Robust Exploration through Generative Replay

Inhyuck Song, Taeyoung Yun, Jaewoo Lee, Jinkyoo Park
KAIST
son9ih@kaist.ac.kr

Abstract

Reinforcement learning (RL) methods typically employ replay buffers to store and leverage past experiences for training policies and value functions. Recent advances in generative models offer a promising alternative to the static replay buffer by capturing online experiences and synthesizing additional transitions beyond them. Guided by conditional generative models, generative replay enables the indirect exploration of unseen states without further environment interactions. However, we identify that existing approaches that utilize conditional generative models struggle to balance between exploration and dynamic plausibility of generated samples, which may exacerbate training instability due to adversarial samples. In this work, we propose a novel framework for **R**obust **E**xploration through **G**enerative **R**eplay (**REGR**). We first define the target distribution as a novelty-tilted distribution to effectively balance between exploration and dynamic plausibility. To sample from the tilted distribution, we fine-tune diffusion models rather than relying on guidance. Through extensive experiments, we confirm that REGR consistently improves performance compared to prior approaches across several continuous control tasks, including sparse-reward environments. Finally, we conduct an in-depth analysis of components such as value-estimation bias and state coverage to validate why REGR achieves robust performance.

1 Introduction

Reinforcement learning has demonstrated remarkable success across diverse domains, including game playing [Mnih et al., 2015, Silver et al., 2016], large language models [Ouyang et al., 2022, Team et al., 2023], drug discovery [Olivecrona et al., 2017, Popova et al., 2018], robotics [Zitkovich et al., 2023, Intelligence et al., 2025], and so on. A key distinction between RL and other machine learning paradigms is that agents learn through their online experiences [Sutton et al., 1998]. To process this continuous stream of data efficiently, most RL methods adopt *replay buffers* to store past experiences and leverage them to train policies and value functions [Lin, 1992].

Recently, several methods have been proposed to utilize generative models as an alternative to the static replay buffer [Lu et al., 2023b, Wang et al., 2024]. This paradigm is referred to as *generative replay*, in which we can sample additional transitions from generative models trained on the collected experiences. Generative replay enables indirect exploration by generating unseen experiences without additional online interactions. Wang et al. [2024] further employs conditional generation models [Ho and Salimans, 2022] to prioritize generating novel experiences.

However, generating exploratory transitions via conditional generation is often suboptimal and susceptible to destabilizing the training of RL agents. As shown in the Figure 1, it is nontrivial to choose an optimal conditioning value and guidance scale that balances exploration and dynamic plausibility. Using a low conditioning value fails to yield novel transitions, whereas excessive conditioning may produce dynamically implausible transitions, which compounds error in learning policy and value function. Furthermore, high guidance scales often lead to severe mode collapse in certain regions, significantly reducing the diversity of generated experiences.

Accepted at ICML 2026 Workshop on RLxF: Reinforcement Learning from World Feedback.

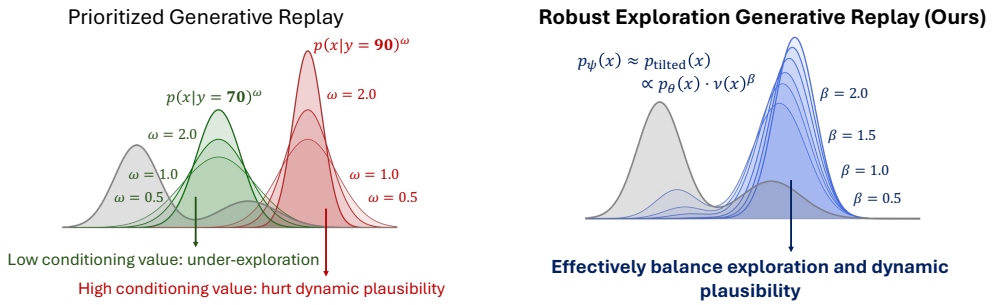

Figure 1: Motivating figure. Conditional generative models struggle to balance exploration and dynamic plausibility due to the non-trivial choice of conditioning value and guidance scale. Our approach tries to generate samples from the tilted distribution by fine-tuning diffusion models to effectively balance exploration and dynamic plausibility.

To address aforementioned issue, we propose **R**obust **E**xploration through **G**enerative **R**eplay **(REGR)**, a new generative replay framework that promotes exploration by generating novel experiences from a novelty-tilted distribution. First, we model the distribution of the replay buffer with a diffusion model p_θ. Then, we define our target distribution as a tilted distribution, $p_{\text{tilted}}(x) \propto p_\theta(x) \cdot \nu(x)^\beta$, where ν is a novelty function such as the curiosity [Pathak et al., 2017] and β is an inverse temperature that controls the greediness of the distribution. As illustrated in Figure 1, sampling from the tilted distribution provides a more principled and intuitive mechanism for balancing between exploration and dynamic plausibility of generated transitions. To sample from the tilted distribution, we fine-tune the pre-trained diffusion model p_θ with a relative trajectory balance (RTB) objective, suggested by Venkatraman et al. [2024]. Finally, we augment the replay buffer with synthesized exploratory transitions to enable the agents to learn from imaginary exploration.

We evaluate REGR on seven continuous control tasks in the OpenAI Gym environments [Brockman et al., 2016] and DeepMind Control Suite [Tassa et al., 2018], including sparse-reward tasks that require extensive exploration. Empirical results demonstrate that our proposed method yields consistent improvements over prior generative replay approaches. Finally, we conduct in-depth analyses of training stability, quality of generated samples, and exploration by measuring Q-normalization bias, dynamic MSE, and state coverage to validate why REGR achieves robust performance.

2 Preliminaries

Reinforcement Learning. We model environment as an infinte-horizon Markov decision process [MDP, Sutton et al., 1998], defined as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(s' \mid s, a)$ for $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ describes the transition dynamics. $r(s, a)$ denotes reward function, and $\gamma \in [0, 1]$ is the discount factor. In online RL, the policy π interacts with the environment and collects transitions $x = (s, a, r, s')$. The Q-function is defined as:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{\pi, P}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r(s'_t, a'_t) \middle| s_t, a_t\right]. \tag{1}$$

The Q-function indicates the expected discounted return after taking action a in state s under the policy π. The primary objective of RL is to learn an optimal policy π^* such that $\pi^*(a|s) = \arg\max_a Q^\pi(s, a)$. For continuous control tasks, we mostly employ actor-critic methods, which directly parametrize policy with neural networks, i.e., $\pi_\phi(a|s) \approx \arg\max_a Q^\pi(s, a)$.

