# Curious Replay for Model-based Adaptation

**Isaac Kauvar** [*1] **Chris Doyle** [*1] **Linqi Zhou** [2] **Nick Haber** [1]

## Abstract

Agents must be able to adapt quickly as an environment changes. We find that existing model-based reinforcement learning agents are unable to do this well, in part because of how they use past experiences to train their world model. Here, we present Curious Replay—a form of prioritized experience replay tailored to model-based agents through use of a curiosity-based priority signal. Agents using Curious Replay exhibit improved performance in an exploration paradigm inspired by animal behavior and on the Crafter benchmark. DreamerV3 with Curious Replay surpasses state-of-the-art performance on Crafter, achieving a mean score of 19.4 that substantially improves on the previous high score of 14.5 by DreamerV3 with uniform replay, while also maintaining similar performance on the Deepmind Control Suite. Code for Curious Replay is available at github.com/AutonomousAgentsLab/curiousreplay.

## 1. Introduction

Change is unavoidable. Robust artificially intelligent (AI) agents must be capable of quickly adapting to changing circumstances. In the face of novel states and shifting conditions, agents—whether self-driving cars, home-assistant robots, or financial decision makers—must effectively update their understanding of the world and their policies for acting in it. Animals have evolved to skillfully contend with the challenges of changing environments. Can they serve as an existence proof and source of inspiration for synthesizing such flexible intelligence?

Consider, for instance, a simple change in one's environment: the appearance of a new object. Animals, from rodents to primates, will often adapt to such a change by first investigating the object (Glickman & Sroges, 1966). This is an effective strategy as it addresses a key aspect of adaptation: assembling data to update models that guide behavior. In this work, we investigate such functionalities of gathering and utilizing information about environmental change in the context of model-based deep reinforcement learning.

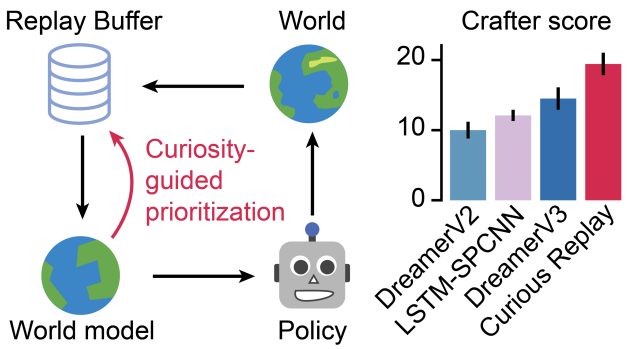


*Figure 1.* Curious Replay closes the loop between experience replay and world model performance by using curiosity-guided prioritization to promote training on experiences the model is least familiar with. Curious Replay improves the adaptability of model-based agents and yields a new state-of-the-art score on Crafter.

We identify surprising deficiencies in the adaptability of a state-of-the-art model-based agent (Dreamer, with Plan2Explore for intrinsically-motivated settings (Hafner et al., 2020; Sekar et al., 2020; Hafner et al., 2023)). In an object interaction assay, we find that—unlike animals—Plan2Explore agents do *not* quickly interact with a novel object. Moreover, in a nonstationary variant of the Deepmind Control Suite, DreamerV2 does not adapt well. These results reveal shortcomings in the Dreamer agent that may broadly impact its ability to achieve its full potential in challenging settings.

We find a root cause of Dreamer's poor adaptability: reliance on uniform sampling of its experience replay buffer. We address this problem with Curious Replay—a new approach to prioritizing replay buffer sampling when updating an agent's models—and demonstrate its profound benefits for adaptation of model-based agents, including achievement of a new state-of-the-art on Crafter (Figure 1).

[*]Equal contribution [1]Graduate School of Education, Stanford University, Stanford, CA, USA [2]Department of Computer Science, Stanford University, Stanford, CA, USA. Correspondence to: Isaac Kauvar <ikauvar@stanford.edu>, Nick Haber <nhaber@stanford.edu>.

*Proceedings of the $40^{th}$ International Conference on Machine Learning*, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

Curious Replay utilizes curiosity as an intrinsic signal — not for action selection, but to choose experiences for model updates. This greatly boosts performance in changing environments while maintaining similar performance in unchanging ones. Moreover, Curious Replay also offers benefits in environments where the need for adaptation is less obvious, like the open-world game Crafter (Hafner, 2021), indicating its potential value for broadly improving model-based agents.

The success of Curious Replay stems from the idea that for a model-based system to adapt effectively, the model must keep pace with changes in the environment. This is crucial as an inaccurate model can hinder appropriate action selection, especially for actions related to new environmental changes, creating a compounding issue where poor actions lead to poor data collection. To overcome this, our approach is to emphasize training the model on unfamiliar or challenging aspects of the environment. This differs from uniform replay buffer sampling, which can lead to the model spending too much time training on old or irrelevant experiences, and neglecting important updates to its world model.

Curious Replay expands on the established success of prioritized experience replay (Schaul et al., 2015) by tailoring it for the model-based setting with curiosity, yielding large improvements in agent adaptability.

In sum, we make the following key contributions:

- We describe Curious Replay (CR), a method that aids model-based RL agent adaptation by prioritizing replay of experiences the agent knows the least about.
- We introduce assays for studying RL in changing environments, including an object interaction task inspired by animal behavior, and we find that DreamerV2 agents fail to quickly adapt to the changing environments.
- We show that CR improves performance in these adaptation assays, with more than 6× improvement at object interaction.
- We show that combining CR with DreamerV3 yields a new state-of-the-art score on Crafter, improving upon DreamerV3 by a factor of 1.33, and while maintaining a similar overall score on Deepmind Control Suite.

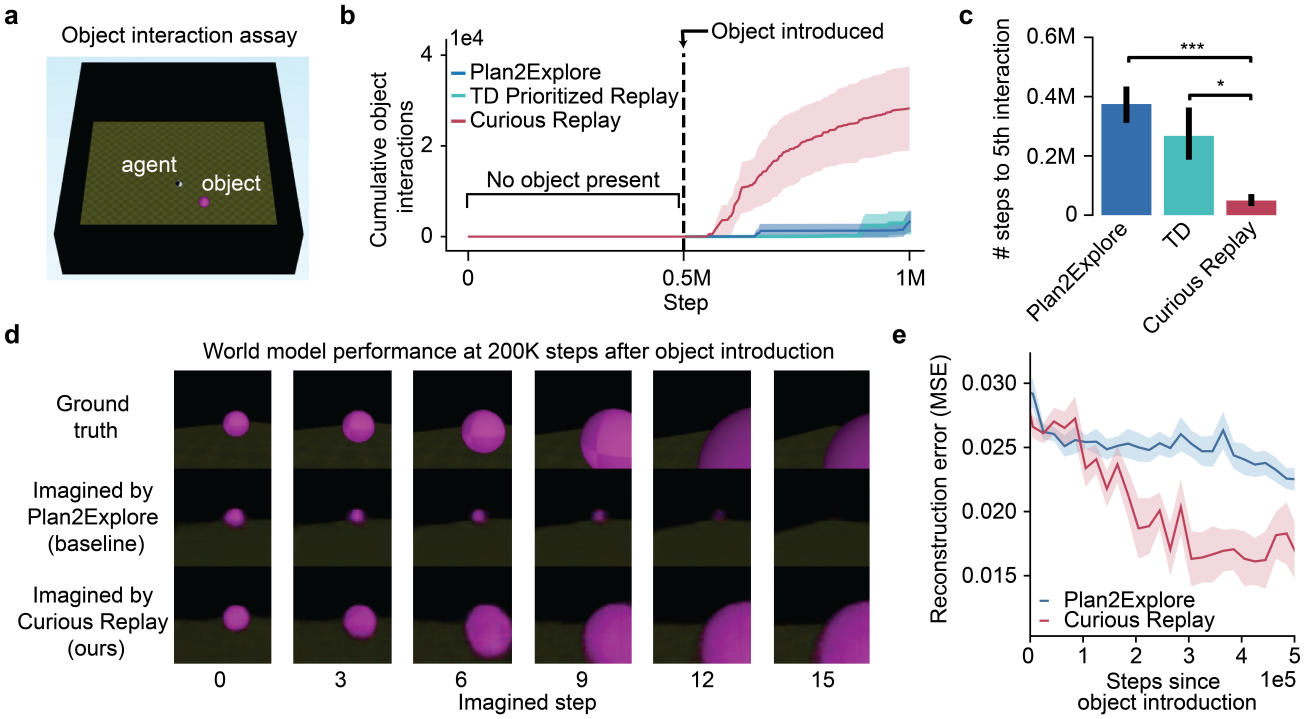


*Figure 2.* Inspired by animal behavior, we investigate adaptation using an object interaction assay. a) In this 3D physically-simulated assay, an intrinsically-motivated agent can explore an empty arena for 500K steps, at which point a novel object is introduced. The expectation is that agents, like animals, will quickly begin to interact with the novel object. b) The baseline Plan2Explore agent does not quickly interact with the object. Curious Replay dramatically improves the adaptability of Plan2Explore. c) Curious Replay yields significantly faster time-to-interaction than both Plan2Explore and Plan2Explore with prioritized experience replay using value-based temporal-difference error (TD) prioritization (Schaul et al., 2015) (n=7 each condition, independent t-test with fdr-bh correction). d) Plan2Explore fails to quickly model the novel object (visualized by an example egocentric imagined rollout), even though it has substantial experience observing the object (see Figure A5). In contrast, the Curious Replay agent quickly learns to accurately model the object. Each displayed rollout is from the lowest-error model (at 200K steps after object introduction) of 7 random seeds. e) Summary of model performance over time, demonstrating the faster rate at which Curious Replay learns to accurately model the object (n=7 each, mean +/- sem).

## 2. Changing environments

### 2.1. RL in changing environments

In reinforcement learning (RL), agents experience an environment as a sequence of states and choose actions for each state to maximize expected reward. The problem can be formulated as a partially observed Markov decision process (POMDP) parameterized by the tuple $(S, A, T, R, \Omega, O, \gamma)$. Here, $S$ is the set of states, which are not directly accessible to the agent, $A$ is the set of actions, $T$ is the action-dependent transition probabilities between states, $R$ is the reward function, $\Omega$ is the set of observations, $O$ is the function that transforms the state to an observation $x \in \Omega$, and $\gamma \in (0, 1)$ is the discount on future rewards. We consider image observations $x \in \mathbb{R}^{M \times N \times C}$, with dimensions $M$ and $N$ and color channels $C$, and both discrete- and continuous-action space environments. Rewards $r \in \mathbb{R}$ may depend on components of the state related to the agent (intrinsic rewards) or the environment (extrinsic rewards). The objective is to learn a policy $\pi$ that maximizes $\mathbb{E}_\pi[\sum_{t \geq 0} \gamma^t r_t]$.

We focus on changing environments, in which there can be a step change in the dynamics or states. Changing environments are a special case of the POMDP where there is a natural representation that separates a discrete latent variable $\chi$ from $S$ to represent phases of the environment. In this framing, $T$, $R$, $\Omega$ and $O$ all become functions of the unobserved variable $\chi$, and $T$ defines both the next state and the next phase of the environment. This framing is natural for certain environments characterized by important step changes. These can occur as a result of factors external to the agent (e.g. at time $t = T_0$, a light in the environment is programmed to turn on), or because of achievements by the agent (e.g. the agent enters a new area). A consequence of such a change is that aspects of the agent's model are no longer accurate or comprehensive. Observations may now be novel, or actions may now have a different effect, and the agent must adapt.

Such changing environments differ from benchmarks such as Deepmind Control Suite (Tassa et al., 2018), where the environment stays consistent throughout learning. They also differ from the task of training across multiple environments followed by evaluation on a test environment, with the agent explicitly cued that it is in the test phase (Parisi et al., 2021). Moreover, we do not focus on leveraging a phase of unsupervised exploration to inform downstream task-completion in a single unchanged environment (Sekar et al., 2020; Laskin et al., 2021). Nor do we focus on the task of robustification from distracting stimuli, for which data augmentation has been successfully applied (Deng et al., 2022; Yarats et al., 2020). Rather, we are interested in scenarios where some aspect of the environment changes—whether the dynamics, the observations, or the possible results of actions—and where no explicit cue tells the agent when a change happens.

We investigate adaptation of model-based agents in three settings: an intrinsically-motivated object interaction assay, variants of the Deepmind Control Suite, and Crafter (Hafner, 2021). While reminiscent of existing settings with a reward-free exploration phase followed by a task adaptation phase (Parisi et al., 2021; Laskin et al., 2021; Sekar et al., 2020), our assays are distinct in testing how agents respond to changes in environment states or dynamics, with the agent not explicitly cued when entering a new phase.

### 2.2. Object interaction assay

The object interaction assay is inspired by animal behavior. In response to the appearance of a novel object, animals will quickly begin to investigate the object (Glickman & Sroges, 1966; Ahmadlou et al., 2021). Notably, animals interact with the object more quickly if it represents a change to the environment (i.e. if it appears after the animal has had time to explore the object-less environment). We verified these phenomena with animal experiments in Figure A1.

We sought to assess the investigatory behavior of AI agents in an analogous setting. We implemented the object interaction assay in a 3D egocentric, image-based environment that extends the dm_control simulation framework (Tunyasuvunakool et al., 2020). In the assay, an agent explores an empty, square arena with black walls. At $T_0 = 500$K steps, a magenta ball appears near the center of the arena. The ball is stationary but untethered, and the agent can collide with it to move it around the arena. The agent has not previously experienced magenta. We also implemented an unchanging version of the assay, with the ball present from the start.

There is no extrinsic reward in this assay, and agents must be guided by an intrinsic reward signal. Because interacting with the ball yields very different observations and dynamics than any other part of the environment, we expect that good exploratory behavior should involve interaction with the ball. We thus quantify exploration by the number of agent-ball interactions (e.g. collisions). We also quantify the accuracy of the learned world model on test episodes, as in Figure 2.

