient algorithms constitute a fundamental class of methods in reinforcement learning (RL), particularly well-suited to problems with high-dimensional or continuous action spaces (Sutton et al., 1999). These methods operate by directly optimizing the expected cumulative reward of a parameterized policy $\pi_\theta(a|s)$, where θ denotes the policy parameters. The objective is typically formalized as:

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} r_t\right], \tag{1}$$

where the expectation is taken over trajectories τ generated by the current policy.

The policy gradient theorem (Sutton et al., 1999) provides an elegant expression for the gradient of this objective:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\cdot \hat{A}_t\right], \tag{2}$$

where $\hat{A}_t$ denotes an estimator of the advantage function, typically related to the return-to-go G_t or the temporal difference error.

REINFORCE. The REINFORCE algorithm (Williams, 1992) uses the full return $G_t = \sum_{k=t}^{T} \gamma^{k-t} r_k$ as an unbiased but high-variance estimate of the advantage:

$$\nabla_\theta J(\theta) \approx \mathbb{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\cdot G_t\right]. \tag{3}$$

While conceptually simple, it often suffers from slow convergence due to the high variance of G_t.

Actor-Critic Methods. To address the variance issue, actor-critic methods introduce a value function $V(s)$ to serve as a baseline. The advantage is computed as $\hat{A}_t = G_t - V(s_t)$, where $V(s_t)$ is learned via bootstrapping. This decomposition reduces variance and allows for more stable learning updates.

Generalized Advantage Estimation (GAE). GAE (Schulman et al., 2015b) improves upon actor-critic by using a bias-variance trade-off parameter λ to compute the advantage from temporal difference errors:

$$\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}, \tag{4}$$

where $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$.

Proximal Policy Optimization (PPO). PPO (Schulman et al., 2017) introduces a clipped surrogate objective to constrain policy updates:

$$L^{\mathrm{CLIP}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t,\ \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right], \tag{5}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t|s_t)}$. This technique improves both training stability and sample efficiency.

Recent Trends. Recent advances in policy gradient methods focus on variance reduction and sample efficiency. Notably, Soft Actor-Critic (SAC) (Haarnoja et al., 2018) introduces entropy-regularized learning, encouraging exploration while stabilizing updates. Other research explores trust-region approaches (Wang et al., 2019) and meta-gradient learning (Xu et al., 2018), which dynamically adapt learning signals. Despite these innovations, many approaches still rely on heuristic reward normalization strategies. In this work, we propose a principled alternative based on probabilistic filtering.

2.2. Reward Normalization

Reward normalization is a widely used technique in policy-based RL to reduce the variance of gradient estimates and accelerate convergence. In practice, using raw returns often leads to unstable learning, particularly in environments with sparse or high-variance rewards.

Z-score Normalization. A widely adopted method is Z-score normalization, which standardizes the return-to-go G_t:

$$\hat{G}_t = \frac{G_t - \mu_t}{\sigma_t + \epsilon}, \tag{6}$$

where μ_t and σ_t are computed via exponential moving averages. While simple and effective, this method assumes reward stationarity and does not account for uncertainty in statistics.

PopArt and Scale-Invariant Targets. PopArt (Van Hasselt et al., 2018) dynamically rescales the output of the value function while preserving the scale of the policy gradient. This enables learning under varying reward magnitudes. Similar normalization strategies are used in multi-task settings (Hessel et al., 2019), where reward scales differ substantially across environments.

Meta-Gradient and Adaptive Normalization. Meta-gradient methods (Xu et al., 2018) adapt learning hyperparameters (including normalization factors) using gradients of the learning signal itself. These approaches improve robustness but add computational overhead and architectural complexity.

Probabilistic Alternatives. Most normalization techniques lack a probabilistic foundation and treat each reward equally. In contrast, our method interprets reward estimation as a latent-variable inference problem. By applying Kalman filtering, we produce a smoothed and uncertainty-aware estimate of the return, leading to more stable updates and adaptability to non-stationary environments.