To improve data efficiency, off-policy RL algorithms utilize a replay buffer $\mathcal{D}$ to store past transitions $x = (s, a, r, s')$ collected by a behavior policy. Sampling from $\mathcal{D}$ enables the reuse of useful past experiences for training the policy and value function [Mnih et al., 2013]. For instance, in soft actor-critic [SAC, Haarnoja et al., 2018], we update the parameters of the Q-function to minimize the following soft Bellman error:

$$J(\eta) = \mathbb{E}_{x \sim \mathcal{D}}\left[\frac{1}{2}\left(Q^\pi_\eta(s, a) - \left(r + \gamma\, \mathbb{E}_{a' \sim \pi_\phi(\cdot|s')}\left[Q^\pi_{\bar{\eta}}(s', a') - \alpha \log \pi_\phi(a'|s')\right]\right)\right)^2\right], \tag{2}$$

where $Q^\pi_{\bar{\eta}}$ is the target Q-function and α is temperature that controls the entropy regularization.

Diffusion models. Diffusion models [Sohl-Dickstein et al., 2015, Ho et al., 2020] are a powerful class of generative models designed to approximate a data distribution, i.e., $p_\theta \approx p_{\text{data}}$. They generate new samples by reversing a Markovian forward noising process that gradually adds Gaussian noise to a data sample $x_0 \sim p_{\text{data}}(x_0)$. A *forward process* is defined as $q(x_t \mid x_{t-1}) = \mathcal{N}(\sqrt{\alpha_t} x_{t-1}, (1-\alpha_t)I)$, where $\alpha_t = 1 - \beta_t$ and $\{\beta_t\}_{t=1}^T$ is a predefined noise schedule. A *reverse process*, $p_\theta(x_{t-1} \mid x_t)$, is defined as a parametrized Gaussian, which gradually transforms Gaussian noise into a sample from the data distribution. Specifically, Ho et al. [2020] parameterize a neural network $\epsilon_\theta(x_t, t)$ to predict the injected noise ϵ during forward step. We train the diffusion model by minimizing the standard noise-prediction objective:

$$\mathcal{L}_{\text{DDPM}}(\theta) = \mathbb{E}_{t\sim\text{Unif}(1,T), x_0\sim\mathcal{D}, \epsilon\sim\mathcal{N}(0,I)} \left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2 \right]. \tag{3}$$

For the conditional generation given a label y, we can employ guidance methods [Dhariwal and Nichol, 2021, Ho and Salimans, 2022] to sample from $\tilde{p}(x_0 \mid y)^\omega \propto p(x_0)\, p(y \mid x_0)^\omega$, where ω controls the guidance strength. Specifically, classifier-free guidance [CFG; Ho and Salimans, 2022] jointly trains unconditional and conditional models, $\epsilon_\theta(x_t, t, \emptyset)$ and $\epsilon_\theta(x_t, t, y)$, via random condition dropping during training as follows:

$$\mathcal{L}_{\text{CFG}}(\theta) = \mathbb{E}_{t\sim\text{Unif}(1,T), x_0, y\sim\mathcal{D}, \epsilon\sim\mathcal{N}(0,I), \lambda\sim\text{Bern}(\mu)} \left[\|\epsilon - \epsilon_\theta(x_t, (1-\lambda)y + \beta\lambda, t)\|^2 \right]. \tag{4}$$

For sampling, it extrapolates from the unconditional prediction toward the conditional prediction:

$$\tilde{\epsilon}_\theta(x_t, t, y) = \epsilon_\theta(x_t, t, \emptyset) + \omega\big(\epsilon_\theta(x_t, t, y) - \epsilon_\theta(x_t, t, \emptyset)\big). \tag{5}$$

Therefore, we need to specify a conditioning value y and guidance scale w to control the sharpness of the sampling distribution.

Diffusion fine-tuning. Given a pre-trained diffusion model $p_\theta(x_0)$, it is often necessary to sample from the unnormalized tilted distribution:

$$p_{\text{tilted}}(x_0) \propto p_\theta(x_0) R(x_0)^\beta, \tag{6}$$

where R represents the constraint or objective function we want to maximize, and β is an inverse temperature that controls the greediness. While one can use guidance-based methods like a classifier(-free) guidance, the measure $R(x_0)$ is mostly defined only on the data space, not on intermediate transition spaces, which implies that we cannot learn an unbiased estimator of $p(y|x_t)$ without several modifications [Chung et al., 2022, Lu et al., 2023a].

RTB [Venkatraman et al., 2024] fine-tunes the diffusion model p_θ to approximate the tilted distribution of eq. (6). We update the parameters of p_ψ, initialized with p_θ, with the following objective:

$$\mathcal{L}_{\text{RTB}}(\tau = (x_T, x_{T-1}, \cdots, x_0); \psi) = \left[\log \frac{Z_\psi}{R(x_0)^\beta} + \sum_{t=1}^{T} \log \frac{p_\psi(x_{t-1}|x_t)}{p_\theta(x_{t-1}|x_t)} \right]^2, \tag{7}$$

where Z_ψ is an estimator of the partition function. As the RTB objective does not need to estimate the value of R in intermediate transition spaces, we can obtain an unbiased estimator that approximates the tilted distribution. Notably, the RTB objective accommodates samples from arbitrary distributions. This flexibility enhances the distributional coverage of the diffusion model by facilitating training on diverse, multi-modal data distributions.

3 Method

In this section, we propose **Robust Exploration through Generative Replay (REGR)**, a framework designed to facilitate robust exploration by providing the RL agent with novel, synthesized experiences. First, we train a diffusion model that approximates the empirical distribution of the replay buffer. Then, by fine-tuning the diffusion model via RTB objective, we effectively shift the generative distribution toward a high-novelty region without deviating excessively from the underlying data manifold. Finally, we generate novel experiences from the fine-tuned model and store them in a synthetic replay buffer. Figure 2 provides an overview of the proposed method.

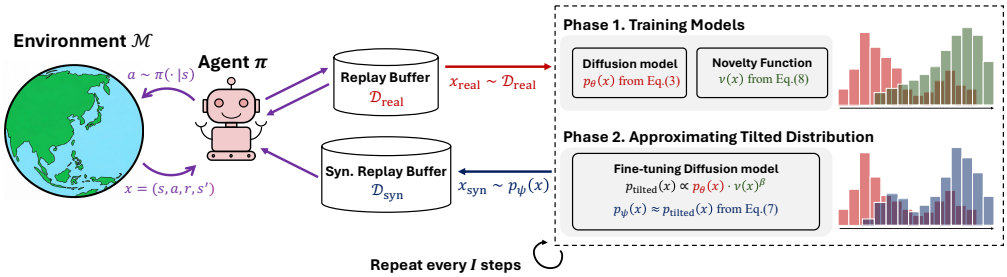

Figure 2: Overview of the proposed method. We first train the diffusion model and the novelty function with the transitions from the replay buffer. Then, we define a tilted distribution and fine-tune the diffusion model to approximate the tilted distribution. Finally, we generate transitions from the diffusion model and store them to synthetic replay buffer.