### 2.3. Deepmind Control Suite

To investigate agent adaptation in an extrinsically-rewarded, changing environment, we modify the Deepmind Control Suite (Tassa et al., 2018) to make it a changing environment. In the Constrained Control Suite variant, we modify the cheetah, cup, and cartpole environments to add constraints on the motion of the agent's appendages (Figure A12). These constraints are released at $T_0 = 500$K steps, necessitating adaptation to the newly available action outcomes. The agent can only achieve maximum extrinsic reward in the unconstrained phase. To adapt effectively, it must respond to the changed constraints and actively investigate the newly available states. In the Background-Swap

**Algorithm 1** Curious Replay

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

Control Suite, we leverage the Distracting Control Suite framework (Stone et al., 2021), using one static image as a background until $T_0 = 1$M, at which point the background changes to a new image. At $T_1 = 2$M, the background reverts to the original image. We can thus test adaptation at $T_0$, and maintenance of performance from the first environment phase at $T_1$. We additionally assess model performance on the original, unchanging, Deepmind Control Suite.

### 2.4. Crafter

We also sought to test adaptation in a more complex and well-validated setting. For this, we turned to Crafter (Hafner, 2021), a procedurally generated, open-world survival game in which an agent pursues a variety of hierarchical achievements. Uncovering an achievement changes the states and achievements available to an agent, and agents must leverage this new knowledge and adapt subsequent behavior. For example, whereas the Collect Drink achievement has no prerequisities, Collect Coal is only unlocked by preceding achievements of Collect Wood, Place Table, and Make Wood Pickaxe. In this manner, the available states change as the agent progresses. The score reflects achievement success across trials. Humans score around 50%.

## 3. Curious Replay

### 3.1. Model-based RL in changing environments

Model-based RL leverages a state-predictive world model to inform learned action selection (Moerland et al., 2023). At least two key advantages are promised relative to model-free RL: more data-efficient training due to the compression of experience into a predictive world model, and more effective policies that can leverage the world model for planning. Recent success with model-based agents has yielded key gains relative to model-free systems, including in data-efficiency and overall task performance (Schrittwieser et al., 2020; Hafner et al., 2020; 2023), and we restrict our investigation here to improving model-based systems.

Dreamer is a particularly successful architecture for model-based RL from images (Hafner et al., 2019a; 2020; 2023; Wu et al., 2022), with three key components: (1) a replay buffer of stored experience (2) a world model that embeds observations to a latent state and uses a forward-dynamics Recurrent State Space Model (Hafner et al., 2019b) to imagine action-conditioned future states, and (3) an actor-critic policy trained on trajectories imagined by the world model. Dreamer can be effectively augmented with intrinsic reward, such as disagreement in Plan2Explore (Sekar et al., 2020).

Training Dreamer consists of three main steps that are cycled until convergence: (1) environment interaction, (2) world model learning, and (3) policy learning. Interaction with the environment is recorded into an experience replay buffer as sequences of observations. The world model is optimized to fit the data in the replay buffer, using uniformly sampled sequences. The actor-critic policy is then trained on imagined trajectories that are simulated by the world model, seeded at initial states recalled from the replay buffer.

Dreamer's world model allows the policy to learn using multi-step predictions in a compact latent space, yielding more sample-efficient training of sophisticated policies. The key drawback, however, is that if the world model is too inaccurate, the policy will be ineffective. This can be particularly problematic in changing environments if the model does not keep up-to-date with observed changes.

Initial object interaction experiments demonstrate that Dreamer suffers from this problem in changing environments. As shown in Figure 2, the baseline Plan2Explore agent struggles to represent a novel object that appears in the playground environment, even though it has observed the object for tens of thousands of steps (Figure A5). Moreover, there is a counterintuitive deficiency in the agent's behavior: the agent is ten times slower to interact with the object in the changing version of the assay than in the unchanging version (Figure A2). This ordering is the opposite of animal behavior, and represents a serious gap in performance.

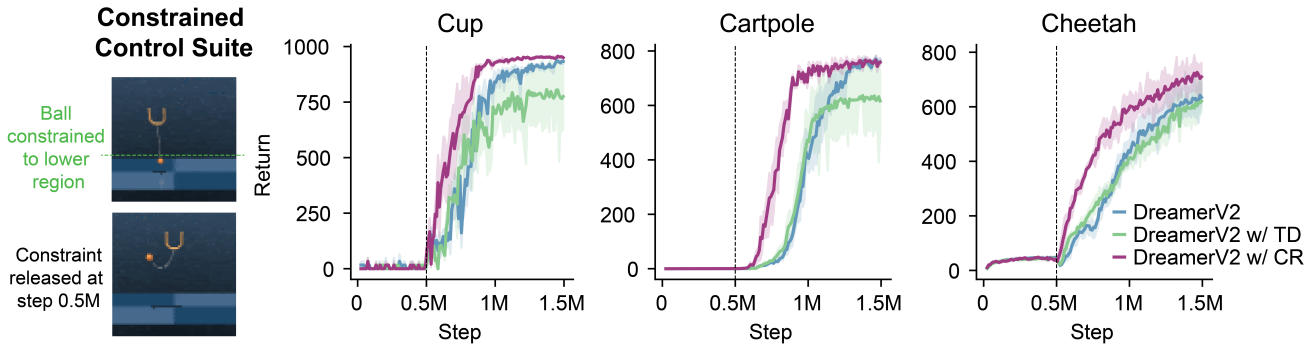


*Figure 3.* DreamerV2 w/ Curious Replay outperforms DreamerV2 and DreamerV2 w/ TD in the Constrained Control Suite (n=6 per method, mean +/- s.e.m.)

### 3.2. Combining curiosity and prioritized replay

Curious Replay addresses the challenge of model-based adaptation in changing environments. It encourages an accurate, adaptive world model by prioritizing optimization on experiences that have been trained on the fewest times or that are least accurately modeled. We take inspiration from two sources: (1) Prioritized Experience Replay, which uses temporal-difference (TD) error of the value estimate for prioritization (Schaul et al., 2015); and (2) Curiosity as an intrinsic motivation, which has received much attention as a signal for guiding exploration in sparsely-rewarded environments (Schmidhuber, 1991; Oudeyer et al., 2007). Such curiosity signals include count-based novelty (Bellemare et al., 2016; Tang et al., 2017) or adversarial model-error (Stadie et al., 2015; Pathak et al., 2017; Haber et al., 2018; Guo et al., 2022). In the following, we describe versions of these two signals used to guide replay. By combining them into our method, Curious Replay, we improve Dreamer's ability to adapt to changing environments—as demonstrated in Figure 2, for example, where it dramatically enhances adaptation speed and world model accuracy.

### 3.3. Count-based Replay

One hypothesis is that Dreamer's uniform replay buffer sampling does not promote training the world model on new data from the changed environment. Inspired by count-based novelty, we develop Count-based Replay to ensure that the model trains on new data, and thus avoids ignoring potentially valuable data it has collected.

Count-based replay biases sampling towards recent experiences, and ensures that the agent revisits each experience multiple times (with high probability). Prioritization relies on tracking the visit count $v_i$, the number of times an experience has been revisited (i.e. incorporated in a training batch). Here, an experience is a single state transition. The counter $v \in \mathbb{R}^{|R|}$, for buffer capacity $|R|$, is used with hyperparameter $\beta \in [0, 1]$ to prioritize sampling as $p_i = \beta^{v_i}$. The priorities are stored in a SumTree (Schaul et al., 2015) to enable efficient normalization to a sampling probability.

### 3.4. Adversarial Replay

Count-based replay is agnostic to the actual content of the experiences. This can be a drawback if particular experiences are challenging to learn, or where certain experiences are redundant with those that have already been learned. We hypothesized that adaptation might be aided by prioritizing experiences that the model is not currently good at. This led to Adversarial Replay, which is inspired by adversarial intrinsic motivation (Stadie et al., 2015; Pathak et al., 2017), and can be similar to the notion of surprise (Berseth et al., 2019). In fact, there is evidence of a model-error related signal being helpful for prioritizing replay (Oh et al., 2021), and we tailor this approach to the Dreamer setting.

Adversarial replay prioritizes experiences that the world model does not accurately predict. It uses model-loss as a prioritization signal, with $p = (|\mathcal{L}| + \epsilon)^\alpha$ for loss $\mathcal{L}$, $\epsilon$ a small positive constant, and $\alpha \in [0, 1]$ that determines the extent of prioritization ($\alpha = 0$ is uniform sampling). $\mathcal{L}$ is the loss used to train the world model, which in the case of Dreamer is $\mathcal{L} = \mathcal{L}_{image} + \mathcal{L}_{reward} + \mathcal{L}_{KL}$. For simplicity and speed, the priority for an experience is only updated when it has been trained on. This has the potential to yield stale priorities, but we did not observe any associated deficiencies in our experiments. Priorities are initialized to the maximum value for new experiences added to the replay buffer. One potential consideration is the susceptibility of adversarial curiosity to the white-noise problem — where completely unpredictable state transitions can prevent the curious agent from being interested in anything else. However, Adversarial Replay is a fundamentally different scenario. Here, the curiosity signal provides prioritization for a fully supervised optimization problem—to fit the world model to the data in the replay buffer—and the experienced state transitions cannot be unpredictable in the same way.

### 3.5. Curious Replay (Adversarial + Count)

To combine benefits of count-based and adversarial replay—prioritizing experiences that are less-frequently replayed or less-understood—we additively merge both approaches into a single Curious Replay prioritization, using a scale factor $c$ to yield an overall priority for experience $i$ of:

$$p_i = c\beta^{v_i} + (|\mathcal{L}_i| + \epsilon)^\alpha \tag{1}$$

where $\beta$ determines the falloff of count-based priority, $v$ counts how many times each experience has been replayed, $\mathcal{L}_i$ is the loss for an experience, $\epsilon$ is a small positive constant, and $\alpha$ determines the sharpness of adversarial prioritization.

### 3.6. Temporal-Difference Prioritized Experience Replay

It is well known that prioritizing sampling based on value can be beneficial (Moore & Atkeson, 1993). Indeed, standard uses of prioritization leverage the temporal-difference error of the value estimate (Schaul et al., 2015; Horgan et al., 2018; Hessel et al., 2018): $\delta_t = r_t + \gamma V(x_{t+1}) - V(x_t)$, for value function $V(x)$ (e.g. critic), and $p_i = (|\delta_i| + \epsilon)^\alpha$. We also investigate use of this prioritization in our setting.

## 4. Experiments

In this section, we investigate the following questions:

Q1. Does Curious Replay consistently outperform other methods in environments with a step change at a single timepoint?
Q2. Does Curious Replay consistently outperform other methods in a more complex, open-world environment requiring continual learning, such as Crafter?
Q3. How much does each component of Curious Replay contribute?
Q4. How much impact does Curious Replay have on performance in unchanging environments?
Q5. Does a Curious Replay agent retain the ability to choose good actions and make good world predictions for earlier states that it has not recently experienced?

### 4.1. Experimental Setup

**Environment Details** Visual observations are $64 \times 64 \times 3$ pixel renderings. The Control Suite performance metric is extrinsic reward return. For object interaction, the primary metric is number of environment steps until the 5th object interaction (a metric determined in part by mouse experiments in Figure A1, but results are robust to the exact choice of interaction number, see Figure A10). For Crafter, the score is the geometric mean of achievement success rates, to account for the difficulty of different possible achievements.

**Baselines** Plan2Explore is our baseline for object interaction (DreamerV2 with latent disagreement as intrinsic motivation (Sekar et al., 2020)). We did not find other suitable baseline agents that were model-based, implemented with intrinsic-motivation, and with publicly-available code (e.g. AdA does not have a public implementation (Team et al., 2023)). DreamerV2 (Hafner et al., 2020) is our baseline for Constrained Control Suite investigations. For Crafter experiments, we include a number of baseline comparisons, including DreamerV2, DreamerPro (Deng et al., 2022), IRIS (Micheli et al., 2022), and DreamerV3 ((Hafner et al., 2023), for which code was not publicly available until after initial submission). In these settings, we also compare against augmenting the baseline agent with temporal-difference (TD) prioritized replay. We additionally compare against DreamerV3 and DreamerV2 on the full Deepmind Control Suite.

**Hyperparameters** Agents used defaults for DreamerV2, DreamerV3, Plan2Explore, and DreamerPro. For Curious Replay, $\beta = 0.7$, $\alpha = 0.7$, $c = 1e4$, $\epsilon = 0.01$, and $p_{MAX} = 1e5$. These were optimized on the object interaction assay and fixed across all environments.

### 4.2. Object interaction in a changing environment

To answer Q1, we use the object interaction assay. Curious Replay outperforms Plan2Explore and Plan2Explore w/ TD, interacting within 50K steps, which is more than six times faster than Plan2Explore (Figure 2). TD slightly improves Plan2Explore, but also has high variance. The Curious Replay agent quickly interacts many times with the object (at a faster pace than the other methods), and then starts to level off, yielding a total count by 900k steps that is similar to that of the other two agents (Figure A6). Figure 2 shows the difference in world model performance. Example agent trajectories demonstrate a clear difference in behavior with Curious Replay (Figure A7). Curious Replay also outperforms TD+Adversarial (Figure A8). We also assess Curious Replay in an unchanging environment, where the object is present from the outset. Baseline Plan2Explore performs much better in this setting, but performs no better than Curious Replay (Figure A9). The conclusions remain the same if different metrics of time-to-object interaction are used (Figure A10). Curious Replay is relatively insensitive to its hyperparameters; all tested values yielded improved object interaction relative to Plan2Explore (Table A1).