2.3. Kalman Filtering

Kalman filtering is a classic Bayesian technique for online state estimation under noise (Kalman, 1960; Welch et al., 1995). Several works have explored the integration of Kalman filtering into reinforcement learning, primarily for value estimation and uncertainty modeling. Shashua and Mannor (Shashua & Mannor, 2019) introduced a Kalman-based update mechanism for trust region policy optimization, where the filter was used to track the uncertainty in value function approximations. Similarly, Parr et al. (Painter-Wakefield & Parr, 2012) proposed kernel-based reinforcement learning frameworks that leverage Kalman filters for local value estimation in continuous state spaces. More recently, filtering-based approaches have been applied in partially observable settings, where Kalman filtering is used for hidden state inference and belief updating (Van Hasselt et al., 2018).

3. Methods

3.1. Problem Formulation and Motivation

In policy gradient methods, the quality and consistency of the reward signal are critical for stable and efficient learning. However, in many environments—particularly those with sparse or noisy feedback—the reward signal can exhibit high variance, causing unstable updates and slow convergence.

To mitigate this, practitioners often employ reward normalization techniques. A widely used approach is Z-score normalization, which standardizes returns using running estimates of the mean and standard deviation. Despite its empirical success, this method has several well-known limitations:

- It assumes a stationary reward distribution, which rarely holds in practice.
- It requires careful tuning of decay parameters for moving averages.
- It provides no formal treatment of uncertainty or noise in the reward signal.

These issues motivate the need for a more principled, adaptive, and uncertainty-aware approach to reward estimation and normalization. Inspired by Bayesian filtering, we propose to model the true underlying reward signal as a latent stochastic process. Instead of assuming static reward statistics, we treat the reward mean as a hidden variable that evolves over time and must be estimated from noisy observations.

This leads us naturally to the Kalman filter, a recursive estimator that fuses prior predictions and observed data in a statistically optimal manner. In the next section, we describe how we adapt this framework to the reinforcement learning setting.

3.2. Kalman Filtering for Adaptive Reward Estimation

We model the return signal G_t as a noisy observation of a latent reward mean x_t, which evolves over time according to a first-order Markov process. Specifically, we define the process model and observation model as:

Latent Reward Dynamics:

$$x_t = x_{t-1} + w_t, \quad w_t \sim \mathcal{N}(0, Q), \tag{7}$$

$$G_t = x_t + v_t, \quad v_t \sim \mathcal{N}(0, R), \tag{8}$$

where w_t and v_t represent process noise and observation noise, respectively, and Q, R are their variances.

Given these dynamics, the Kalman filter maintains a recursive estimate of the reward mean x_t and its uncertainty P_t via a two-step update:

Predict Step:

$$x_{t|t-1} = x_{t-1}, \tag{9}$$

$$P_{t|t-1} = P_{t-1} + Q, \tag{10}$$

Update Step:

$$K_t = \frac{P_{t|t-1}}{P_{t|t-1} + R}, \tag{11}$$

$$x_t = x_{t|t-1} + K_t(G_t - x_{t|t-1}), \tag{12}$$

$$P_t = (1 - K_t)P_{t|t-1}. \tag{13}$$

Using the filtered estimate x_t and variance P_t, we define the normalized return:

$$\hat{G}_t^{\text{Kalman}} = \frac{G_t - x_t}{\sqrt{P_t + \epsilon}}, \tag{14}$$

where ϵ is a small constant to prevent division by zero.

This formulation provides a dynamically updated baseline that automatically adapts to changes in the reward signal and incorporates uncertainty into the normalization process. Unlike Z-score normalization, which applies uniform decay over time, the Kalman gain K_t dynamically balances prior and observed information based on noise levels.

In practice, this leads to a more robust and data-efficient normalization strategy, especially in the early stages of training or in highly non-stationary environments.

3.3. Integration into Policy Gradient Algorithms

We embed the Kalman filtering mechanism into the policy gradient pipeline at the stage where returns or advantages are typically normalized. The Kalman-filtered reward estimate serves as an adaptive baseline, reducing variance in gradient estimation while maintaining sensitivity to non-stationary rewards.

The vanilla policy gradient objective is:

$$\nabla_\theta J(\theta) = \mathbb{E}_\pi \left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot G_t \right], \tag{15}$$

where G_t is the discounted return. We replace G_t with a normalized variant:

$$\hat{G}_t = \frac{G_t - x_t}{\sqrt{P_t + \epsilon}}, \tag{16}$$

where x_t and P_t are the filtered mean and variance of G_t computed using the Kalman equations. This yields a more stable signal for gradient estimation.