Algorithm 1 REGR: Robust Exploration through Generative Replay

Input: Synthetic data ratio α, total environment steps K, update interval I.
Initialize: RL agent π, RL environment $\mathcal{M}$, Real replay buffer $\mathcal{D}_{\text{real}} \leftarrow \emptyset$, Synthetic replay buffer $\mathcal{D}_{\text{syn}} \leftarrow \emptyset$, Generative model p_θ, Novelty function ν.

for $k = 1, \ldots, K$ **do**
  Collect transitions $x_{\text{real}} = (s, a, r, s')$ with π in the $\mathcal{M}$ and update $\mathcal{D}_{\text{real}} \leftarrow \mathcal{D}_{\text{real}} \cup \{x_{\text{real}}\}$
  if $k \,\%\, I = 0$ **then**
    Initialize p_θ and $\mathcal{D}_{\text{syn}} \leftarrow \emptyset$
    Train p_θ with $\mathcal{D}_{\text{real}}$ using Equation (3)
    Initialize $p_\psi \leftarrow p_\theta$ and fine-tune p_ψ with $\mathcal{D}_{\text{real}}$ and ν using Equation (7)
    Generate a batch of x_{syn} from p_ψ and update $\mathcal{D}_{\text{syn}} \leftarrow \mathcal{D}_{\text{syn}} \cup \{x_{\text{syn}}\}$
  end if
  Train π on mini-batch samples from $\mathcal{D}_{\text{real}} \cup \mathcal{D}_{\text{syn}}$ mixed with ratio α
end for

3.1 Motivation

Recent advances in deep generative models [Ho et al., 2020, Karras et al., 2022] enable the synthesis of novel transitions by modeling the distribution of the replay buffer [Lu et al., 2023b]. However, naive replication of the replay buffer is often suboptimal [Lee et al., 2024]. To accelerate the training of an RL agent, a more effective approach is to generate highly informative transitions that encourage the agent to learn from underexplored or critical regions of the state space.

PGR [Wang et al., 2024] attempts to enhance exploration through novelty-conditioned sampling via CFG [Ho and Salimans, 2022]. Specifically, PGR samples transitions from a guided distribution $p_{\text{CFG}}(x|y)^\omega \propto p_{\text{data}}(x)p(y \mid x)^\omega$, where y denotes a novelty measure from Pathak et al. [2017], and ω is the guidance scale. Despite its efficacy, we observe that choosing appropriate values for y and ω to balance exploration towards novel regions and dynamic plausibility is notoriously difficult in practice. Employing a high guidance scale or extreme conditioning values often degrades the diversity of generated samples and produces infeasible transitions, exacerbating training instability. Conversely, using moderate values degenerates into behavior similar to unconditional generation, hindering exploration. Consequently, there remains a critical need for an advanced generative replay mechanism that ensures robust exploration.

3.2 REGR

REGR consists of three core phases: 1) pre-training the diffusion model, 2) fine-tuning the diffusion model to match a novelty-tilted distribution, and 3) sampling from the tilted distribution to generate high-novelty transitions. In the following section, we detail each step. The pseudocode for REGR is presented in Algorithm 1.

Training models. Every I environment steps, we train an unconditional diffusion model p_θ to serve as a prior transition generator. Our diffusion prior model approximates the distribution of transitions

$x = (s, a, r, s')$ within the empirical replay buffer $\mathcal{D}_{\text{real}}$ by minimizing Equation (3). Additionally, we optionally train a forward dynamics model f_ξ to measure the novelty of transitions.

Approximating Tilted Distribution. Following pre-training stage, we define the target distribution as a novelty-tilted distribution, $p_{\text{tilted}}(x) \propto p_\theta(x) \cdot \nu(x)^\beta$, where ν is a novelty function that shifts the distribution towards exploratory regions of the state space. Unlike conditional sampling, this approach eliminates the need to choose an arbitrary novelty value target. To approximate the target distribution, we fine-tune the prior diffusion model p_θ with the RTB objective outlined in Equation (7). Notably, the RTB framework allows us to fine-tune the diffusion model using samples drawn from both the empirical replay buffer and the distribution of the current diffusion model. We further empirically evaluate the benefits of self-generated samples in Section 4.3.

Sampling After fine-tuning, the resulting diffusion sampler p_ψ approximates the novelty-tilted distribution. We then synthesize a large batch of transitions (e.g., 1M samples) using the fine-tuned diffusion model and append them to the synthetic replay buffer $\mathcal{D}_{\text{syn}}$. During online training, the RL agent π samples a mini-batch of transitions from both the replay buffer and the generative replay with a synthetic data ratio α. We repeat this procedure periodically by the end of policy training.

3.3 Novelty Functions

To demonstrate that REGR generalizes across different novelty functions, we evaluate two representative intrinsic reward measures: *Curiosity* [Pathak et al., 2017] and *RND* [Burda et al., 2019]. These measures and their respective training losses are defined as:

$$\nu_{Curiosity}(x;\xi) = \frac{1}{2}\left\|s' - f_\xi(s,a)\right\|^2, \quad \text{where} \quad \mathcal{L}(\xi) = \mathbb{E}_{x\sim\mathcal{D}_{\text{real}}}\left[\left\|s' - f_\xi(s,a)\right\|^2\right]. \tag{8}$$

$$\nu_{RND}(x;\zeta) = \frac{1}{2}\left\|g_\zeta(s') - g_{\zeta^\star}(s')\right\|^2, \quad \text{where} \quad \mathcal{L}(\zeta) = \mathbb{E}_{x\sim\mathcal{D}_{\text{real}}}\left[\left\|g_\zeta(s') - g_{\zeta^\star}(s')\right\|^2\right], \tag{9}$$

where ζ^* denotes the fixed random network parameters. While we evaluate REGR using these two representative measures, our framework is agnostic to the choice of function and generalizes to other functions, such as those based on reachability [Savinov et al., 2018] or variational information maximization [Houthooft et al., 2016].

4 Experiments

In this section, we present experimental results validating our method across various online RL tasks. Furthermore, we conduct several in-depth analyses to explain why our method yields robust improvements over generative other baselines.