### 4.3. Constrained Control Suite

We further answer Q1 with the Constrained Control Suite, and find that on all three tasks, Cup Catch, Cartpole Swingup Sparse, and Cheetah Run, DreamerV2 w/ Curious Replay adapts faster than DreamerV2, (Figure 3). Quantified at step 750K, Curious Replay has an average return across tasks of 508 versus 121 for DreamerV2.

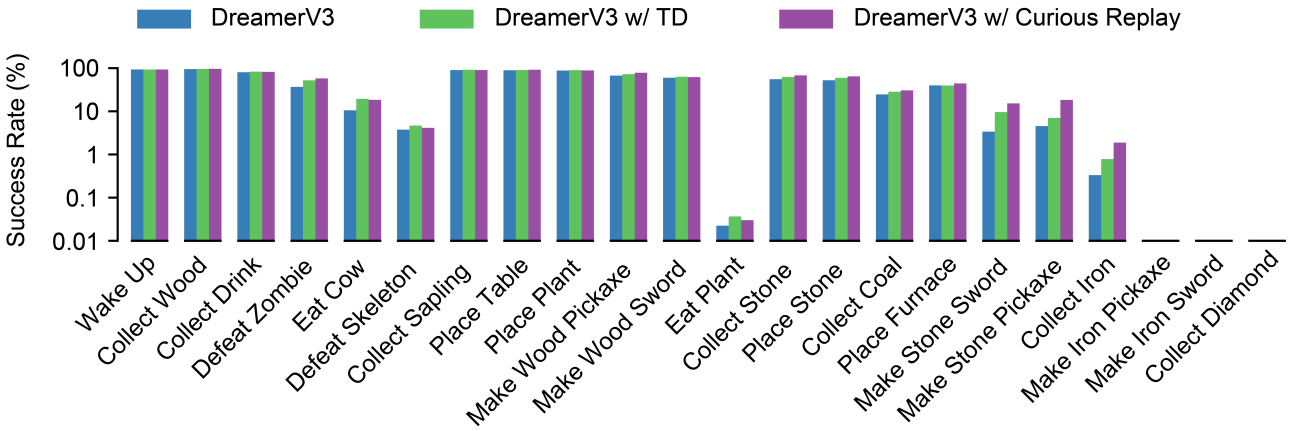


*Figure 4.* Agent ability spectrum for Crafter, ordered left to right by number of prerequisites for an achievement. (n=10 each).

### 4.4. Crafter

Answering Q2 in Table 1, Curious Replay substantially improves performance on Crafter. DreamerV3 w/ Curious Replay achieves a new state-of-the-art score on Crafter. Moreover, for each baseline (DreamerV3, DreamerV2, DreamerPro, and Plan2Explore), Curious Replay has better performance than the baseline as well as the baseline w/ TD prioritization. This high score is reflected in the agent ability spectrum (Figure 4), with Curious Replay exhibiting higher success at the more challenging achievements such as Collect Iron and Make Stone Pickaxe. Impressively, there were even a few rare instances when Curious Replay succeeded at an achievement never reached by DreamerV3, with 2/10 seeds of DreamerV3 w/ Curious Replay succeeding at Make Iron Sword (and 1/10 seeds of DreamerV3 w/ TD succeeding) . Curious Replay also improves the ability spectrum of Plan2Explore (Figures A13, A14) and DreamerV2 (Figure A15), especially for more challenging achievements.

We assessed the hyperparameter sensitivity of Curious Replay's performance on Crafter, using DreamerV2 (Table A2). We found that while all tested CR hyperparameters (including our default hyperparameters $\alpha$=0.7, $\beta$=0.7) yielded a higher mean score than DreamerV2, it was also possible to obtain an even higher score through hyperparameter search.

We also tested whether increasing the train ratio could improve the performance of Curious Replay (D'Oro et al., 2022). Highlighted in Table A3, we found that increasing the train frequency by 8x improves the Crafter score of DreamerV2 w/ Curious Replay (to 15.7 $\pm$ 2.4) but not DreamerV2.

**Why does CR help on crafter?** The set of resources held by the Crafter agent can be considered to be a new phase $\chi$ of the POMDP (see Section 2.1), and discovering a new resource means there is a new phase to learn about. We find that Curious Replay prioritizes and trains more frequently on experiences where the agent is holding recently discovered resources. To show this, we focused on the chain of achievements that includes "collecting iron," which the Curious Replay agent achieves at a higher rate than the baseline. To collect iron, a number of intermediate resources and tools must first be acquired sequentially: wood, a wood pickaxe, stone, a stone pickaxe, and then iron.

Experiences where the agent holds each of these resources beyond wood have a relative sampling probability $> 1$, meaning they are more likely to be sampled than with a uniform distribution (Table A4). Additionally, the experi-

*Table 1.* Crafter scores compared to previous algorithms. DreamerV3 with Curious Replay surpasses DreamerV3 to achieve a new state-of-the-art score. (CR = Curious Replay, TD = temporal difference prioritization, $n$=10 seeds per method, mean $\pm$ s.d.). $^*$=with optimized CR hyperparameters, see Table A2.

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

$^\dagger$Published results, see (Hafner et al., 2023)

ences of holding recently discovered resources are replayed more often during training (Table A5). The effect appears magnified for resources further in the chain, where there is a larger increase in relative training frequency.

The behavioral impact of Curious Replay is faster progression through the sequence of achievements. We analyzed the steps required for the average agent to reach 1% success on an achievement, in a 20K step moving average. Curious Replay leads to a substantially faster progression through the sequence of achievements, ultimately helping the agent succeed at more challenging achievements (Table A6).

### 4.5. Ablations

Answering Q3, we assessed individual contributions of the Adversarial and Count-based components of Curious Replay (Table 2). Each improved performance relative to the baseline, but the combination as Curious Replay was better.

*Table 2.* Performance of ablated versions of Curious Replay (ours), which combines Adversarial and Count Replay. Plan2Explore is used for object interaction, measured by # of steps to 5th interaction. DreamerV3 is used for Crafter, measured by the score.

| Assay | Adversarial | Count | CR (ours) |
|---|---|---|---|
| Object ↓ | $1.8 \pm 0.5$ | $2.2 \pm 0.7$ | $\mathbf{0.5 \pm 0.1}$ |
| Crafter ↑ | $17.3 \pm 2.7$ | $16.2 \pm 1.3$ | $\mathbf{19.4 \pm 1.6}$ |

### 4.6. Deepmind Control Suite

Answering Q4, we assessed Curious Replay's impact in unchanging environments by using the full Deepmind Control Suite (20 tasks). DreamerV3 w/ CR had a similar mean and median score across all 20 tasks as compared with DreamerV3 (Table 3). DreamerV2 w/ CR was slightly worse than DreamerV2. Interestingly, CR substantially improved performance on tasks such as Quadruped Run whereas it reduced performance on tasks such as Cartpole Swingup Sparse and Pendulum Swingup (Table A8, Figures A18, A19). There is thus perhaps a pattern, where CR yields improvement on complex locomotion but has a failure mode with sparsely rewarded swingup agents, suggesting an interesting direction for future investigation. Overall, and especially with DreamerV3, Curious Replay has little or no impact on performance across the full unchanging Deepmind Control Suite.

We additionally assessed whether Curious Replay is impacted by unpredictable environment noise. We used the Distracting Control Suite Walker Walk task with a noisy background, where at each time step the background is a randomly selected image. This experiment tests whether Curious Replay is inordinately hampered by complex and unpredictable distractions. We found that Curious Replay performs just as well as the baseline (Figure A17).

*Table 3.* Standard Deepmind Control Suite, mean and median return across 20 tasks, n=3 for DreamerV3 CR, n=2 for DreamerV2 CR. †Published results, see (Hafner et al., 2023).

| Method | Mean | Median |
|---|---|---|
| DreamerV3 CR | 734.8 | **815.1** |
| DreamerV3† | **739.6** | 808.5 |
| DreamerV2 CR | 643.6 | 635.8 |
| DreamerV2 | 715.7 | 770.7 |

### 4.7. Forgetting

We address Q5 with two approaches. First, we assess world model performance in a variant of the object interaction assay, where we introduce the object at $T_0 = 500$K steps and then remove the object at $T_1 = 1.5$M steps. Using the same approach as in Figure 2, we assess model performance on a set of test episodes. An increase in model error after object removal would signify evidence that the model is forgetting about the object. We find no such evidence for Plan2Explore or Plan2Explore w/ CR (Figure A4). To calibrate what catastrophic forgetting might look like, we assessed the performance of Plan2Explore with a replay buffer that is cleared at $T_1$, which has been shown to help with adaptation but cause catastrophic forgetting (Wan et al., 2022). Indeed, with a cleared replay buffer the model error dramatically worsens. Thus, Curious Replay is far more resilient to catastrophic forgetting than simply clearing the replay buffer. It achieves this by using a large buffer to store and revisit old experiences, but remains adaptable by using prioritization to focus training on the newer states.

Additionally, we used the Background-Swap Control Suite to assess policy forgetting (Figure 5). Curious Replay yields better adaptation at $T_0 = 1$M, and matches the baseline after $T_1 = 2$M, demonstrating that in this experiment Curious Replay does not elicit catastrophic forgetting. This was also true across four additional tasks (Figure A16, Table A7).

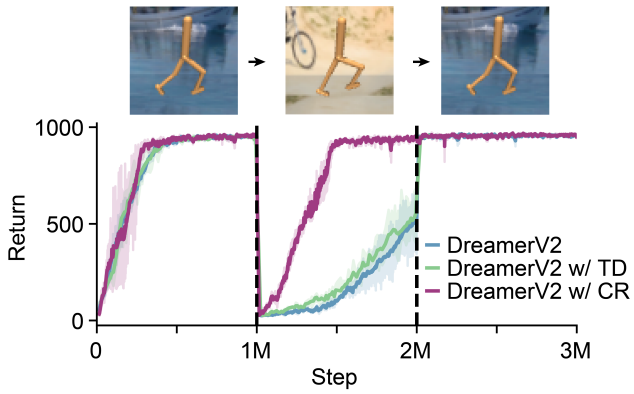


*Figure 5.* Background-Swap walker_walk. Background changes at step 1M, and reverts at step 2M. Curious Replay improves performance after 1M, without degrading performance after 2M.

## 5. Related work

**Model-based RL** Recent advances have propelled model-based RL to success on classic board games (Schrittwieser et al., 2020) and on tasks with high dimensional input (Ha & Schmidhuber, 2018) including benchmarks such as Deepmind Control Suite (Hafner et al., 2019a) and Atari (Kaiser et al., 2019; Hafner et al., 2020). These successes have been extended in a number of settings (Wu et al., 2022; Mendonca et al., 2021; Hafner et al., 2022; Micheli et al., 2022; Chen et al., 2022; Deng et al., 2022; Hafner et al., 2023).

**Adaptation** Adaptation has been studied as a component of continual learning (Thrun, 1998; Khetarpal et al., 2022; Julian et al., 2020; Xie et al., 2021). Approaches toward adaptation include using system identification (Bongard et al., 2006; Cully et al., 2015; Kumar et al., 2021), metalearning (Finn et al., 2017; Al-Shedivat et al., 2017; Song et al., 2020; Nagabandi et al., 2018; Yu et al., 2020), planning using recent states (Wan et al., 2022), and changepoint detection (Adams & MacKay, 2007; Fearnhead & Liu, 2007; Hadoux et al., 2014; Banerjee et al., 2017).

**Prioritized replay** Experience replay (Lin, 1992; Mnih et al., 2015) has been enhanced in a number of settings through prioritization. Priority signals include changes in value (Moore & Atkeson, 1993), temporal-difference error (Schaul et al., 2015; Pan et al., 2022), state frequency (Novati & Koumoutsakos, 2019; Sinha et al., 2022; Sun et al., 2020), and regret (Liu et al., 2021). The sampling strategy itself can even be optimized (Zha et al., 2019; Oh et al., 2020; Burega et al., 2022). Pioneering work has also investigated use of model errors for prioritization (Oh et al., 2021), with an implementation and application complementary to ours.

**Intrinsically-motivated RL** Curiosity-based intrinsic motivation has emerged as a valuable strategy for guiding exploration in sparsely-rewarded environments (Schmidhuber, 1991; Oudeyer et al., 2007). Such signals can include model error (Stadie et al., 2015; Pathak et al., 2017; Guo et al., 2022), surprise (Achiam & Sastry, 2017), model learning progress (Kim et al., 2020), model uncertainty (Pathak et al., 2019; Sekar et al., 2020), latent state novelty (Raileanu & Rocktäschel, 2020), information gain (Still & Precup, 2012; Houthooft et al., 2016), state novelty (Bellemare et al., 2016; Machado et al., 2020; Burda et al., 2018; Yarats et al., 2021; Tang et al., 2017; Parisi et al., 2021), diversity (Eysenbach et al., 2018), and empowerment (Klyubin et al., 2005; Mohamed & Jimenez Rezende, 2015; Gregor et al., 2016).

## 6. Discussion

**Limitations** There may be instances in which the Curious Replay signal is not aligned with a task, potentially because the task requires accuracy on a very specific subset of the world. Future work may incorporate task-related prioritization signals, or use different prioritizations for updating the world model versus the policy. Curious Replay is also not currently tailored to account for changing reward functions, as in the LoCA setting, which will be an interesting direction to investigate (Wan et al., 2022; Van Seijen et al., 2020). Although there was no evidence of catastrophic forgetting in our experiments, further steps can be taken mitigate possible forgetting, such as occasionally recalculating the prioritization weights across the buffer. Additionally, substantially increasing the speed of model training may mitigate the need for replay prioritization by allowing the model to frequently train on the entire replay buffer. Finally, although Curious Replay improved the speed to object interaction, agent behavior still does not fully mirror animal behavior (Figure A2), leaving room for future work.