The procedure is outlined in Algorithm 1. Note that our method requires no changes to the policy architecture and incurs minimal computational cost, as the Kalman filter operates on scalar values with constant-time updates.

Algorithm 1 Policy Gradient with Kalman-Normalized Returns

```
1: Initialize: policy parameters $\theta$, Kalman filter state $(x_0, P_0)$, noise parameters $Q, R$
2: for each training iteration do
3:     Sample trajectory $\tau = \{(s_t, a_t, r_t)\}_{t=1}^N$ using policy $\pi_\theta$
4:     Compute discounted returns $G_t = \sum_{k=t}^N \gamma^{k-t} r_k$
5:     for each timestep $t$ in $\tau$ do
6:         Update Kalman filter: $(x_t, P_t) \leftarrow \mathcal{K}.\text{update}(G_t)$
7:         Normalize return: $\hat{G}_t = \frac{G_t - x_t}{\sqrt{P_t + \epsilon}}$
8:     end for
9:     Compute gradient estimate:
           $g = \frac{1}{N} \sum_{t=1}^N \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \hat{G}_t$
10:    Update policy: $\theta \leftarrow \theta + \eta \cdot g$
11: end for
```

This integration results in improved sample efficiency, smoother training curves, and enhanced adaptability to changing environments—qualities especially valuable in real-world RL deployments.

3.4. Theoretical Analysis

To better understand the benefits of our proposed Kalman-based reward normalization, we analyze its convergence behavior and compare it against traditional sample mean estimation under the same noise model. Specifically, we show that the Kalman estimator achieves a favorable mean squared error (MSE), particularly in the early stages of training where data is limited and noise is high.

Sample Mean Estimation. Consider the standard setting where the observed return G_t is drawn from a stationary distribution with true mean r_{true} and additive noise:

$$G_t = r_{\text{true}} + v_t, \quad v_t \sim \mathcal{N}(0, \sigma^2). \tag{17}$$

The sample mean after t episodes is defined as:

$$\bar{G}_t = \frac{1}{t} \sum_{i=1}^t G_i, \tag{18}$$

with expected mean squared error:

$$\text{MSE}(\bar{G}_t) = \mathbb{E}[(\bar{G}_t - r_{\text{true}})^2] = \frac{\sigma^2}{t}. \tag{19}$$

This estimator is unbiased and consistent, but converges slowly and is sensitive to early outliers when t is small.

Kalman Estimator Error Behavior. Now consider the Kalman filter applied to the same sequence $\{G_1, G_2, \dots\}$, assuming the latent mean x_t is constant (i.e., process noise $Q \to 0$). The Kalman update reduces to a form of exponential moving average with adaptive weighting:

$$x_t = x_{t-1} + K_t(G_t - x_{t-1}), \tag{20}$$

where K_t denotes the Kalman gain. The posterior variance P_t evolves as:

$$P_t = (1 - K_t)P_{t|t-1}, \quad \text{with} \quad K_t = \frac{P_{t|t-1}}{P_{t|t-1} + R}. \tag{21}$$

Under mild assumptions, the Kalman filter converges to a steady-state variance:

$$P_\infty = \frac{\sqrt{Q^2 + 4QR} - Q}{2} \approx cQ, \tag{22}$$

for some constant c depending on the ratio Q/R. This indicates that, unlike the sample mean which converges to zero variance, the Kalman estimate maintains a fixed level of uncertainty determined by the noise parameters. This is especially advantageous in online or non-stationary settings, where abrupt changes in the environment may invalidate old samples.

Comparison and Interpretation. To compare the two estimators, we ask: when does the Kalman filter outperform the sample mean in terms of error? The Kalman filter achieves lower MSE than the sample mean when:

$$\frac{\sigma^2}{t} > P_\infty \quad \Rightarrow \quad t < \frac{\sigma^2}{P_\infty}. \tag{23}$$

That is, during the early stages of training (small t), the Kalman filter provides a more reliable estimate by incorporating uncertainty into its updates, rather than averaging potentially noisy samples uniformly.

This is particularly important in reinforcement learning, where agents often begin with high-variance rewards and limited data. In such cases, the Kalman estimator leads to smoother gradients and more stable policy updates, ultimately accelerating convergence.