4.1 Experimental Setup

Environments. We evaluate REGR on several continuous control tasks from the DeepMind Control suite benchmark [DMC; Tunyasuvunakool et al., 2020] (`cheetah-run`, `reacher-hard`, `quadruped-walk`), as well as OpenAI Gym environments [Brockman et al., 2016] (`HalfCheetah`, `Hopper`, `Walker2d`). We follow the online RL evaluation protocol used in prior work by Lu et al. [2023b] to ensure a fair comparison. In addition to these conventional tasks, we evaluate our method on `finger-turn-hard` to explicitly assess the exploration capabilities of generative replay in the sparse-reward environment.

Baselines. To validate the improvements gained by our framework, we compare REGR against the two recent generative replay baselines: SER [Lu et al., 2023b] and PGR [Wang et al., 2024]. SER employs an unconditional diffusion model to generate stochastic imaginary transitions, whereas PGR utilizes a conditional diffusion model to steer generated transitions toward novel regions. For the base RL algorithm, we employ SAC [Haarnoja et al., 2017] and increase the update-to-data (UTD) ratio to 20, following prior literature. Except for the method used to construct the generative replay, we fix all other components, including the number of generated samples, the generative model update interval, and the synthetic data ratio. Please refer to Appendices A and B for more details regarding training and evaluation hyperparameters.

Table 1: Performance comparison of REGR over continuous control benchmarks. We report the mean and one standard deviation of average returns over five seeds.

Environment	MuJoCo-100K			DMC-100K			Sparse Reward-300K
	HalfCheetah	Hopper	Walker2d	Cheetah-Run	Reacher-Hard	Quadruped-Walk	Finger-Turn-Hard
SAC	5316.90 ± 417.58	478.01 ± 130.54	663.09 ± 362.44	276.64 ± 122.18	502.15 ± 248.44	240.35 ± 104.35	441.18 ± 134.00
SER	6221.22 ± 256.28	2077.48 ± 835.88	1485.95 ± 1351.88	308.95 ± 134.61	783.66 ± 112.03	658.34 ± 307.95	312.62 ± 145.24
PGR (Fixed)	6286.16 ± 981.34	1036.00 ± 656.22	2439.14 ± 889.97	410.55 ± 51.28	575.24 ± 152.54	856.27 ± 84.00	218.42 ± 116.04
PGR (Grid Search)	7216.21 ± 907.76	2379.49 ± 954.32	3215.65 ± 946.35	410.55 ± 51.28	679.94 ± 186.52	856.27 ± 84.00	336.14 ± 78.93
REGR (Fixed)	**7390.86 ± 592.30**	1432.76 ± 817.63	3049.81 ± 497.27	405.54 ± 121.46	763.06 ± 138.19	**910.25 ± 57.73**	429.78 ± 256.68
REGR (Grid Search)	**7390.86 ± 592.30**	**2895.03 ± 749.79**	**3267.16 ± 1178.91**	**514.55 ± 88.81**	**840.77 ± 106.14**	**910.25 ± 57.73**	**583.30 ± 145.76**

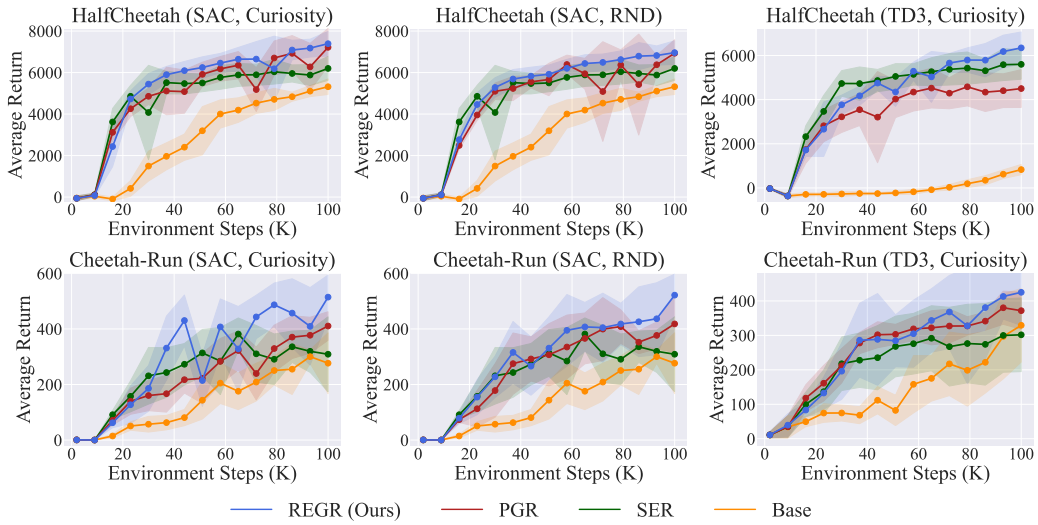

Figure 3: Main experiment results on continuous control tasks. **(Left)** Experiment results with SAC and Curiosity novelty function. **(Middle)** SAC with RND novelty function. **(Right)** TD3 with Curiosity novelty function. For all experiments, we run five seeds and report the mean and one standard deviation. Please refer to Figure 8 for more visualization results.

4.2 Results

Main experimental results. Table 1 summarizes the final performance across diverse continuous control tasks. We evaluate two settings: *Fixed*, which uses a single, best-performing hyperparameter configuration shared across all environments, and *Grid Search*, which utilizes environment-specific hyperparameters selected via grid search. As shown in the table, REGR consistently outperforms prior generative replay baselines across diverse tasks in both settings. We also visualize the learning curve of evaluated methods in Figure 3 and Figure 8.

Robustness on novelty function and base algorithms. By default, we use *Curiosity* as the novelty function to guide generative replay toward underexplored regions. To demonstrate the robustness of REGR to the choice of novelty function, we also conduct experiments using *RND* [Burda et al., 2019] as a novelty function. As shown in Table 2, REGR consistently outperforms PGR when attached with the RND function.

Furthermore, we evaluate our framework using TD3 [Fujimoto et al., 2018] as a base RL algorithm to verify that the benefits of REGR are not exclusive to SAC. As shown in Table 3, REGR consistently surpasses prior generative replay baselines, underscoring the broader efficacy of sampling transitions from the novelty-tilted distribution with generative models.

Trade-off between novelty and dynamic plausibility. We define a novelty-tilted distribution as a target sampling distribution of generative replay to improve the trade-off between novelty and dynamic plausibility. To assess the agent's exploratory capability, we estimate the state coverage by computing the state entropy using a non-parametric estimator [Singh et al., 2003]. Specifically, we sample 4K transitions from the replay buffer at the end of the training and compute the state entropy. Please refer to Appendix C for details on how to measure the state coverage.