**Animal inspiration** At its core, Curious Replay was inspired by animal behavior. The prowess of animals at adapting to changing environments highlighted clear deficiencies in existing model-based agents, and led us to develop Curious Replay to fix these shortcomings. The effectiveness of Curious Replay may allow the favor to be returned, by providing inspiration to the study of animal physiology. For example, Curious Replay suggests hypotheses about how animals might replay their past: experiences that are more recent or surprising should be replayed more frequently. Intriguingly, recent experiments have begun to uncover precisely this phenomenon, using electrical recordings of hippocampal replay in rats as they explore environments with varying levels of familiarity (Gorriz et al., 2023).

**Outlook** We present Curious Replay, a method that improves agent adaptation in changing environments. Inspired by the use of curiosity as an intrinsic motivation for selecting actions, we use curiosity signals for selecting experiences to use during training. Future directions include incorporating Curious Replay into other model-based and model-free agents, investigating use of additional curiosity signals, and testing performance in an even wider variety of settings. In sum, by using curiosity to guide experience replay, Curious Replay opens avenues for more effective world model training, exploration, and adaptation.

## 7. Acknowledgements

I.K. is a Merck Awardee of the Life Sciences Research Foundation, and a Wu Tsai Stanford Neurosciences Institute Interdisciplinary Scholar. Thank you to Karl Deisseroth for access to mice for animal experiments. This work is also in part funded by Human-Centered AI Hoffman-Yee and Google Cloud Credit Grants, the Stanford Graduate School of Education and the Stanford Accelerator for Learning.

## References


Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

Abbas, Z., Sokota, S., Talvitie, E., and White, M. Selective dyna-style planning under limited model capacity. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1–10. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/abbas20a.html.

Achiam, J. and Sastry, S. Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv preprint arXiv:1703.01732*, 2017.

Adams, R. P. and MacKay, D. J. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

Ahmadlou, M., Houba, J. H., van Vierbergen, J. F., Giannouli, M., Gimenez, G.-A., van Weeghel, C., Darbanfouladi, M., Shirazi, M. Y., Dziubek, J., Kacem, M., et al. A cell type–specific cortico-subcortical brain circuit for investigatory and novelty-seeking behavior. *Science*, 372 (6543):eabe9681, 2021.

Al-Shedivat, M., Bansal, T., Burda, Y., Sutskever, I., Mordatch, I., and Abbeel, P. Continuous adaptation via meta-learning in nonstationary and competitive environments. *arXiv preprint arXiv:1710.03641*, 2017.

Balloch, J., Lin, Z., Wright, R., Peng, X., Hussain, M., Srinivas, A., Kim, J., and Riedl, M. O. Neuro-symbolic world models for adapting to open world novelty. *arXiv preprint arXiv:2301.06294*, 2023.

Banerjee, T., Liu, M., and How, J. P. Quickest change detection approach to optimal control in markov decision processes with model changes. In *2017 American control conference (ACC)*, pp. 399–405. IEEE, 2017.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

Berseth, G., Geng, D., Devin, C., Rhinehart, N., Finn, C., Jayaraman, D., and Levine, S. Smirl: Surprise minimizing reinforcement learning in unstable environments. *arXiv preprint arXiv:1912.05510*, 2019.

Bodnar, C., Hausman, K., Dulac-Arnold, G., and Jonschkowski, R. A geometric perspective on self-supervised policy adaptation. *arXiv preprint arXiv:2011.07318*, 2020.

Bongard, J., Zykov, V., and Lipson, H. Resilient machines through continuous self-modeling. *Science*, 314(5802): 1118–1121, 2006.

Burda, Y., Edwards, H., Storkey, A., and Klimov, O. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Burega, B., Martin, J. D., and Bowling, M. Learning to prioritize planning updates in model-based reinforcement learning. In *Sixth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems*, 2022.

Buzzega, P., Boschini, M., Porrello, A., and Calderara, S. Rethinking experience replay: a bag of tricks for continual learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 2180–2187. IEEE, 2021.

Cassirer, A., Barth-Maron, G., Brevdo, E., Ramos, S., Boyd, T., Sottiaux, T., and Kroiss, M. Reverb: A framework for experience replay, 2021.

Chen, C., Wu, Y.-F., Yoon, J., and Ahn, S. Transdreamer: Reinforcement learning with transformer world models. *arXiv preprint arXiv:2202.09481*, 2022.

Clark, A. Pillow (pil fork) documentation, 2015. URL https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf.

Cully, A., Clune, J., Tarapore, D., and Mouret, J.-B. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

Da Silva, B. C., Basso, E. W., Bazzan, A. L., and Engel, P. M. Dealing with non-stationary environments using context detection. In *Proceedings of the 23rd international conference on Machine learning*, pp. 217–224, 2006.

Deng, F., Jang, I., and Ahn, S. Dreamerpro: Reconstruction-free model-based reinforcement learning with prototypical representations. In *International Conference on Machine Learning*, pp. 4956–4975. PMLR, 2022.

D'Oro, P., Schwarzer, M., Nikishin, E., Bacon, P.-L., Bellemare, M. G., and Courville, A. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *Deep Reinforcement Learning Workshop NeurIPS*, 2022.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

Fearnhead, P. and Liu, Z. On-line inference for multiple changepoint problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(4):589–605, 2007.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Finn, C., Rajeswaran, A., Kakade, S., and Levine, S. Online meta-learning. In *International Conference on Machine Learning*, pp. 1920–1930. PMLR, 2019.

Glickman, S. E. and Sroges, R. W. Curiosity in zoo animals. *Behaviour*, 26(1-2):151–187, 1966.

Gorriz, M. H., Takigawa, M., and Bendor, D. The role of experience in prioritizing hippocampal replay. *bioRxiv*, pp. 2023–03, 2023.

Gregor, K., Rezende, D. J., and Wierstra, D. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

Guo, Z. D., Thakoor, S., Pîslar, M., Pires, B. A., Altché, F., Tallec, C., Saade, A., Calandriello, D., Grill, J.-B., Tang, Y., et al. Byol-explore: Exploration by bootstrapped prediction. *arXiv preprint arXiv:2206.08332*, 2022.

Ha, D. and Schmidhuber, J. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

Haber, N., Mrowca, D., Wang, S., Fei-Fei, L. F., and Yamins, D. L. Learning to play with intrinsically-motivated, self-aware agents. *Advances in neural information processing systems*, 31, 2018.

Hadoux, E., Beynier, A., and Weng, P. Sequential decision-making under non-stationary environments via sequential change-point detection. In *Learning over multiple contexts (LMCE)*, 2014.

Hafner, D. Benchmarking the spectrum of agent capabilities. *arXiv preprint arXiv:2109.06780*, 2021.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019b.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Hafner, D., Lee, K.-H., Fischer, I., and Abbeel, P. Deep hierarchical planning from pixels. *arXiv preprint arXiv:2206.04114*, 2022.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Hansen, N., Jangir, R., Sun, Y., Alenyà, G., Abbeel, P., Efros, A. A., Pinto, L., and Wang, X. Self-supervised policy adaptation during deployment. *arXiv preprint arXiv:2007.04309*, 2020.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Harrison, J., Sharma, A., Finn, C., and Pavone, M. Continuous meta-learning without tasks. *Advances in neural information processing systems*, 33:17571–17581, 2020.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Horgan, D., Quan, J., Budden, D., Barth-Maron, G., Hessel, M., Van Hasselt, H., and Silver, D. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., and Abbeel, P. Vime: Variational information maximizing exploration. *Advances in neural information processing systems*, 29, 2016.

Hunter, J. D. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

Javed, K. and White, M. Meta-learning representations for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Julian, R., Swanson, B., Sukhatme, G. S., Levine, S., Finn, C., and Hausman, K. Never stop learning: The effectiveness of fine-tuning in robotic reinforcement learning. *arXiv preprint arXiv:2004.10190*, 2020.

Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

Khetarpal, K., Riemer, M., Rish, I., and Precup, D. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.

Kim, K., Sano, M., De Freitas, J., Haber, N., and Yamins, D. Active world model learning with progress curiosity. In *International conference on machine learning*, pp. 5306–5315. PMLR, 2020.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Klyubin, A. S., Polani, D., and Nehaniv, C. L. All else being equal be empowered. In *European Conference on Artificial Life*, pp. 744–753. Springer, 2005.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

Kumar, A., Gupta, A., and Levine, S. Discor: Corrective feedback in reinforcement learning via distribution correction. *Advances in Neural Information Processing Systems*, 33:18560–18572, 2020.

Kumar, A., Fu, Z., Pathak, D., and Malik, J. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

Laskin, M., Yarats, D., Liu, H., Lee, K., Zhan, A., Lu, K., Cang, C., Pinto, L., and Abbeel, P. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

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

Liu, X.-H., Xue, Z., Pang, J., Jiang, S., Xu, F., and Yu, Y. Regret minimization experience replay in off-policy reinforcement learning. *Advances in Neural Information Processing Systems*, 34:17604–17615, 2021.

Machado, M. C., Bellemare, M. G., and Bowling, M. Count-based exploration with the successor representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5125–5133, 2020.

Mendonca, R., Rybkin, O., Daniilidis, K., Hafner, D., and Pathak, D. Discovering and achieving goals via world models. *Advances in Neural Information Processing Systems*, 34:24379–24391, 2021.

Micheli, V., Alonso, E., and Fleuret, F. Transformers are sample efficient world models. *arXiv preprint arXiv:2209.00588*, 2022.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Moerland, T. M., Broekens, J., Plaat, A., Jonker, C. M., et al. Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 16(1): 1–118, 2023.

Mohamed, S. and Jimenez Rezende, D. Variational information maximisation for intrinsically motivated reinforcement learning. *Advances in neural information processing systems*, 28, 2015.

Moore, A. W. and Atkeson, C. G. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine learning*, 13(1):103–130, 1993.

Nagabandi, A., Finn, C., and Levine, S. Deep online learning via meta-learning: Continual adaptation for model-based rl. *arXiv preprint arXiv:1812.07671*, 2018.

Novati, G. and Koumoutsakos, P. Remember and forget for experience replay. In *International Conference on Machine Learning*, pp. 4851–4860. PMLR, 2019.

Oh, Y., Lee, K., Shin, J., Yang, E., and Hwang, S. J. Learning to sample with local and global contexts in experience replay buffer. *arXiv preprint arXiv:2007.07358*, 2020.

Oh, Y., Shin, J., Yang, E., and Hwang, S. J. Model-augmented prioritized experience replay. In *International Conference on Learning Representations*, 2021.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. V. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2): 265–286, 2007.

Pan, Y., Mei, J., and Farahmand, A.-m. Frequency-based search-control in dyna. *arXiv preprint arXiv:2002.05822*, 2020.

Pan, Y., Mei, J., Farahmand, A.-m., White, M., Yao, H., Rohani, M., and Luo, J. Understanding and mitigating the limitations of prioritized replay. In *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022.

Parisi, S., Dean, V., Pathak, D., and Gupta, A. Interesting object, curious agent: Learning task-agnostic exploration. *Advances in Neural Information Processing Systems*, 34: 20516–20530, 2021.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

Pathak, D., Gandhi, D., and Gupta, A. Self-supervised exploration via disagreement. In *International conference on machine learning*, pp. 5062–5071. PMLR, 2019.

Raileanu, R. and Rocktäschel, T. Ride: Rewarding impact-driven exploration for procedurally-generated environments. *arXiv preprint arXiv:2002.12292*, 2020.

Ritter, S., Wang, J., Kurth-Nelson, Z., Jayakumar, S., Blundell, C., Pascanu, R., and Botvinick, M. Been there, done that: Meta-learning with episodic recall. In *International conference on machine learning*, pp. 4354–4363. PMLR, 2018.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., and Wayne, G. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Sagawa, S., Koh, P. W., Lee, T., Gao, I., Xie, S. M., Shen, K., Kumar, A., Hu, W., Yasunaga, M., Marklund, H., et al. Extending the wilds benchmark for unsupervised adaptation. *arXiv preprint arXiv:2112.05090*, 2021.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Schmidhuber, J. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*, pp. 222–227, 1991.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

Sekar, R., Rybkin, O., Daniilidis, K., Abbeel, P., Hafner, D., and Pathak, D. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pp. 8583–8592. PMLR, 2020.

Sinha, S., Song, J., Garg, A., and Ermon, S. Experience replay with likelihood-free importance weights. In *Learning for Dynamics and Control Conference*, pp. 110–123. PMLR, 2022.

Song, X., Yang, Y., Choromanski, K., Caluwaerts, K., Gao, W., Finn, C., and Tan, J. Rapidly adaptable legged robots via evolutionary meta-learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3769–3776. IEEE, 2020.

Stadie, B. C., Levine, S., and Abbeel, P. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*, 2015.

Still, S. and Precup, D. An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences*, 131(3):139–148, 2012.

Stone, A., Ramirez, O., Konolige, K., and Jonschkowski, R. The distracting control suite–a challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

Sun, P., Zhou, W., and Li, H. Attentive experience replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5900–5907, 2020.

Tang, H., Houthooft, R., Foote, D., Stooke, A., Xi Chen, O., Duan, Y., Schulman, J., DeTurck, F., and Abbeel, P. # exploration: A study of count-based exploration for deep reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Team, A. A., Bauer, J., Baumli, K., Baveja, S., Behbahani, F., Bhoopchand, A., Bradley-Schmieg, N., Chang, M., Clay, N., Collister, A., et al. Human-timescale adaptation in an open-ended task space. *arXiv preprint arXiv:2301.07608*, 2023.

Thrun, S. Lifelong learning algorithms. In *Learning to learn*, pp. 181–209. Springer, 1998.