Summary. Kalman-based reward normalization:

- Achieves lower MSE than the sample mean during early training;
- Maintains a bounded uncertainty that stabilizes updates in non-stationary settings;
- Provides a principled variance-aware alternative to heuristic normalization schemes.

This theoretical analysis supports our empirical findings and highlights the suitability of Kalman filtering for adaptive reward estimation in reinforcement learning.

4. Experiment

4.1. Experimental Setup

We evaluate the effectiveness of our Kalman-based reward normalization across two classic control environments from OpenAI Gym:

- **CartPole-v1**: A simple balancing task with a discrete action space and dense reward. The agent receives $+1$ for every timestep it balances the pole, up to a maximum episode length of 500. The environment is considered solved when the average reward over 100 episodes exceeds 475.
- **LunarLander-v2**: A more challenging control task with a larger discrete action space and sparse, high-variance rewards. The task is considered solved when the agent achieves an average reward above 200.

All algorithms are implemented using PyTorch and trained using the standard OpenAI Gym interface. Each model is trained for a fixed number of episodes or until it reaches the predefined reward threshold. Due to time constraints and in order to reduce randomness to some content, we set up both a training environment and a separate evaluation environment for each baseline. These environments share identical configurations but are initialized with different random seeds. Training is terminated once the agent reaches the predefined `REWARD_THRESHOLD` in the evaluation environment.

For our Kalman reward normalization, we implement two variants:

- **Simple Kalman**: uses fixed process and measurement noise values (Q, R) selected via grid search for each model and environment.
- **Adaptive Kalman**: adaptively updates the measurement noise R_t using an exponential moving average of the squared residuals, with update rule

  $$R_t = \alpha R_{t-1} + (1 - \alpha)(z_t - x_{t-1})^2 \tag{24}$$

  where $\alpha = 0.9$ unless otherwise specified.

These variants are used in both baseline comparisons and ablation studies to evaluate the impact of dynamic versus fixed noise modeling.

4.2. Baselines and Variants

To evaluate the effect of Kalman-based reward normalization, we compare its performance against several widely used policy gradient methods with standard return or advantage estimation techniques. The baselines include:

- **Actor-Critic (AC)**: A standard policy gradient method where the advantage is computed as $A_t = G_t - V(s_t)$ using a learned value function as a baseline.
- **Generalized Advantage Estimation (GAE)** (Schulman et al., 2015b): Computes a bias-variance tradeoff approximation of the advantage by using an exponentially weighted sum of temporal differences.
- **Proximal Policy Optimization (PPO)** (Schulman et al., 2017): A robust on-policy optimization method using clipped surrogate objectives and value function baselines.

We augment each of these methods with our proposed Kalman-based return normalization. Specifically, we apply a 1D Kalman filter to the computed return G_t before it is used in policy gradient or critic updates. This gives rise to two variants:

- **Simple Kalman**: Uses a fixed noise Kalman filter for reward normalization, with (Q, R) selected via grid search.
- **Adaptive Kalman**: Updates the measurement noise R_t dynamically using an exponential moving average of residuals, as described in Section 4.1.

These variants allow us to examine the effect of reward normalization independently of the underlying optimization algorithm, and to study the benefit of adaptive uncertainty modeling in high-variance environments.

4.3. Evaluation Metric

To evaluate and compare convergence speed across different algorithms and normalization strategies, we adopt a threshold-based metric. For each environment, we define a fixed `REWARD_THRESHOLD` and measure the number of training episodes required for the agent to reach or exceed this threshold in the evaluation environment.

- **CartPole-v1**: `REWARD_THRESHOLD` is set to 475, consistent with the standard OpenAI Gym success criterion.
- **LunarLander-v2**: `REWARD_THRESHOLD` is set to 200, indicating successful landing behavior over multiple trials.

For each method, we record the number of episodes needed until the evaluation environment achieves an average return above the threshold. As described in Section 4.1, training is terminated as soon as this condition is met. This setup reflects a practical goal in real-world RL applications: reaching reliable performance with minimal training steps.

The reported metric is the episodes to convergence, which can represent the learning efficiency, rather than final asymptotic performance, and highlights the potential of Kalman-based reward normalization to accelerate policy optimization.

4.4. Results and Analysis

We evaluate our method in terms of convergence speed, measured by the number of episodes required to reach a predefined reward threshold in the evaluation environment. All experiments are conducted using separate training and evaluation environments with different random seeds to assess generalization performance.