Table 2: Performance comparison of REGR with a SAC agent using RND as the novelty function. We report the mean and one standard deviation of average returns over five seeds.

Environment	MuJoCo-100K	DMC-100K
	HalfCheetah	Cheetah-Run
SAC	5316.90 ± 417.58	276.64 ± 122.18
SER	6221.22 ± 256.28	308.95 ± 134.61
PGR (RND)	6930.22 ± 653.06	418.11 ± 103.32
REGR (RND)	**6960.52 ± 637.93**	**521.72 ± 84.62**

Table 3: Performance comparison of REGR with a TD3 agent using Curiosity as the novelty function. We report the mean and one standard deviation of average returns over five seeds.

Environment	MuJoCo-100K	DMC-100K
	HalfCheetah	Cheetah-Run
TD3	831.44 ± 231.02	329.26 ± 106.65
SER	5597.98 ± 673.99	301.68 ± 106.83
PGR (Curiosity)	4501.98 ± 856.43	371.77 ± 53.99
REGR (Curiosity)	**6344.89 ± 719.82**	**425.07 ± 96.04**

Table 4: Comparison of state coverage across DMC suite tasks. Values represent the mean ± standard deviation of average state entropy over five seeds after training. State entropy was computed using 4K transitions randomly sampled from the agent's real replay buffer.

Environment	MuJoCo-100K			DMC-100K			Sparse Reward-300K
	HalfCheetah	Hopper	Walker2d	Cheetah-Run	Reacher-Hard	Quadruped-Walk	Finger-Turn-Hard
SER	1.838 ± 0.091	0.865 ± 0.012	1.731 ± 0.063	1.810 ± 0.013	0.525 ± 0.052	2.953 ± 0.036	0.766 ± 0.015
PGR (Curiosity)	1.749 ± 0.104	0.762 ± 0.029	1.630 ± 0.106	1.795 ± 0.032	0.521 ± 0.035	**2.992 ± 0.022**	0.742 ± 0.045
REGR (Curiosity)	**1.931 ± 0.095**	**0.907 ± 0.030**	**1.884 ± 0.025**	**1.827 ± 0.071**	**0.544 ± 0.024**	2.969 ± 0.032	**0.808 ± 0.015**

As shown in Table 4, REGR achieves higher state entropy compared to both SER and PGR. This underscores that sampling transitions from the novelty-tilted distribution via fine-tuning of diffusion models effectively drives the agent to explore previously unseen regions of the state space.

We also investigate whether REGR preserves adherence to the underlying transition dynamics while promoting exploration. Following prior work [Lu et al., 2023b, Lee et al., 2024], we measure the dynamic mean-squared error (MSE) of the generated samples against the true transition dynamics f and reward function r. Given a generated transition $x = (s, a, r, s')$, we compare s' and r with the true next state $s'_{\text{true}} = f(s, a)$ and reward $r_{\text{true}} = r(s, a)$: $0.5 \times \left(\|s'_{\text{true}} - s'\|^2 + (r'_{\text{true}} - r')^2\right)$. As depicted in Figure 4, REGR produces a distribution similar to SER, demonstrating that REGR maintains high dynamic plausibility. Conversely, PGR generates transitions with slightly high violations in terms of dynamic MSE, which may induce unstable training.

Training stability. Generating dynamically implausible samples via excessive guidance often leads to severe training instabilities [Lee et al., 2024]. To analyze this effect, we compute the mean and standard deviation of the normalized Q-bias during training. Please refer to Appendix D for details on how to compute the bias. As illustrated in Figure 5, PGR occasionally exhibits extremely high bias, which severely degrades overall training performance. In contrast, REGR consistently maintains a relatively low bias, underscoring the robustness of our framework.

4.3 Ablation Studies

Analysis on β. The inverse temperature β controls the greediness of the novelty-tilted distribution. To analyze the effect of β, we evaluate the novelty of samples generated by the diffusion model during the fine-tuning stage as β varies. As illustrated in Figure 6, we observe that increasing β consistently leads to transitions toward high-novelty regions. Notably, we do not observe severe degradation in novelty or over-exploration when varying β, indicating that framing the target distribution as a novelty-tilted distribution inherently provides robust exploration to the RL agent.

Fine-tuning approaches. As discussed in Section 3.2, the RTB objective allows us to fine-tune the diffusion model using samples not only from the real replay buffer but also from the current diffusion model. To verify its effectiveness, we evaluate a variant of REGR that fine-tunes the diffusion model solely on the samples from the replay buffer. As shown in Figure 7, while relying on the replay buffer demonstrates competitive results, actively incorporating self-generated samples from the current diffusion model during the fine-tuning phase consistently achieves higher performance.

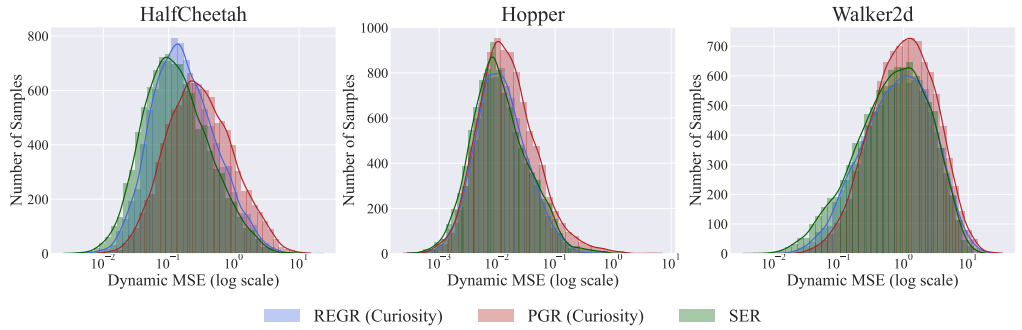

Figure 4: Histogram of dynamic MSE over 10K generated transitions from different generative replay approaches across OpenAI gym environments.