Tunyasuvunakool, S., Muldal, A., Doron, Y., Liu, S., Bohez, S., Merel, J., Erez, T., Lillicrap, T., Heess, N., and Tassa, Y. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638. doi: https://doi.org/10.1016/j.simpa.2020.100022. URL https://www.sciencedirect.com/science/article/pii/S2665963820300099.

Van Seijen, H., Nekoei, H., Racah, E., and Chandar, S. The loca regret: a consistent metric to evaluate model-based behavior in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:6562–6572, 2020.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Wan, Y., Rahimi-Kalahroudi, A., Rajendran, J., Momennejad, I., Chandar, S., and Van Seijen, H. H. Towards evaluating adaptivity of model-based reinforcement learning methods. In *International Conference on Machine Learning*, pp. 22536–22561. PMLR, 2022.

Waskom, M. L. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021. URL https://doi.org/10.21105/joss.03021.

Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman (eds.), *Proceedings of the 9th Python in Science Conference*, pp. 56 – 61, 2010. doi: 10.25080/Majora-92bf1922-00a.

Wu, P., Escontrela, A., Hafner, D., Goldberg, K., and Abbeel, P. Daydreamer: World models for physical robot learning. *arXiv preprint arXiv:2206.14176*, 2022.

Xie, A., Harrison, J., and Finn, C. Deep reinforcement learning amidst continual structured non-stationarity. In *International Conference on Machine Learning*, pp. 11393–11403. PMLR, 2021.

Yarats, D., Kostrikov, I., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2020.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Reinforcement learning with prototypical representations. In *International Conference on Machine Learning*, pp. 11920–11931. PMLR, 2021.

Yu, W., Tan, J., Bai, Y., Coumans, E., and Ha, S. Learning fast adaptation with meta strategy optimization. *IEEE Robotics and Automation Letters*, 5(2):2950–2957, 2020.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

Zha, D., Lai, K.-H., Zhou, K., and Hu, X. Experience replay optimization. *arXiv preprint arXiv:1906.08387*, 2019.

Zintgraf, L., Shiarli, K., Kurin, V., Hofmann, K., and Whiteson, S. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, pp. 7693–7702. PMLR, 2019.

## A. Societal Impact

This work has the potential for wide-ranging applications in autonomous systems such as robotics and decision systems. Such systems offer the potential for many benefits, including with manufacturing assistance, healthcare, transportation, and scientific discovery. However, they also have the potential for negative societal impact, such as in autonomous weapons or workforce displacement. Moreover, they present risks if they are allowed to freely explore and interact with the environment, and safeguards must be put in place to ensure that such interactions do not harm humans and animals. Additionally, consideration must be made to protect against biases in an agent's learned model of the world that may not align with human values. Developing methods to test for and potentially correct such biases may thus also be an important line of research.

## B. Extended Related Work

**Model-based RL** At least two key advantages are promised by model-based reinforcement learning relative to model-free RL: more sample efficient training due to the compression of experience into a predictive world model, and more effective policies that can leverage the world model for planning. Recent advances have propelled model-based RL to success on classic board games (Schrittwieser et al., 2020) and on tasks with high dimensional input (Ha & Schmidhuber, 2018) including benchmarks such as Deepmind Control Suite (Hafner et al., 2019a) and Atari (Kaiser et al., 2019; Hafner et al., 2020). These successes have been extended to robotics settings (Wu et al., 2022), goal-conditioning (Mendonca et al., 2021), hierarchical planning (Hafner et al., 2022), use of a transformer architecture for improved sample-efficiency (Micheli et al., 2022) and memory-based reasoning (Chen et al., 2022), and use of prototypes for enhanced robustness against distractions (Deng et al., 2022). Remaining challenges include fast adaptation, and improved model performance.

**Adaptation** Quick adjustment in response to changing tasks, data distributions, or environment conditions can be a very challenging problem, and is a component of continual learning (Thrun, 1998; Khetarpal et al., 2022; Julian et al., 2020; Xie et al., 2021). In robotics, system identification can be used to inform adaptation to a changing environment (Kumar et al., 2021) or body (Bongard et al., 2006; Cully et al., 2015). Metalearning methods (Finn et al., 2017) train a model across a variety of situations such that it can adapt quickly to a new task through few-shot fine-tuning. In this approach, by seeing many different examples of how an environment might change, the agent can develop strategies for adapting and focusing on the essential aspects of an environment (Al-Shedivat et al., 2017; Song et al., 2020; Harrison et al., 2020; Nagabandi et al., 2018; Finn et al., 2019; Javed & White, 2019; Zintgraf et al., 2019; Yu et al., 2020). Self-supervised objectives can be used to guide policy adaptation in changing environments (Hansen et al., 2020; Bodnar et al., 2020; Balloch et al., 2023). Changepoint detection (Adams & MacKay, 2007; Fearnhead & Liu, 2007; Hadoux et al., 2014; Banerjee et al., 2017) and context detection (Da Silva et al., 2006) can be used to directly signal that adaptation is required. Considerations have also been made towards the problem of catastrophic forgetting, wherein a model may lose capabilities it had previously learned (Kirkpatrick et al., 2017; Li & Hoiem, 2017; Zenke et al., 2017; Rolnick et al., 2019; Buzzega et al., 2021; Ritter et al., 2018; Rusu et al., 2016). Robustness to shifting data distributions is also a problem in supervised and self-supervised learning, and benchmarks such as WILDS have been established to encourage progress (Koh et al., 2021; Sagawa et al., 2021).

**Prioritized replay** Experience replay consists of storing experienced episodes in a buffer and revisiting them during model updates (Lin, 1992). This increases efficiency by enabling data reuse, and aids training by breaking correlations that would arise in the consecutive data samples of a strictly online approach (Mnih et al., 2015). Uniform sampling from the replay buffer is a common approach (Hafner et al., 2019a). However, it is well known that prioritized sampling can yield benefits, with emphasis on prioritizing by the change in value (Moore & Atkeson, 1993). In model-free contexts, there has been particular success prioritizing by the magnitude of the temporal-difference error (Schaul et al., 2015; Horgan et al., 2018; Hessel et al., 2018). Alternatively, more frequent states under the current policy can be prioritized (Novati & Koumoutsakos, 2019; Sinha et al., 2022; Sun et al., 2020). Prioritization can also aim to minimize regret (Liu et al., 2021), error in the value function (Kumar et al., 2020), or where the value function is more difficult to predict (Pan et al., 2020). Variations in the sampling strategy can account for the context of data relative to the rest of the replay buffer (Oh et al., 2020) or can mitigate deficiencies of outdated priorities and insufficient sample space coverage (Pan et al., 2022). The sampling strategy itself can even be optimized (Zha et al., 2019; Burega et al., 2022). In model-based contexts, states that the model can confidently make predictions about can be prioritized (Abbas et al., 2020). Particularly relevant and complementary to Curious replay is the work of (Oh et al., 2021), which investigates the use of model error for prioritization. They augment the Q-function with a parameter-sharing one-step state prediction, and uses the error of that prediction to aid prioritization. This is similar to the adversarial component of our Curious Replay, but differs in that the model itself is not used for planning. They also do not

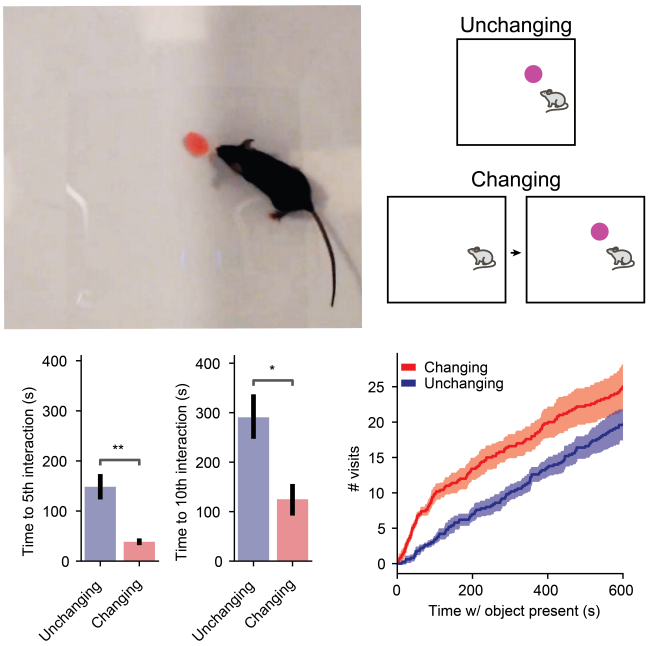


*Figure A1.* Novel-object interaction of wildtype mice in changing and unchanging environments. The 5th and 10th interactions both occur after significantly less time in the changing environment (n=5 each condition, independent t-test, * $p < 0.05$, ** $p < 0.001$).

combine model error with a count-based novelty, which we find to be particularly important. Perhaps most importantly, we demonstrate the utility of a model-loss-based prioritization for improving adaptation performance in the face of changing environments, which is not something they investigate. Moreover, while they rely on modifying the critic network to add a transition model, our approach does not require such a modification and instead uses the loss computed directly during world model updates of model-based agents. Curious Replay is explicitly targeted at improving algorithms like Dreamer that are fundamentally reliant on a world model.

**Intrinsically-motivated RL** Intrinsically-motivated reinforcement learning has emerged as a strategy for guiding exploration in sparsely-rewarded environments (Oudeyer et al., 2007). For many such methods, world models play a central role in computing the intrinsic motivation signal (Schmidhuber, 1991). Such signals can include model error (Stadie et al., 2015) as with ICM (Pathak et al., 2017) and BYOL-Explore (Guo et al., 2022); surprise (Achiam & Sastry, 2017); model learning progress as with $\gamma$-progress (Kim et al., 2020); model uncertainty as with ensemble disagreement (Pathak et al., 2019; Sekar et al., 2020); novelty of latent state representations as with RIDE (Raileanu & Rocktäschel, 2020); and information gain (Still & Precup, 2012; Houthooft et al., 2016). Many signals do not require an explicit world model, such as novelty as with pseudocounts of states (Bellemare et al., 2016; Machado et al., 2020) and changes (Parisi et al., 2021), random network distillation (Burda et al., 2018), state-hashing (Tang et al., 2017), entropy with prototypical representations (Yarats et al., 2021); diversity (Eysenbach et al., 2018); and empowerment (Klyubin et al., 2005; Mohamed & Jimenez Rezende, 2015; Gregor et al., 2016).

## C. Validation of expected object interaction behavior with mice

We validated the expected animal behavior with C57Bl/6 mice (Black 6, Jackson Laboratory, 664). Mice were allowed to freely investigate a 12" x 12" white acrylic box. In the unchanging environment condition, a small ($\sim$0.5" diameter) pink rubber ball was present in the box from the outset, before the mouse was placed in the box. In the changing environment condition, the ball was placed in the box (using metal tongs) after 10 minutes of exploration of the empty box. In both conditions, the mouse was allowed to explore the box while the ball was present for ten minutes. The behavior was recorded using a webcam, and the time of each interaction between the mouse and ball was manually scored. All procedures were performed in accordance with protocols approved by the Stanford University Institutional Animal Care and Use Committee (IACUC) and guidelines of the National Institutes of Health.

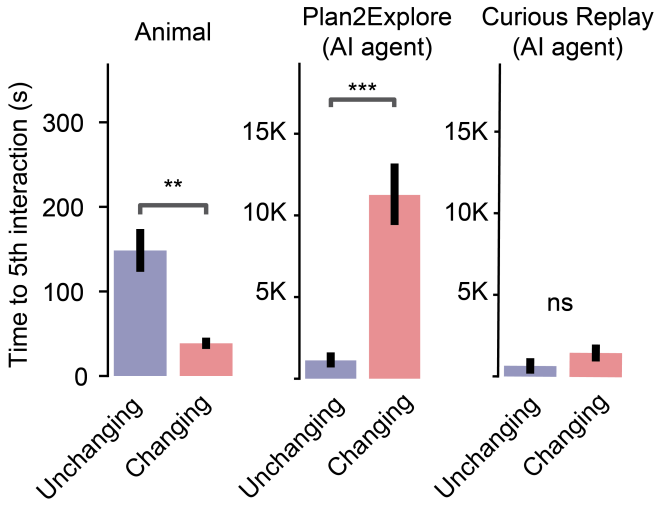


*Figure A2.* Comparison of animal (mouse) behavior and AI agent behavior. Baseline Dreamer AI agent has the opposite behavior as animals, with a longer time to interaction in the changing environment. Curious Replay reduces this gap in behavior, bring behavior in the changing environment on par with in the unchanging environment. There still remains a gap, however, that future work can investigate. AI agent steps have been converted into seconds using the 0.03 seconds/step timescale of the simulated environment. (Animal: n=5 each condition, Baseline: n=7 each condition, Curious Replay: n=7 each condition, independent t-test, * $p < 0.05$, ** $p < 0.001$).

# D. Technical details

## D.1. Implementation details

We use the STArr python package as a fast SumTree implementation for DreamerV2 and DreamerPro. For Curious Replay, a running minimum (across the entire run) was subtracted from the loss before a priority value was computed. We did not use the running minimum when prioritizing with temporal difference. Dreamer is implemented in Tensorflow2. We customize the MuJoCo-based dm_control library for object interaction assay implementation, with a control timestep of 0.03 s and a physics simulation timestep of 0.005 s. For the Background-Swap Control Suite, a ground-plane alpha of 0.1 is used. Code will be made publicly available.

In DreamerV3, sequences of length 64 are stored in the replay buffer. In the Curious Replay implementation for DreamerV3, the probability of training on a sequence is based on the priority calculated for the last step of the sequence. The Reverb replay buffer (Cassirer et al., 2021) is used to store the sequences and priorities and to select the samples. After each training step, the training count and priority for each step in the sequence is updated. Unlike in the DreamerV2 implementation, the loss used to calculate the priority is not adjusted by the running minimum.