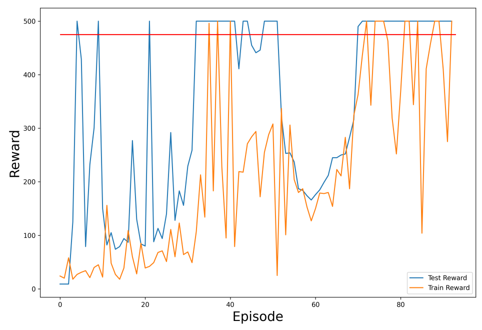

Figure 1: Actor-Critic on CartPole using Z-score normalization. High variance in early training causes slow convergence.

CartPole with Actor-Critic Figure 1 and Figure 2 compare the training performance of the Actor-Critic algorithm on the CartPole environment using two different reward normalization methods: Z-score normalization and Kalman-based normalization. Under Z-score normalization (Figure 1), the policy eventually reaches the target reward threshold of 475, but only after approximately 70 episodes. During the early phase of training (episodes 0–50), both the training and evaluation rewards fluctuate significantly, with frequent spikes and collapses, indicating unstable learning dynamics and sensitivity to noise in the return signal.

In contrast, Kalman-based normalization (Figure 2) enables the agent to achieve the reward threshold much earlier—within 25 episodes—and maintain consistent performance thereafter. The training curve is noticeably smoother, and the test reward stabilizes around the optimal level with minimal variance. This improvement can be attributed to the Kalman filter's ability to incorporate uncertainty into its reward estimation, allowing the policy to ignore early outliers and adapt cautiously in high-variance regimes. These results highlight the effectiveness of Kalman normalization in accelerating convergence and enhancing stability, especially during the critical early stages of training.

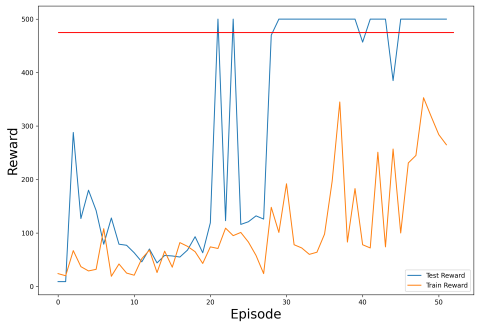

Figure 2: Actor-Critic on CartPole using Kalman normalization. Faster and more stable convergence compared to Z-score.

CartPole with GAE Figure 3 depicts training results using Generalized Advantage Estimation (GAE) with conventional normalization. Similar to the Actor-Critic case, the agent eventually converges but suffers from unstable learning in the initial episodes.

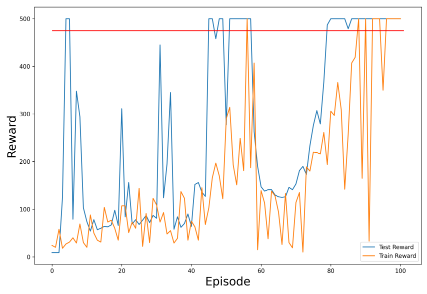

Figure 3: GAE on CartPole using Z-score normalization. Convergence is slow with high variance.

In contrast, Kalman-based normalization (Figure 4) leads to more consistent and rapid performance improvements, with less variance and more stable policy updates.

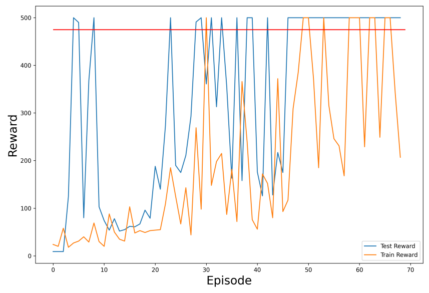

Figure 4: GAE on CartPole using Kalman normalization. Noticeably improved learning dynamics.

LunarLander with Actor-Critic Figure 5 illustrates the training rewards for the LunarLander environment using Actor-Critic with Kalman-based normalization. This environment features sparse and high-variance rewards, making stabilization particularly challenging.

Our Kalman-filtered variant outperforms mean-std normalization by achieving faster convergence and significantly smoother training curves, indicating superior handling of reward noise and non-stationarity.