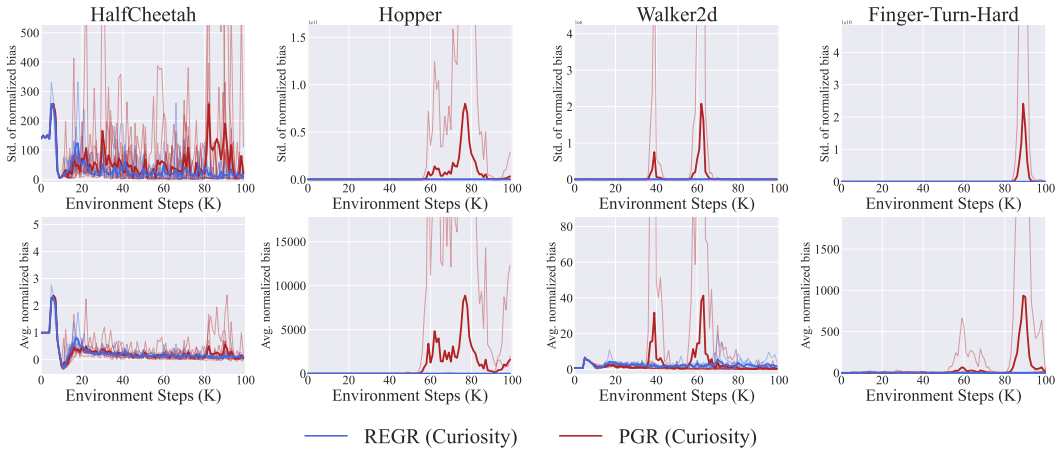

Figure 5: Stability results of PGR and REGR across various hyperparameters. In each environment, the translucent curves denote the normalized average bias and its standard deviation for individual hyperparameter configurations in 8, computed from five random seeds, while the solid curves represent the mean across all configurations. Experiments are conducted on three OpenAI Gym environments and one sparse-reward DMC task.

5 Related Works

Exploration in RL. Exploration methods can be roughly categorized into three paradigms: random-based exploration, Boltzmann exploration, and intrinsic reward-based exploration. Random-based exploration methods, such as ϵ-greedy [Sutton et al., 1998, Mnih et al., 2013], parameter noising [Fortunato et al., 2018, Plappert et al., 2018], or action perturbation [Lillicrap et al., 2016], are simple yet effective, but they often underperform in high-dimensional, complex control tasks. Boltzmann exploration, based on Maximum Entropy RL [Ziebart et al., 2008], significantly enhances model-free RL agents by incorporating entropy regularization [Haarnoja et al., 2017, 2018, Ishfaq et al., 2025]. Lastly, intrinsic reward-based methods augment the original reward with a novelty measure quantified via density estimation [Tang et al., 2017, Bellemare et al., 2016, Ostrovski et al., 2017], variational inference [Houthooft et al., 2016], state uncertainty [Pathak et al., 2017, Burda et al., 2019, Mutti et al., 2021, Seo et al., 2021, Mutti et al., 2022], or information gain [Sukhija et al., 2023, 2024]. These approaches demonstrate success, particularly on sparse-reward control tasks, by emphasizing novel experiences. In this work, we focus on exploration via generative replay, which enables indirect exploration by generating novel transitions.

Generative Replay Buffer. Given the success of generative models in capturing high-dimensional, complex distributions across diverse domains [Ho et al., 2020, 2022], several strategies have been studied to replace the standard replay buffer with generative models. Lu et al. [2023b] first introduced SER, which utilizes diffusion models [Ho et al., 2020, Karras et al., 2022] to approximate the distribution of the replay buffer and improve the sample efficiency of online RL methods by increasing

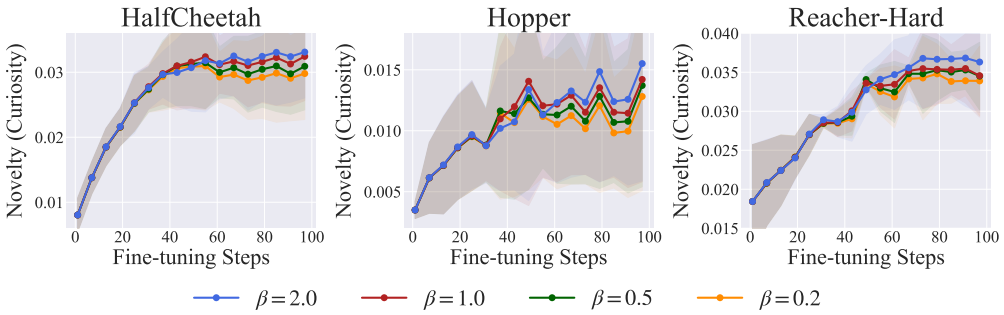

Figure 6: Ablation study on β. Experiments are conducted in DMC and OpenAI Gym environments using three random seeds.

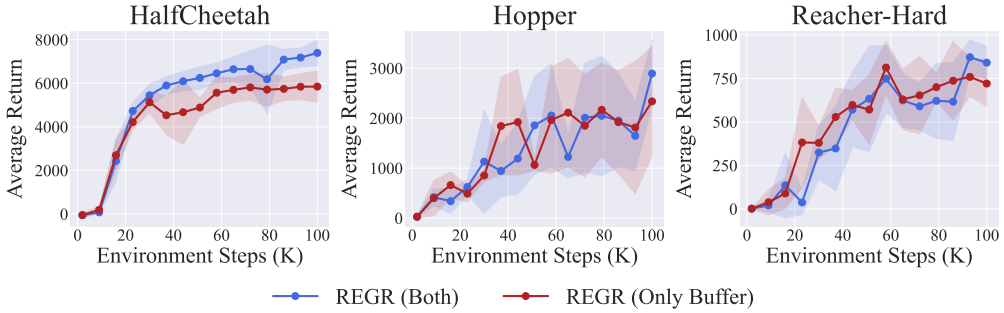

Figure 7: Ablation study on fine-tuning approaches. Experiments are conducted in DMC and OpenAI Gym environments using five random seeds.

the UTD ratio. PGR [Wang et al., 2017], which is an extension of SER, leverages classifier-free guidance [Ho and Salimans, 2022] to steer the diffusion model synthesize novel transitions under high-novelty conditions from various novelty functions Pathak et al. [2017], Burda et al. [2019]. Although PGR successfully encourages exploration into higher-novelty regions, we observe that the generated samples are susceptible to the choice of hyperparameters. To this end, we propose a novel approach that defines a carefully designed sampling distribution for generative replay and then provides a recipe that accurately samples from the target distribution.

6 Conclusion

In this paper, we propose REGR, a novel generative replay framework that promotes robust exploration in online reinforcement learning. We analyze the limitations of prior generative replay approaches that rely on conditional generative models and propose a novelty-tilted sampling distribution that yields a better trade-off between exploration and dynamic plausibility. To sample from the tilted distribution, we fine-tune unconditional diffusion models via RTB [Venkatraman et al., 2024] objective. Our extensive experiments on DMC suite [Tassa et al., 2018] and OpenAI Gym [Brockman et al., 2016], along with an in-depth analysis, demonstrate the effectiveness and robustness of our method.

