Curious Replay for Model-based Adaptation

July 2, 2025

Reference

Isaac Kauvar, Chris Doyle, Linqi Zhou, and Nick Haber. Curious replay for model-based adaptation, 2023. URL https://arxiv.org/abs/2306.15934.

Curious Replay for Model-based Adaptation ([Kauvar et al., 2023])

Curious Replay Algorithm

```
1: Input: Replay buffer $R$ that uses a SumTree structure to store the priority $p_i$ of each transition
2: Hyperparameters: $c$, $\beta$, $\alpha$, $\epsilon$, environment steps per train step $L$, batch size $B$, maximum priority $p_{\text{MAX}}$
3: for iteration 1, 2, ... do
4:     Collect $L$ transitions $(x_t, a_t, r_t, x_{t+1})$ with policy
5:     Add transitions to replay buffer $R$, each with priority $p_i \leftarrow p_{\text{MAX}}$ and visit count $v_i \leftarrow 0$
6:     Sample batch of $B$ transitions from $R$ using probability for selecting transition $i$ as $p_i / \sum_{j=1}^{|R|} p_j$
7:     Train world model and policy using batch, and cache loss $\mathcal{L}_i$ for each transition in batch
8:     for transition $i$ in batch do
9:         $p_i \leftarrow c\beta^{v_i} + (|\mathcal{L}_i| + \epsilon)^\alpha$
10:        $v_i \leftarrow v_i + 1$
11:    end for
12: end for
```

Curious Replay: Motivation

- **Challenge:** After an environment change, the world model is suddenly inaccurate on new data.
- **Goal:** Adapt quickly by focusing learning on the most relevant transitions.
- **Solution: Curious Replay** prioritizes replay buffer sampling to:
  - Fix transitions the model predicts poorly *right now*
  - Ensure new, rarely trained transitions are not neglected

Curious Replay: The Two Key Signals

1. **Model Error:** Use the world model's prediction loss $\mathcal{L}_i$ for each transition i.
   - Higher $|\mathcal{L}_i|$ means the model is more wrong $\rightarrow$ sample it more.
2. **Under-Replay:** Track a visit count v_i for each transition.
   - Fewer replays (lower v_i) $\rightarrow$ boost its priority to ensure coverage, especially for new data.

Curious Replay: Priority Formula

Priority for transition *i*:

$$p_i \leftarrow c\,\beta^{v_i} + (|\mathcal{L}_i| + \epsilon)^{\alpha}$$

- $c > 0$: base curiosity bonus for rarely trained items
- $0 < \beta < 1$: decays the curiosity bonus as *i* is replayed more
- $\alpha \in (0, 1]$: softens extremes (like in PER)
- $\epsilon > 0$: avoids zero priority

Sampling probability:

$$P(i) = \frac{p_i}{\sum_j p_j}$$

After sampling *i*-th transition:

$$v_i \leftarrow v_i + 1$$

Curious Replay: Why It Works

- **After a change:**
  - New/shifted transitions have high $|\mathcal{L}_i|$ and low $v_i \rightarrow$ get sampled more.
- **As learning progresses:**
  - Both error and curiosity bonus decrease, so focus shifts to other transitions.
- **Result:** The world model adapts quickly, and the policy can trust its imagined rollouts again.

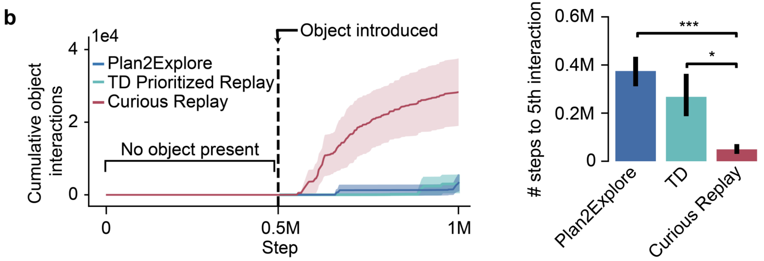
Curious Replay vs. Classic PER

- **PER:** Prioritizes by value TD error (how surprising the reward is).
- **Curious Replay:** Prioritizes by world-model error (how surprising the transition is) **and** a freshness bonus.
- **Why?** In Dreamer, model accuracy is the bottleneck for adaptation after distribution shifts.