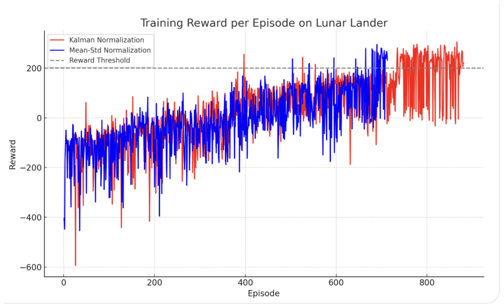

Figure 5: Actor-Critic on LunarLander with Kalman normalization. Achieves faster convergence and lower variance.

PPO on CartPole with Multiple Normalization Methods To evaluate the generality of our method, we apply PPO (Schulman et al., 2017) with three normalization variants: adaptive Kalman, simple Kalman, and Z-score.

Figure 6 shows that adaptive Kalman normalization achieves the fastest convergence with minimal variance. The dynamic adjustment of measurement noise R_t allows the agent to quickly adapt to changing reward patterns.

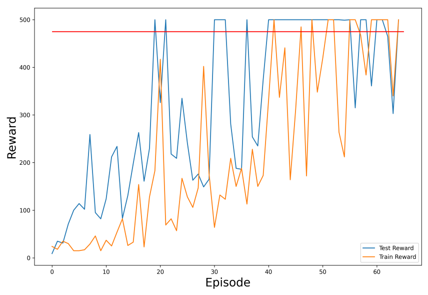

Figure 6: PPO on CartPole with Adaptive Kalman normalization. Most stable and fastest convergence.

Figure 7 shows the baseline PPO performance with conventional normalization. The learning curve is noisier, and the convergence takes considerably longer.

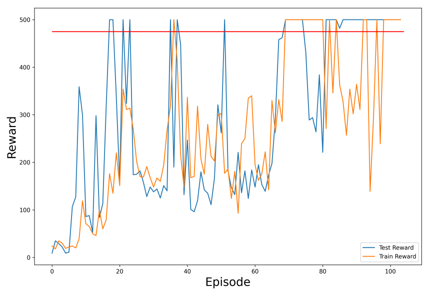

Figure 7: PPO on CartPole using Z-score normalization. Slower and noisier convergence.

Figure 8 displays the result of using a simple Kalman filter with fixed noise parameters. While it improves over Z-score, it underperforms compared to the adaptive Kalman variant, highlighting the benefit of dynamic noise estimation.

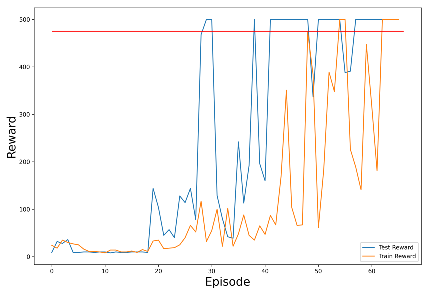

Figure 8: PPO on CartPole with Simple Kalman normalization. Intermediate performance.

Ablation Study: Effect of Process Noise Q We further study the effect of the process noise Q in Kalman filtering by fixing $R = 1$ and varying Q across multiple orders of magnitude. Table 1 summarizes the number of episodes needed to reach the reward threshold in PPO on CartPole.

Table 1: Effect of process noise Q on convergence speed (PPO on CartPole, $R = 1$ fixed)

Q	R	Episodes to Threshold
1×10^{-4}	1	75
1×10^{-3}	1	71
1×10^{-2}	1	69
1×10^{-1}	1	170
1	1	98

These results confirm that smaller values of Q—which imply more trust in the prior estimate—tend to accelerate convergence, as they prevent overreaction to noisy returns. However, overly small Q values may lead to underadaptation in highly non-stationary environments, suggesting a trade-off that can be mitigated by adaptive noise tuning.

Summary Across all experiments, Kalman-based reward normalization leads to faster convergence and more stable learning compared to conventional normalization. The benefits are particularly evident in early training and in environments with high reward variance. The adaptive variant consistently outperforms both fixed Kalman and Z-score baselines, validating the importance of dynamic uncertainty modeling in reinforcement learning.

5. Conclusions, Limitations and Future Work

Conclusions. We proposed a lightweight, principled alternative to reward normalization in policy gradient reinforcement learning based on Kalman filtering. By estimating a smoothed and adaptive baseline from the return signal, our method improves convergence speed and stability across multiple algorithms and environments. Empirical results on CartPole and LunarLander show that Kalman-based normalization outperforms standard mean-std normalization in early training and high-variance scenarios, with particularly strong gains in sample efficiency.