Episode length is 1K steps for Control Suite and 100K steps for object interaction to allow for substantial uninterrupted exploration. In Crafter, episode length depends on the agent's survival. An action repeat of 2 (Hafner et al., 2019a) is applied across object interaction and Control Suite environments, with no action repeat for Crafter.

## D.2. Detailed hyperparameters

For object interaction and Control Suite, batches are $B = 10$ sequences of fixed length $L = 50$. For continuous control tasks, we used a nondiscrete latent space, which we found performed better. For Crafter, we used a slightly more updated version of DreamerV2 that includes layer normalization, $B = 16$ and a discrete latent space. Model training and evaluation used Google Cloud T4 GPU instances. Each agent's online return is logged every episode for Control Suite, and every 20 steps for object interaction.

**Object interaction assay and Control Suite** The Recurrent State Space Machine (RSSM) is nondiscrete with a 200 unit hidden state, a GRU with 200 units, and a stochastic state with 32 units. The convolutional image encoder has 4 layers with depth 48, kernel sizes of 4, and an output of size 512. The convolutional image decoder has 4 layers of depth 48, kernels of size [5, 5, 6, 6], and an output of size 64 x 64 x 3. The reward prediction head, actor, and critic are each an MLP with 4

layers of 400 units. The world model is trained using just the image gradient. The actor is trained using straight-through dynamics backpropagation. World model optimization uses Adam with learning rate of 3e-4 and epsilon 1e-5, weight decay 1e-6, and clipping of 100, and actor-critic optimization has a learning rate of 8e-5. The models are trained every 5 environment steps. Plan2Explore uses a disagreement ensemble of 10 action-conditioned one-step MLP models with 4 layers each with 400 units. The disagreement target is the stochastic state. Ensemble optimization uses Adam with learning rate 3e-4 and epsilon 1e-5.

**Crafter** The settings for Crafter are the same, except that the RSSM is discrete with 1024 unit hidden layer, 1024 unit GRU, and 32 x 32 dimensional discrete stochastic state. Layer normalization is also applied. Optimization of the world model and actor critic use Adam with a learning rate of 1e-4, epsilon 1e-5, weight decay 1e-6, and clipping of 100.

## E. Additional discussion of Curious Replay

We tested the hypothesis that Dreamer's uniform replay buffer sampling does not promote training the world model on new data from the changed environment. In the object interaction assay, we cleared the replay buffer at $T_0$, when the environment changed. Clearing the buffer, which forces the model to train only on new data, substantially improves the time to object interaction (Figure A3), suggesting that a key issue is how data is sampled from the replay buffer. However, clearing the replay buffer is a problematic approach. In particular, it is important to not be trigger-happy in clearing the buffer, and thus high accuracy for detecting changepoints is required. Moreover, even with accurate changepoint detection, prematurely clearing valuable data may lead to catastrophic forgetting or other issues. We thus sought a solution with the performance benefits of clearing the buffer but with more generality and less brittleness. Count-based Replay is one such solution, inspired by count-based exploration (Bellemare et al., 2016).

## F. Method comparisons

For comparison to other methods, we used the official implementations corresponding to the publications. For DreamerV2, we forked commit 3a711b4. For DreamerPro, we forked commit 809b417. For IRIS, we forked commit e6aaa67. For DreamerV3, we forked commit 84ecf191.

## G. Helpful python libraries

We thank the creators of matplotlib (Hunter, 2007), pandas (Wes McKinney, 2010), seaborn (Waskom, 2021), scipy (Virtanen et al., 2020), numpy (Harris et al., 2020), pillow (Clark, 2015), and tensorflow (Abadi et al., 2015).

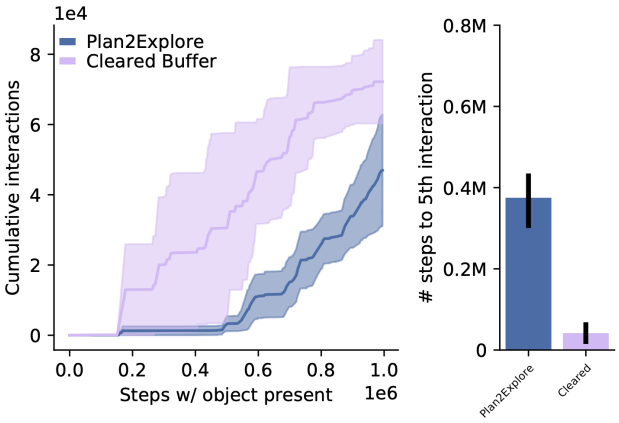


*Figure A3.* Clearing the replay buffer at the time the environment changes (step 5e5) substantially improves the time to object interaction. (Baseline n=7, Cleared buffer n=2).

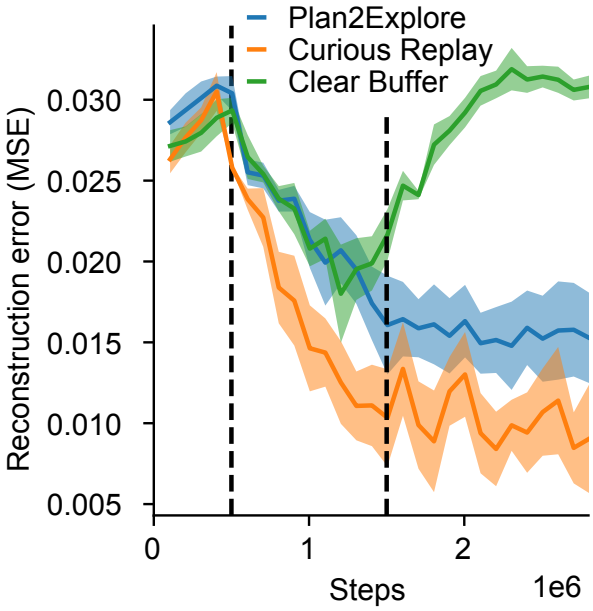


*Figure A4.* Testing catastrophic forgetting of the world model. Here, the object is introduced at step 0.5e6 and removed at step 1.5e6. Model performance is assessed using a test suite of trajectories that include the object. There are clear signs of catastrophic forgetting when clearing the replay buffer (at step 1.5e6) but not with Curious Replay (n=6, mean $\pm$ s.e.m.).

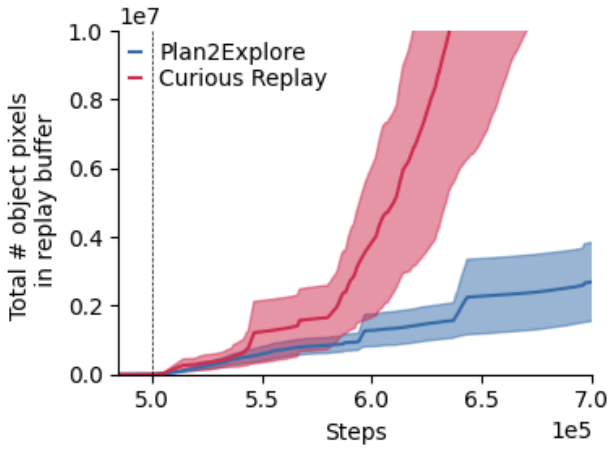


*Figure A5.* The agents are quickly exposed to the object once it appears in the environment. By quantifying the number of object pixels that are in the replay buffer, we see that at step 5e5 (the step when the object is introduced into the environment) the agent begins to experience observing the object (n=7, 7 for uniform and MT agents, mean +/- s.e.m.). For the first 5e4 steps, Curious Replay and Plan2Explore experience the object similarly, but diverge as Curious Replay begins to interact with the object sooner.

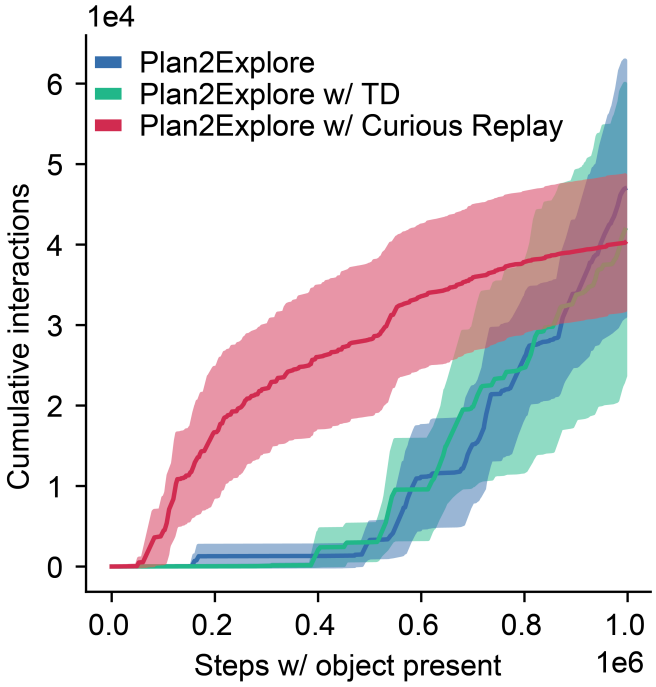


*Figure A6.* Cumulative object interactions over time. (n=7 each condition, mean +/- s.e.m.).

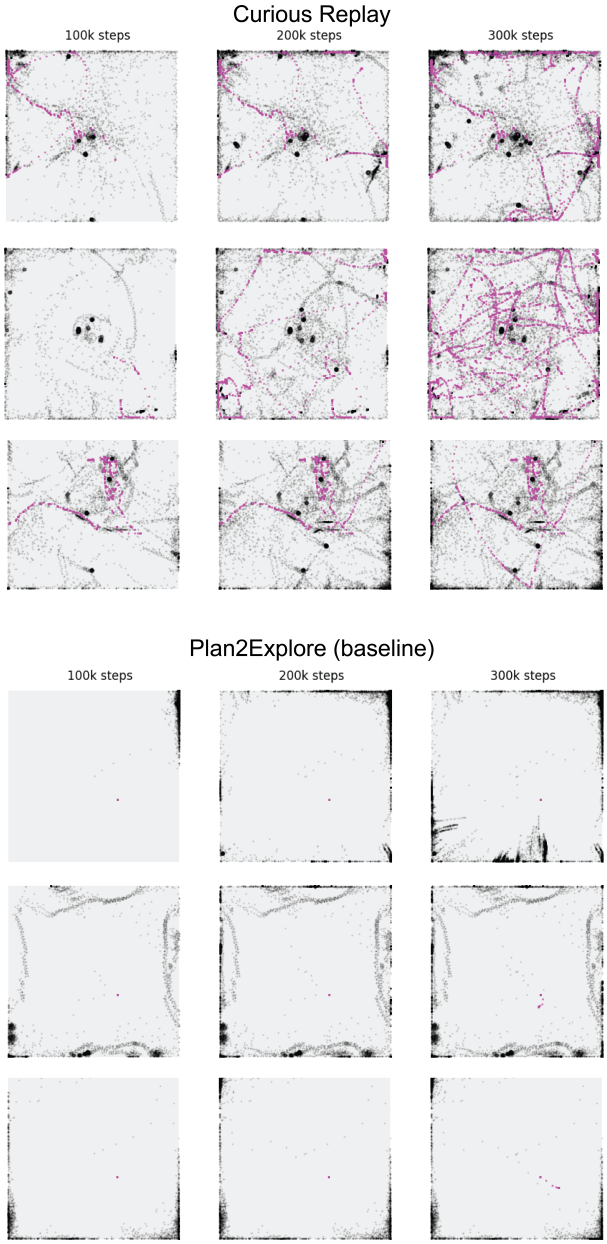


*Figure A7.* Example baseline Dreamer and Curious Replay agent trajectories at 100K, 200K, and 300K steps after the object has been introduced. Black dot represents the location of the agent, and magenta represents the location of the object.

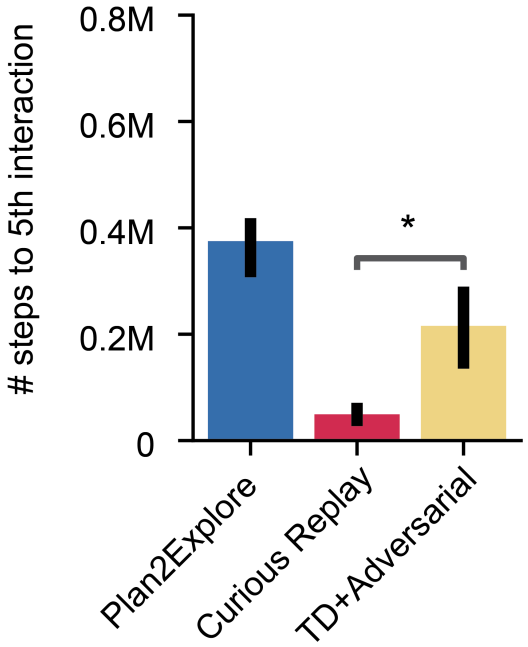


*Figure A8.* Comparison of Curious Replay with combined TD + Adversarial prioritization. To combine TD and Adversarial priorities while accounting for potentially very different scaling of the priority values, two separate priority arrays were used, and experiences were sampled according to each prioritization, with a fraction $f \in [0, 1]$ experiences according to the TD prioritization, and $(1 - f)$ according to the Adversarial prioritization. Results shown here investigate a range of $f$ values from 0.1 to 0.9, none of which yielded a result on par with Curious Replay (n=7 Curious Replay, n=6 total TD+Adversarial, mean +/- s.e.m., independent t-test).