Limitation and future work. While our framework shows robust performance improvement across different RL methods and novelty functions, it requires training diffusion models from scratch multiple times. Furthermore, sampling a large batch of transitions remains computationally expensive. Although our analysis indicates that REGR incurs minimal additional wall-clock overhead, one may consider a lightweight framework that efficiently reuses trained models from previous rounds or accelerates sampling through one-step generative modeling [Geng et al., 2025, Deng et al., 2026].

References

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations (ICLR)*, 2019.

Xinyue Chen, Che Wang, Zijian Zhou, and Keith W Ross. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations (ICLR)*, 2021.

Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *International Conference on Learning Representations (ICLR)*, 2022.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770*, 2026.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Matteo Hessel, Ian Osband, Alex Graves, Volodymyr Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy networks for exploration. In *International Conference on Learning Representations (ICLR)*, 2018.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning (ICML)*, 2018.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning (ICML)*, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, 2018.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: a vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

Haque Ishfaq, Guangyuan Wang, Sami Nur Islam, and Doina Precup. Langevin soft actor-critic: Efficient exploration through uncertainty-driven critic learning. *International Conference on Learning Representations (ICLR)*, 2025.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Jaewoo Lee, Sujin Yun, Taeyoung Yun, and Jinkyoo Park. Gta: Generative trajectory augmentation with guidance for offline reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2016.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3):293–321, 1992.

Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2023a.

Cong Lu, Philip Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023b.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Mirco Mutti, Lorenzo Pratissoli, and Marcello Restelli. Task-agnostic exploration via policy gradient of a non-parametric state entropy estimate. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

Mirco Mutti, Riccardo De Santi, and Marcello Restelli. The importance of non-markovianity in maximum state entropy exploration. In *International Conference on Machine Learning (ICML)*, 2022.

Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):48, 2017.

Georg Ostrovski, Marc G Bellemare, Aäron Oord, and Rémi Munos. Count-based exploration with neural density models. In *International Conference on Machine Learning (ICML)*, 2017.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, 2017.

Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. *International Conference on Learning Representations (ICLR)*, 2018.

Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7):eaap7885, 2018.

Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. *International Conference on Learning Representations (ICLR)*, 2018.

Younggyo Seo, Lili Chen, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. State entropy maximization with random encoders for efficient exploration. In *International Conference on Machine Learning (ICML)*, 2021.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Harshinder Singh, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23 (3-4):301–321, 2003.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *International Conference on Learning Representations (ICLR)*, 2020.

Bhavya Sukhija, Lenart Treven, Cansu Sancaktar, Sebastian Blaes, Stelian Coros, and Andreas Krause. Optimistic active exploration of dynamical systems. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Bhavya Sukhija, Stelian Coros, Andreas Krause, Pieter Abbeel, and Carmelo Sferrazza. Maxinforl: Boosting exploration in reinforcement learning through information gain maximization. *International Conference on Learning Representations (ICLR)*, 2024.

Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

Siddarth Venkatraman, Moksh Jain, Luca Scimeca, Minsu Kim, Marcin Sendera, Mohsin Hasan, Luke Rowe, Sarthak Mittal, Pablo Lemos, Emmanuel Bengio, et al. Amortizing intractable inference in diffusion models for vision, language, and control. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Renhao Wang, Kevin Frans, Pieter Abbeel, Sergey Levine, and Alexei A Efros. Prioritized generative replay. *International Conference on Learning Representations (ICLR)*, 2024.

Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando De Freitas. Sample efficient actor-critic with experience replay. *International Conference on Learning Representations (ICLR)*, 2017.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2008.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning (CoRL)*, 2023.

A Architecture

Through all our experiments, we use the same architectural choices of models over across baselines [Lu et al., 2023b, Wang et al., 2024]. The details of our denoising, policy, and critic networks are represented in Table 5, 6, 7.

Table 5: Denoising Network

Hyperparameter	Value
Architecture	6-layer MLP
Hidden dimension	1024
Activation	Mish
Optimizer	AdamW
Learning rate (LR)	3×10^{-4}
LR scheduler	Cosine
Batch size	256
Diffusion steps T	1000
Noise schedule	Linear
Sampling steps	100

Table 6: Policy Network

Hyperparameter	Value
Architecture	3-layer MLP
Hidden dimension	256
Activation	ReLU
Optimizer	Adam
Learning rate (LR)	3×10^{-4}

Table 7: Critic Network

Hyperparameter	Value
Architecture	3-layer MLP
Hidden dimension	256
Activation	ReLU
Optimizer	Adam
Learning rate (LR)	3×10^{-4}
Target smoothing τ	0.005
Discount factor γ	0.99
Batch size	256

B Hyperparmeters and Implementation details

To reproduce the experimental results of the baselines in Table 1, we faithfully follow the instructions of Wang et al. [2024]. For environments where no hyperparameter instructions are provided, we conduct a grid search over the search space. If the hyperparameters reported in the original paper yield lower performance than other combinations covered in the grid search, we report the best-performing configuration in our experiments.

Table 8: Hyperparameter search space for the evaluated methods.

Algorithms	**Hyperparameter**	**Search Space**
PGR	y	$\{0.25, 0.5\}$
	ω	$\{1.0, 2.0, 5.0\}$
REGR	β	$\{0.1, 0.25, 0.5, 1.0\}$

Our generative model follows the framework of Ho et al. [2020], and the sampling procedure is accelerated using the approach of Song et al. [2020]. The detailed hyperparameter settings of the diffusion model are summarized in Table 5. The original SAC algorithm uses an update-to-data ratio of 1. In contrast, our baselines (SER and PGR) maintain a synthetic replay buffer composed of samples generated by a generative model, enabling substantially more gradient updates per environment interaction. Following the baseline setup, we use a ratio of 20 for both our method and the baselines. The same configuration is also applied when evaluating performance with TD3 to ensure consistent comparisons.

When the novelty function ν is implemented using learned models such as neural networks, a small number of samples can exhibit extremely large novelty values. To obtain a more reliable estimate of epistemic uncertainty in transitions, we clip novelty values at the 95th percentile.

During diffusion model fine-tuning, we use a learning rate of 1×10^{-4} with a batch size of 1024. We generate 1M synthetic samples, update the generative replay every 10,000 environment steps, and set the synthetic data ratio to 0.5.