Limitations. Despite these promising results, our approach has several limitations. First, both the simple and adaptive Kalman variants require manual initialization of the process and measurement noise parameters (Q, R). Although these can be tuned per task, the absence of an automated or learned mechanism restricts general applicability. Second, our method uses a one-dimensional Kalman filter applied solely to the scalar return signal G_t. While effective in simple environments, this limits the capacity to capture temporal or structural dependencies present in more complex tasks.

Future Work. Future research can explore multi-dimensional or structured extensions of the Kalman filter, enabling richer representations of uncertainty that consider state dynamics or policy structure. One natural direction is to make the Kalman filter fully differentiable and learnable within an end-to-end training framework, allowing adaptive estimation of (Q, R) directly from data. Additionally, applying Kalman filtering beyond reward normalization—such as to value estimation or advantage calculation—could further improve learning stability. Finally, evaluating this approach in continuous control, offline RL, and safety-critical domains would validate its robustness and broader utility.

Accessibility

We have taken multiple steps to improve the accessibility of this submission. All figures use colorblind-friendly color palettes and include distinct markers for different curves where appropriate. Each figure is accompanied by a descriptive caption that summarizes the main findings. We use standard LaTeX math formatting with consistent notation and avoid excessive inline equations for improved screen reader compatibility. No color is used as the sole means to convey information. The language is written in a clear and concise manner, minimizing jargon where possible. We have also ensured that the PDF passes basic accessibility checks (e.g., embedded fonts, tagged structure where applicable).

Software and Data

To ensure reproducibility, we plan to release the full implementation of our Kalman-based reward normalization framework upon acceptance. This includes training scripts, experiment configurations, and plotting utilities used to generate all figures in the paper. The code is implemented in Python using PyTorch and will be made available as an anonymized GitHub repository or through the Supplementary Material section during the review process. All experiments are conducted on publicly available environments from OpenAI Gym (`CartPole-v1` and `LunarLander-v2`), and no proprietary data or resources are required to reproduce our results.

Impact Statement

This paper presents a theoretically grounded method that enhances the stability and efficiency of reinforcement learning (RL) training by replacing heuristic reward normalization with a principled Kalman filtering approach. Our method is lightweight, broadly applicable, and does not modify existing network architectures, making it practical for a wide range of RL scenarios.

While the primary goal is to advance the field of machine learning, especially in improving sample efficiency and learning stability in noisy or dynamic environments, the potential societal consequences are generally aligned with those of broader RL research. For example, improved stability in RL could accelerate progress in robotics, autonomous systems, and intelligent decision-making under uncertainty.

However, as with many RL advances, care should be taken when deploying such systems in safety-critical or ethically sensitive domains. Increased learning speed and adaptability might inadvertently lead to unforeseen behaviors if not properly constrained or interpreted.

We do not foresee any specific adverse societal consequences from this work. Nevertheless, we encourage practitioners to consider the broader implications of reinforcement learning applications, particularly in areas such as surveillance, automation, and decision-making systems.

References

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

Hessel, M., Soyer, H., Espeholt, L., Czarnecki, W., Schmitt, S., and Van Hasselt, H. Multi-task deep reinforcement learning with popart. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3796–3803, 2019.

Kalman, R. E. A new approach to linear filtering and prediction problems. 1960.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PmLR, 2016.

Painter-Wakefield, C. and Parr, R. Greedy algorithms for sparse reinforcement learning. *arXiv preprint arXiv:1206.6485*, 2012.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015a.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

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

Shashua, S. D.-C. and Mannor, S. Trust region value optimization using kalman filtering. *arXiv preprint arXiv:1901.07860*, 2019.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Van Hasselt, H., Doron, Y., Strub, F., Hessel, M., Sonnerat, N., and Modayil, J. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

Wang, Y., He, H., Tan, X., and Gan, Y. Trust region-guided proximal policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Welch, G., Bishop, G., et al. An introduction to the kalman filter. 1995.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Xu, Z., van Hasselt, H. P., and Silver, D. Meta-gradient reinforcement learning. *Advances in neural information processing systems*, 31, 2018.