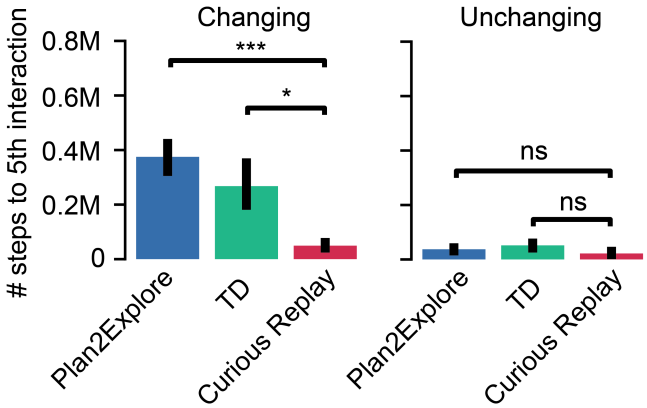


*Figure A9.* Object interaction assay, comparing Plan2Explore, Plan2Explore with temporal-difference prioritization (TD), and Plan2Explore with Curious Replay (ours). Curious Replay is significantly quicker in the changing environment. In the unchanging environment, Curious Replay remains as fast as Plan2Explore (n=7 each condition, independent t-test with fdr-bh correction).

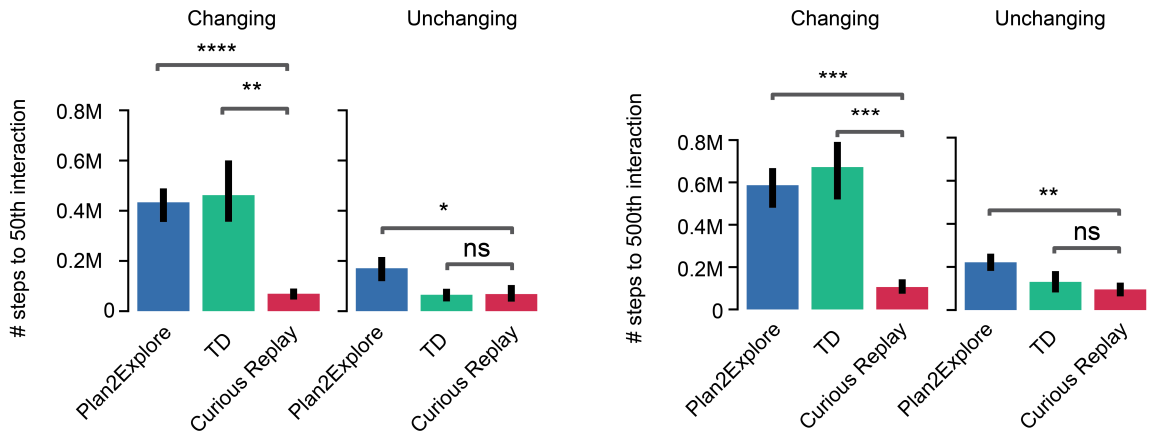


*Figure A10.* Robustness of Curious Replay performance to measurement by different metrics of object interactions (time to 50th or 500th interaction). (n=7 each condition, mean +/- s.e.m., independent t-test with fdr-bh correction).

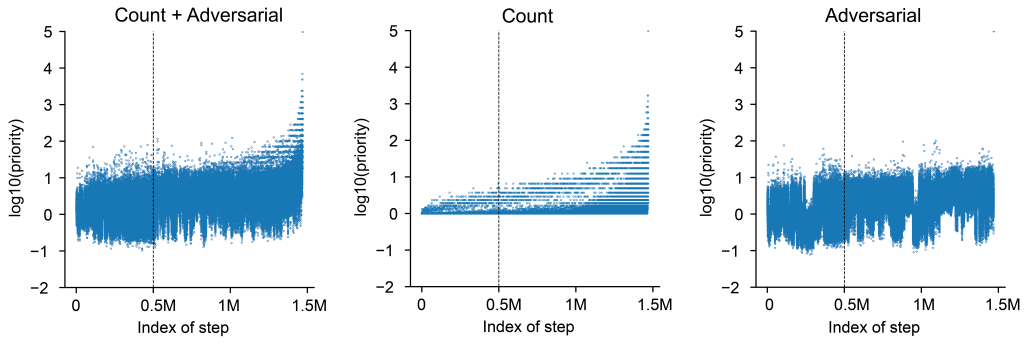


*Figure A11.* Example prioritizations across replay buffer containing 1.5e6 steps from playground environment (object is introduced at step 0.5e6).

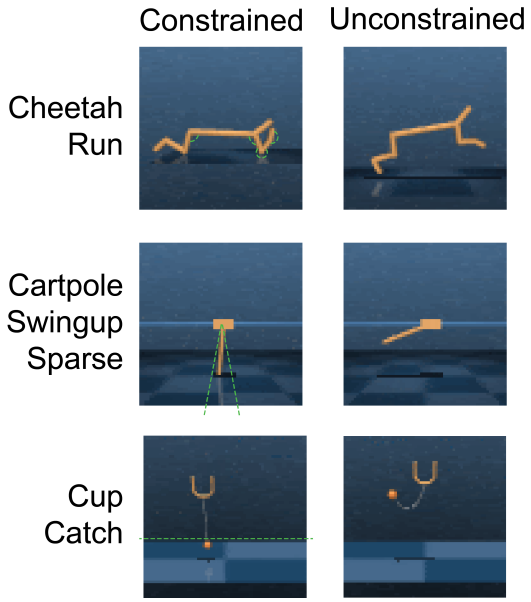


*Figure A12.* Constrained Control Suite tasks. Green dashed lines represent the constraints. For cup, the z-range of the ball was limited to [-.151 -.091]. For cartpole, the hinge of the pole was limited to angles [175 185]. For cheetah, the limits of the joints were bthigh [50 60], fthigh [-57 -52], fshin [-70 -60], ffoot [-28 -26]. In all cases, at $T_0 = 500K$, all constraints were released.

*Table A1.* Hyperparameter optimization on object interaction assay (n=2 seeds per setting, mean $\pm$ s.e.m.). The meaningful hyperparameters are $\alpha$ and $\beta$ (see Equation 1, which control the degree of adversarial prioritization, and the fall-off rate of count-based prioritization, respectively). The hyperparameter $c$ can further trade off the relative importance of the two prioritizations; we set it to scale the count-based signal to be slightly stronger than the adversarial signal (as informed by the scale of the average model loss, here we set $c = 1e4$). We set $\epsilon$ to 0.01 and never modified, and $maxval$ was set such that it was greater than any other priority. Thus $\alpha$ , $\beta$, and potentially $c$ are the only hyperparameters that require attention. We optimized the hyperparameters in the object-interaction assay, as we were developing the method. Due to computational cost, we did not do a full grid search. We anchored our search around $\alpha$=0.7 and $\beta$=0.7, based on the hyperparameters in the original Prioritized Experience Replay paper. Our search varied $\alpha$ or $\beta$ , with the other value set to 0.7. We found that higher values of $\alpha$ and $\beta$ were better, with ($\alpha$ =0.7, $\beta$=0.7) the best among settings that we tried. Importantly, all tested hyperpameter values yielded better object interaction than the Plan2Explore baseline. We thus used ($\alpha$ =0.7, $\beta$=0.7) for the rest of the experiments.

| **Value of $\beta$ (with $\alpha$=0.7)** | Baseline | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
|---|---|---|---|---|---|---|
| # Steps to 5th interaction (x$10^5$) | $3.7 \pm 0.6$ | $1.2 \pm 0.9$ | $1.1 \pm 0.3$ | $1.4 \pm 0.4$ | $\mathbf{0.5 \pm 0.1}$ | $0.6 \pm 0.2$ |

| **Value of $\alpha$ (with $\beta$=0.7)** | Baseline | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
|---|---|---|---|---|---|---|
| # Steps to 5th interaction (x$10^5$) | $3.7 \pm 0.6$ | $2.5 \pm 1.2$ | $1.5 \pm 0.4$ | $2.0 \pm 0.6$ | $\mathbf{0.5 \pm 0.1}$ | $0.9 \pm 0.3$ |

*Table A2.* Hyperparameter optimization on Crafter ($10 \geq n \geq 5$ seeds per setting, mean $\pm$ s.d.). To assess sensitivity to Curious Replay hyperameters, we tested a variety of hyperparameters for DreamerV2 with CR on Crafter, by varying either $\alpha$ or $\beta$ , with the other value set to 0.7. We further tested the impact of using $c$=1e3 instead of $c$=1e4 (the value we had used for all other experiments). We found that all tested hyperparameter settings yielded better Crafter performance than the DreamerV2 and DreamerV2 w/ TD baselines. Notably, some of the hyperparameter settings actually yielded a further improved score. Thus the improvement by CR relative to the baseline lacks hyperparameter sensitivity, and promisingly, there appears the potential for hyperparameter optimization on specific tasks to yield even better performance. In the main text, we report the optimized DreamerV2 w/ CR score ($\alpha = 0.5$, $\beta = 0.7$) even though all other results in the main text use a fixed ($\alpha = 0.7$, $\beta = 0.7$).

| **Method** | **Crafter Score** |
|---|---|
| DreamerV2 | $11.7 \pm 0.5$ |
| DreamerV2 w/ TD | $10.8 \pm 0.6$ |
| DreamerV2 w/ CR: $\alpha$=0.7, $\beta$=0.9, c=1e4 | $11.9 \pm 0.8$ |
| DreamerV2 w/ CR: $\alpha$=0.7, $\beta$=0.8, c=1e4 | $13.2 \pm 1.9$ |
| DreamerV2 w/ CR: $\alpha$=0.7, $\beta$=0.7, c=1e4 | $12.0 \pm 1.2$ |
| DreamerV2 w/ CR: $\alpha$=0.7, $\beta$=0.6, c=1e4 | $13.2 \pm 1.5$ |
| DreamerV2 w/ CR: $\alpha$=0.7, $\beta$=0.5, c=1e4 | $12.7 \pm 1.1$ |
| DreamerV2 w/ CR: $\alpha$=0.9, $\beta$=0.7, c=1e4 | $13.0 \pm 1.7$ |
| DreamerV2 w/ CR: $\alpha$=0.8, $\beta$=0.7, c=1e4 | $13.2 \pm 1.4$ |
| DreamerV2 w/ CR: $\alpha$=0.6, $\beta$=0.7, c=1e4 | $12.5 \pm 1.2$ |
| DreamerV2 w/ CR: $\alpha$=0.5, $\beta$=0.7, c=1e4 | $\mathbf{13.3 \pm 1.3}$ |
| DreamerV2 w/ CR: $\alpha$=0.7, $\beta$=0.7, c=1e3 | $12.6 \pm 2.0$ |

*Table A3.* Increased train ratio benefits Curious Replay (n=10, mean $\pm$ s.d.).

| **Method** | **Crafter Score** |
|---|---|
| DreamerV2 | $11.7 \pm 0.5$ |
| DreamerV2, 8x train frequency | $11.0 \pm 1.5$ |
| DreamerV2 w/ CR ($\alpha$=0.7, $\beta$=0.7) | $12.0 \pm 1.2$ |
| DreamerV2 w/ CR ($\alpha$=0.7, $\beta$=0.7), 8x train frequency | $\mathbf{15.7 \pm 2.4}$ |

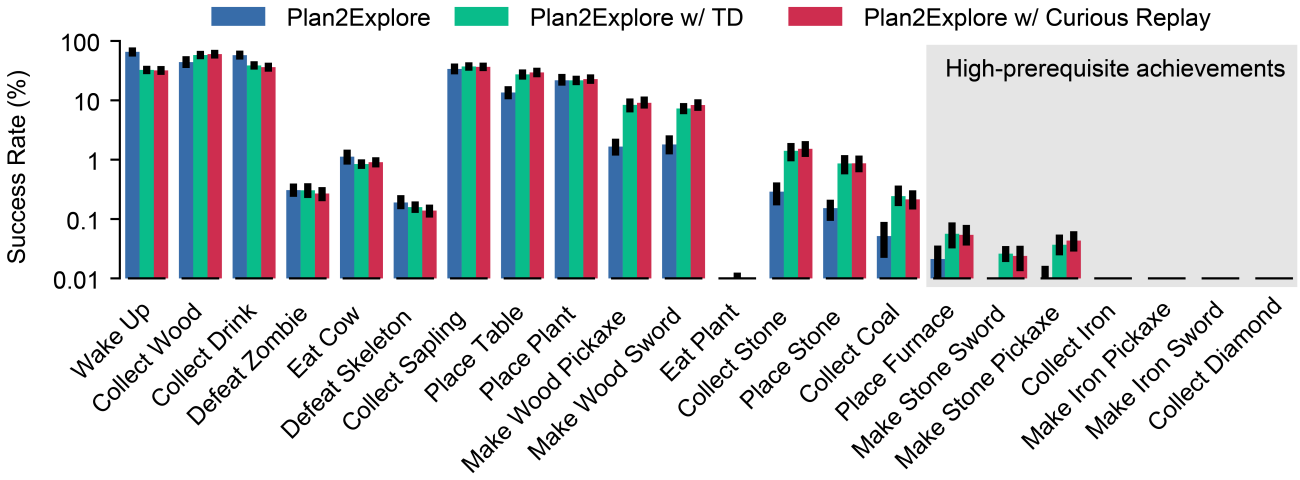


*Figure A13.* Agent ability spectrum showing success rates for unsupervised Crafter, ordered from left to right by number of prerequisites for an achievement (n=8 each condition, mean $\pm$ s.e.m.). Highlighted high-prerequisite achievements need at least four preceding achievements.

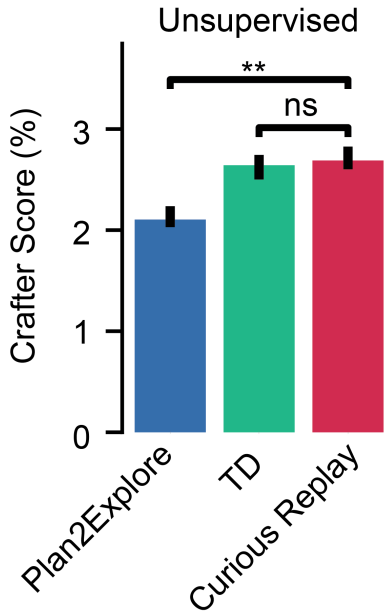


*Figure A14.* Score on the unsupervised (no extrinsic reward) version of Crafter (n=8 each condition, mean +/- s.e.m., independent t-test with fdr-bh correction).