C State Entropy

To verify that our method promotes exploration through the effective use of generative replay, we evaluate the state coverage of the agent during training. State-coverage serves as a proxy for the diversity of states visited by the policy and is quantified through the entropy of the state distribution.

Let $s \in \mathcal{S} \subset \mathbb{R}^d$ denote a state and let $p(s)$ denote the state distribution induced by the trajectories collected during training. The state entropy is defined as

$$H(p) = -\int_{\mathcal{S}} p(s) \log p(s)\, ds. \tag{10}$$

In practice, the true state distribution $p(s)$ is unknown and the expectation above cannot be computed directly. Instead, the entropy must be estimated from samples of visited states.

To estimate the entropy, we adopt the k-nearest neighbor entropy estimator, following Singh et al. [2003]. Let $\mathcal{D}_s = \{s_1, s_2, \ldots, s_n\}$ be a set of states sampled from the replay buffer. For each state s_i, let $R_{i,k}$ denote the Euclidean distance to its k-th nearest neighbor among the samples. The entropy can then be estimated as

$$\hat{H}_k = \frac{d}{n} \sum_{i=1}^{n} \log R_{i,k} + \log \Gamma\left(\frac{d}{2} + 1\right) + \gamma - L_{k-1} + \log n, \tag{11}$$

where d is the dimensionality of the state space, $\Gamma(\cdot)$ is the Gamma function, γ is the Euler–Mascheroni constant, and $L_{k-1} = \sum_{j=1}^{k-1} \frac{1}{j}$.

The entropy estimator is based on the distances to the k-nearest neighbors. Intuitively, these distances reflect the local density of samples in the state space. When the states in the dataset are concentrated in a small region, the k-nearest neighbor distances tend to be small. Conversely, when the states are more spread out, the neighbor distances tend to be larger. Since the entropy estimator depends on the logarithm of these distances, larger average neighbor distances generally lead to larger entropy estimates. Estimating or maximizing the entropy of the state distribution has been explored in prior work as a mechanism for encouraging exploration in reinforcement learning [Mutti et al., 2021, 2022, Seo et al., 2021]. In all experiments in this paper, we set $k = 12$.

D Q-Bias Estimation

The learned Q-function may systematically deviate from the true action-value function of the current policy. Q-bias is defined as the difference between the predicted Q-value and the ground-truth return. Let $Q_\eta(s, a)$ denote the learned Q-function and $Q^\pi(s, a)$ denote the true action-value function under the current policy π. The bias at a state–action pair is defined as $Q_\eta(s, a) - Q^\pi(s, a)$, where positive values indicate overestimation, and negative values indicate underestimation.

Since the ground-truth action-value function $Q^\pi(s, a)$ is generally unknown, following Chen et al. [2021], we estimate $Q^\pi(s, a)$ using Monte Carlo rollouts under the current policy. Specifically, for a given state–action pair (s, a) encountered during training, we execute the current policy to obtain the discounted return

$$G(s, a) = \mathbb{E}_{\pi|s_0=s,a_0=a}\left[\sum_{t=0}^{T} \gamma^t r_t\right] \tag{12}$$

which serves as an unbiased estimator of $Q^\pi(s, a)$ in expectation. The empirical bias is then approximated as $Q_\eta(s, a) - G(s, a)$.

To summarize the key statistics across visited state–action pairs, two metrics are commonly reported. The *average bias* measures the mean deviation of the learned Q-values from the ground-truth values, indicating whether the estimator systematically overestimates or underestimates the return. Because reward scales vary across environments, the bias is often normalized by the average magnitude of returns. In contrast, the *standard deviation of bias* measures the variability of the bias across different state–action pairs. A large standard deviation indicates that the bias is highly non-uniform across the state space, which can lead to unstable policy updates even when the bias is small. Together, these metrics provide a clear view of the stability of value function training.

E Time complexity

Table 9 reports the empirical time costs of each algorithm on the DMC `reacher-hard` environment, measured on a single RTX 3090 (24GB) GPU. The reported values correspond to the computation time of diffusion model operations executed every training interval I. Our method introduces an additional fine-tuning stage on top of the prior training and sampling procedures. The longer prior training time of PGR arises from learning conditional noise, which requires evaluating all samples in the replay buffer by using the novelty function network to construct conditioning labels.

Table 9: Time complexity comparison.

Algorithms	**Time (min)**			**Total time (min)**
	Training prior	Fine-tuning	Sampling	
SER	22.39	-	5.89	28.29
PGR	26.31	-	11.90	38.21
REGR (Ours)	22.35	1.10	11.68	35.13

F Additional Results

In this section, we present additional experimental results that are not included in the main manuscript. We plot learning curves over the course of training for different tasks in Figure 8. As shown in the figure, REGR consistently outperforms other generative replay baselines in terms of both sample efficiency and performance.

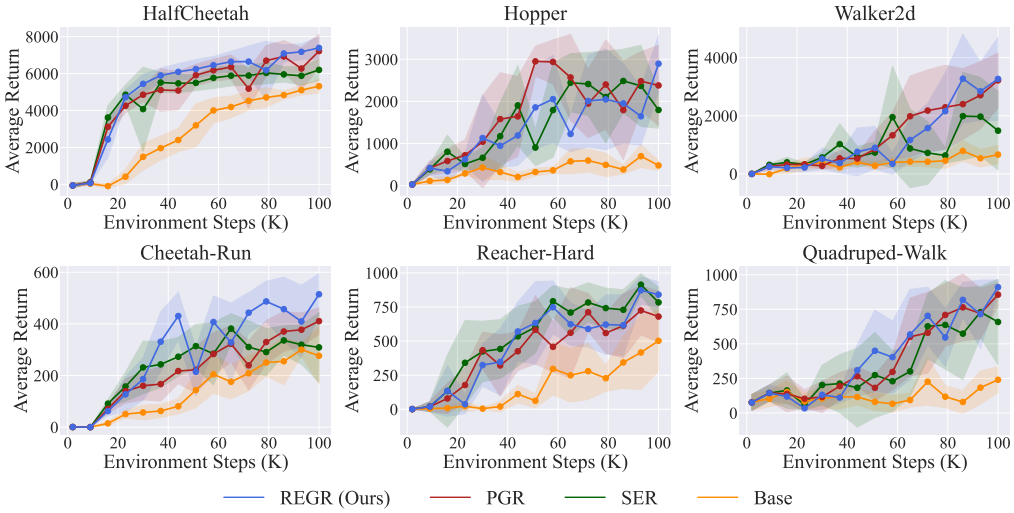

Figure 8: Extended main experiment results on continuous control tasks. For all experiments, we run five seeds and report the mean and one standard deviation.