Curious Replay: Practical Tips

- Start with $\alpha \in [0.5, 1]$, $\beta \in [0.8, 0.99]$, small ϵ
- Choose c so new items get sampled promptly but don't dominate
- Cap priorities at p_{MAX} to avoid outliers
- (Optional) Add importance-sampling weights for strict bias control, but the simple scheme works well in practice

Experiment: Interaction with Object Introduced Halfway



Experiment: Constrained Control Released Halfway

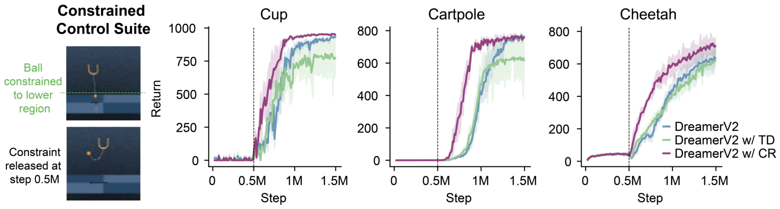
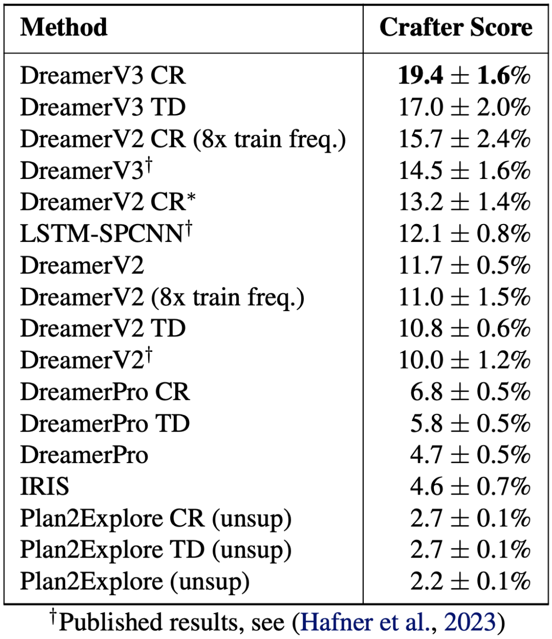


Figure 3. DreamerV2 w/ Curious Replay outperforms DreamerV2 and DreamerV2 w/ TD in the Constrained Control Suite (n=6 per method, mean +/- s.e.m.)

Experiment: CR Helps for Crafter



Method	Crafter Score
DreamerV3 CR	**19.4 ± 1.6%**
DreamerV3 TD	17.0 ± 2.0%
DreamerV2 CR (8x train freq.)	15.7 ± 2.4%
DreamerV3[†]	14.5 ± 1.6%
DreamerV2 CR*	13.2 ± 1.4%
LSTM-SPCNN[†]	12.1 ± 0.8%
DreamerV2	11.7 ± 0.5%
DreamerV2 (8x train freq.)	11.0 ± 1.5%
DreamerV2 TD	10.8 ± 0.6%
DreamerV2[†]	10.0 ± 1.2%
DreamerPro CR	6.8 ± 0.5%
DreamerPro TD	5.8 ± 0.5%
DreamerPro	4.7 ± 0.5%
IRIS	4.6 ± 0.7%
Plan2Explore CR (unsup)	2.7 ± 0.1%
Plan2Explore TD (unsup)	2.7 ± 0.1%
Plan2Explore (unsup)	2.2 ± 0.1%

[†]Published results, see (Hafner et al., 2023)

Experiment: Background Swap in DMC

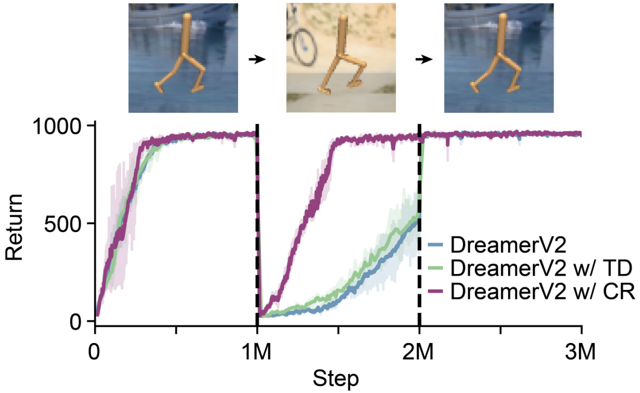


Figure 5. Background-Swap walker_walk. Background changes at step 1M, and reverts at step 2M. Curious Replay improves performance after 1M, without degrading performance after 2M.