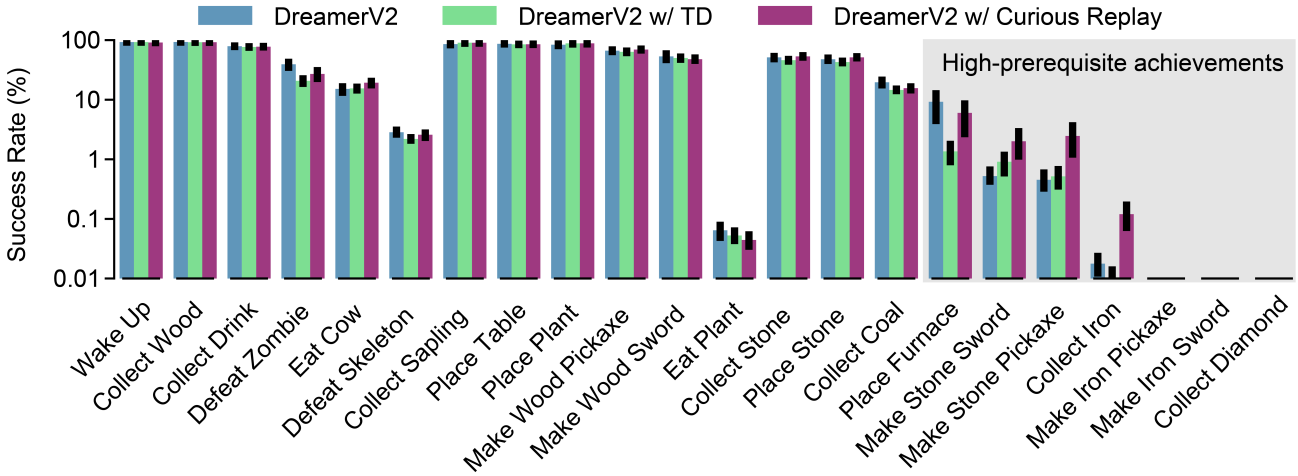


*Figure A15.* DreamerV2 agent ability spectrum showing the success rates for supervised Crafter, ordered from left to right by number of prerequisites for an achievement (n=8 each condition, mean $\pm$ s.e.m.). Curious Replay succeeds at challenging achievements that are unlocked only by many prerequisite. Highlighted high-prerequisite achievements need at least four preceding achievements.

*Table A4.* In Crafter, Curious Replay increases the sampling probability of the steps in the replay buffer where recently discovered resources are held by the agent. This is measured by calculating the median sampling probability for every step where a resource is held relative to the sampling probability in a uniform distribution. The probabilities used are from a saved checkpoint, chosen to be the next multiple of 100k steps after the 1% achievement threshold is reached (see Table A6). This threshold is intended to represent a small amount of experience with the resource but before the agent may be fully proficient in its use. (n=10 seeds for DreamerV2 w/ Curious Replay, n=5 seeds for DreamerV3 w/ Curious Replay).

| **Resource** | **Probability: DreamerV2 w/ CR** | **Checkpoint** | **Probability: DreamerV3 w/ CR** | **Checkpoint** |
|---|---|---|---|---|
| Wood | 1.2x | 100k | 1.0x | 100k |
| Wood Pickaxe | 1.7x | 100k | 1.1x | 100k |
| Stone | 1.9x | 100k | 1.2x | 100k |
| Stone Pickaxe | 2.6x | 300k | 1.1x | 300k |
| Iron | 3.1x | 900k | 1.2x | 800k |

*Table A5.* Curious Replay increases the number of times steps are trained on when the agent is holding a recently discovered resource. We assessed this for DreamerV3 w/ Curious Replay. This is measured by dividing 1) the number of times each data point with a given resource is trained on by 2) the estimated number of times that each data point's position in the replay buffer would be trained on with a uniform distribution. The checkpoint chosen for each resource is selected to be the nearest multiple of 10k environment steps after the first encounter with the resource. This checkpoint is chosen to give the most granular view of the impact of Curious Replay on new experiences. (n=5 seeds for DreamerV3 w/ Curious Replay).

| **Resource** | **Relative training count** |
|---|---|
| Wood | 1.1x |
| Wood Pickaxe | 1.9x |
| Stone | 5.5x |
| Stone Pickaxe | 4.6x |
| Iron | 6.2x |

*Table A6.* Steps to 1% achievement threshold in Crafter (20K step moving average) (n=10 seeds each, 4 for DreamerV3)

| **Achievement** | DreamerV2 | DreamerV2 w/ CR | DreamerV3 | DreamerV3 w/ CR |
|---|---|---|---|---|
| Collect Wood | 1K | 1.5K | 0.5K | 0.5K |
| Place Table | 7K | 8K | 4K | **3K** |
| Make Wood Pickaxe | 46K | 47K | 42K | **26K** |
| Collect Stone | 131K | 97K | 110K | **43K** |
| Make Stone Pickaxe | 533K | 236K | 745K | 244K |
| Collect Iron | never | 869K | 973K | **710K** |

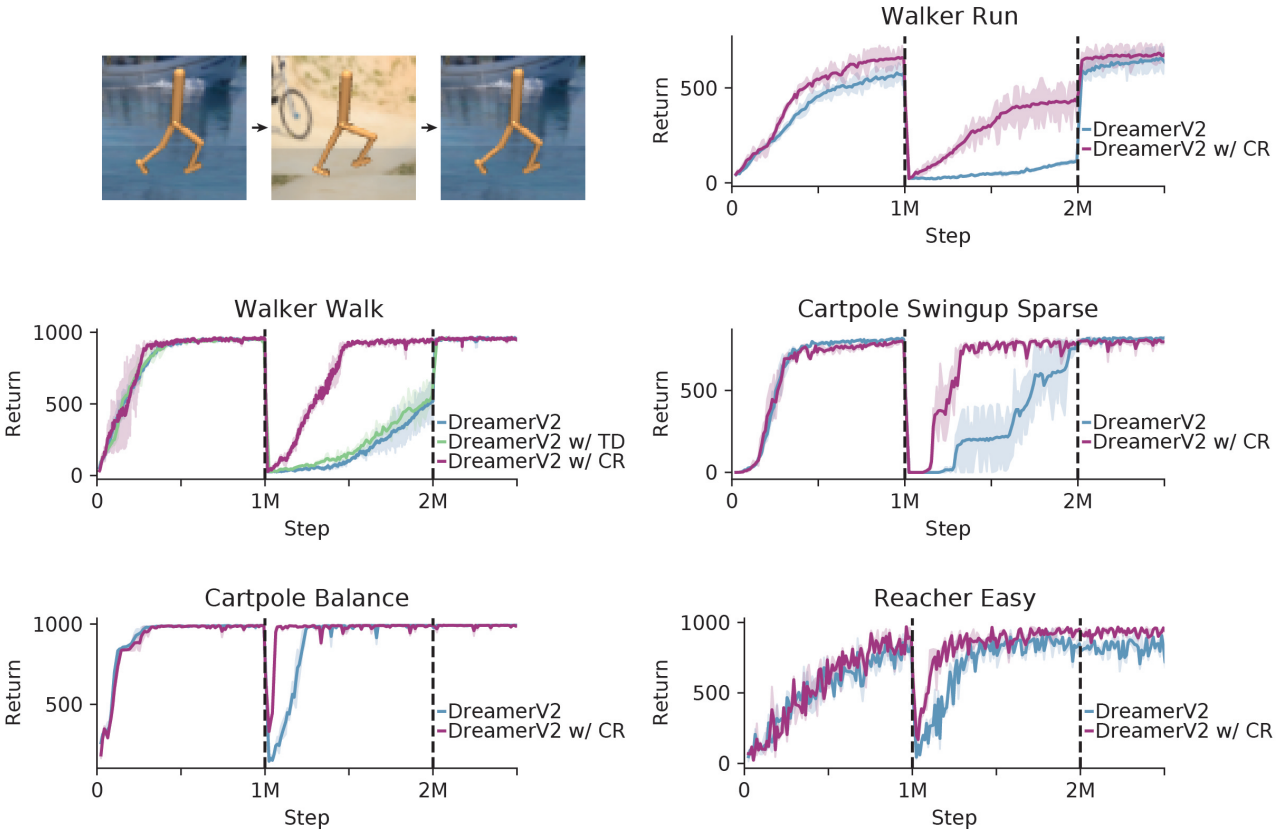


*Figure A16.* Background-Swap Control Suite, comparing DreamerV2 and DreamerV2 with Curious Replay. As schematized in the top left, the background image changes at 1M steps, and reverts at 2M steps. This tests adaptation at step 1M, and also assesses the presence (or, as we observe, absence) of catastrophic forgetting at step 2M (n=3 seeds, mean s.e.m.).

*Table A7.* Background-Swap Control Suite shows no evidence of catastrophic forgetting, as indicated by the similarly high score between DreamerV2 and DreamerV2 w/ CR at 2.05M steps, right after the background image has reverted to the original image. (CR = Curious Replay, $n$=3 seeds per method, mean $\pm$ s.e.m.)

| **Method** | **Walker Walk** | **Cartpole Swingup Sparse** | **Cartpole Balance** | **Walker Run** | **Reacher Easy** |
|---|---|---|---|---|---|
| t=50K steps | | | | | |
| DreamerV2 | $164.2 \pm 25.4$ | $2.9 \pm 1.0$ | $313.1 \pm 45.7$ | $62.5 \pm 3.8$ | $68.8 \pm 29.1$ |
| DreamerV2 w/ CR | $194.6 \pm 32.2$ | $1.1 \pm 1.1$ | $327.6 \pm 23.6$ | $71.8 \pm 17.8$ | $23.5 \pm 19.3$ |
| t=990K steps | | | | | |
| DreamerV2 | $954.6 \pm 4.0$ | $812.3 \pm 2.5$ | $989.9 \pm 1.1$ | $578.3 \pm 55.5$ | $849.9 \pm 92.0$ |
| DreamerV2 w/ CR | $956.3 \pm 6.5$ | $787.1 \pm 13.6$ | $991.8 \pm 0.4$ | $655.9 \pm 56.7$ | $881.2 \pm 59.4$ |
| t=2.05M steps | | | | | |
| DreamerV2 | $948.0 \pm 3.9$ | $817.6 \pm 0.8$ | $988.8 \pm 4.1$ | $588.8 \pm 52.2$ | $808.0 \pm 33.9$ |
| DreamerV2 w/ CR | $953.8 \pm 7.8$ | $804.3 \pm 16.9$ | $990.3 \pm 1.6$ | $657.4 \pm 59.7$ | $943.1 \pm 12.9$ |

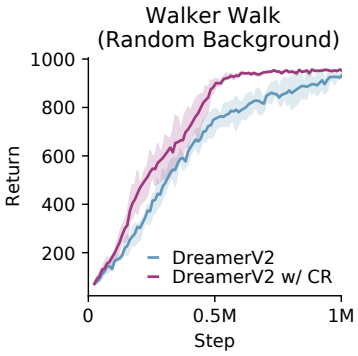


*Figure A17.* Walker Walk task from Distracting Deepmind Control Suite, with a background that changes to a different randomly selected image at each timestep, comparing DreamerV2 and DreamerV2 with Curious Replay (n=5 per task, mean $\pm$ s.e.m.).

*Table A8.* Deepmind Control Suite, (mean across n=3 seeds per task for DreamerV3 CR, and n=2 seeds each for DreamerV2 CR and DreamerV2). Scores $> 50$ more than their counterpart are bolded. $^{\dagger}$=Published results, see (Hafner et al., 2023).

| **Task** | **DreamerV3 w/ CR** | **DreamerV3**$^{\dagger}$ | **DreamerV2 w/ CR** | **DreamerV2** |
|---|---|---|---|---|
| Acrobot swingup | 172.9 | 210.0 | 36.3 | **321.5** |
| Cartpole balance | 996.4 | 996.4 | 990.9 | 994.3 |
| Cartpole balance sparse | 1000.0 | 1000.0 | 985.6 | 991.5 |
| Cartpole swingup | 858.5 | 819.1 | 801.6 | **871.0** |
| Cartpole swingup sparse | 346.3 | **792.9** | 328.7 | **723.2** |
| Cheetah run | **852.7** | 728.7 | 578.3 | **786.4** |
| Cup catch | 969.5 | 957.1 | 958.8 | 960.7 |
| Finger spin | 557.9 | **818.5** | **552.2** | 357.0 |
| Finger turn easy | **906.3** | 787.7 | 483.8 | **752.8** |
| Finger turn hard | 777.6 | 810.8 | 404.5 | **845.3** |
| Hopper hop | 385.1 | 369.6 | 161.8 | **253.5** |
| Hopper stand | 936.7 | 900.6 | 543.4 | **756.7** |
| Pendulum swingup | 566.4 | **806.3** | 743.5 | 784.8 |
| Quadruped run | **431.1** | 352.3 | 495.0 | **549.7** |
| Quadruped walk | **750.8** | 352.6 | **880.5** | 549.4 |
| Reacher easy | **956.5** | 898.9 | **977.1** | 915.9 |
| Reacher hard | 518.8 | 499.2 | 345.7 | **498.8** |
| Walker run | 772.8 | 757.8 | **693.3** | 526.7 |
| Walker stand | 978.5 | 976.7 | 977.0 | 971.4 |
| Walker walk | 961.7 | 955.8 | 933.9 | 904.1 |

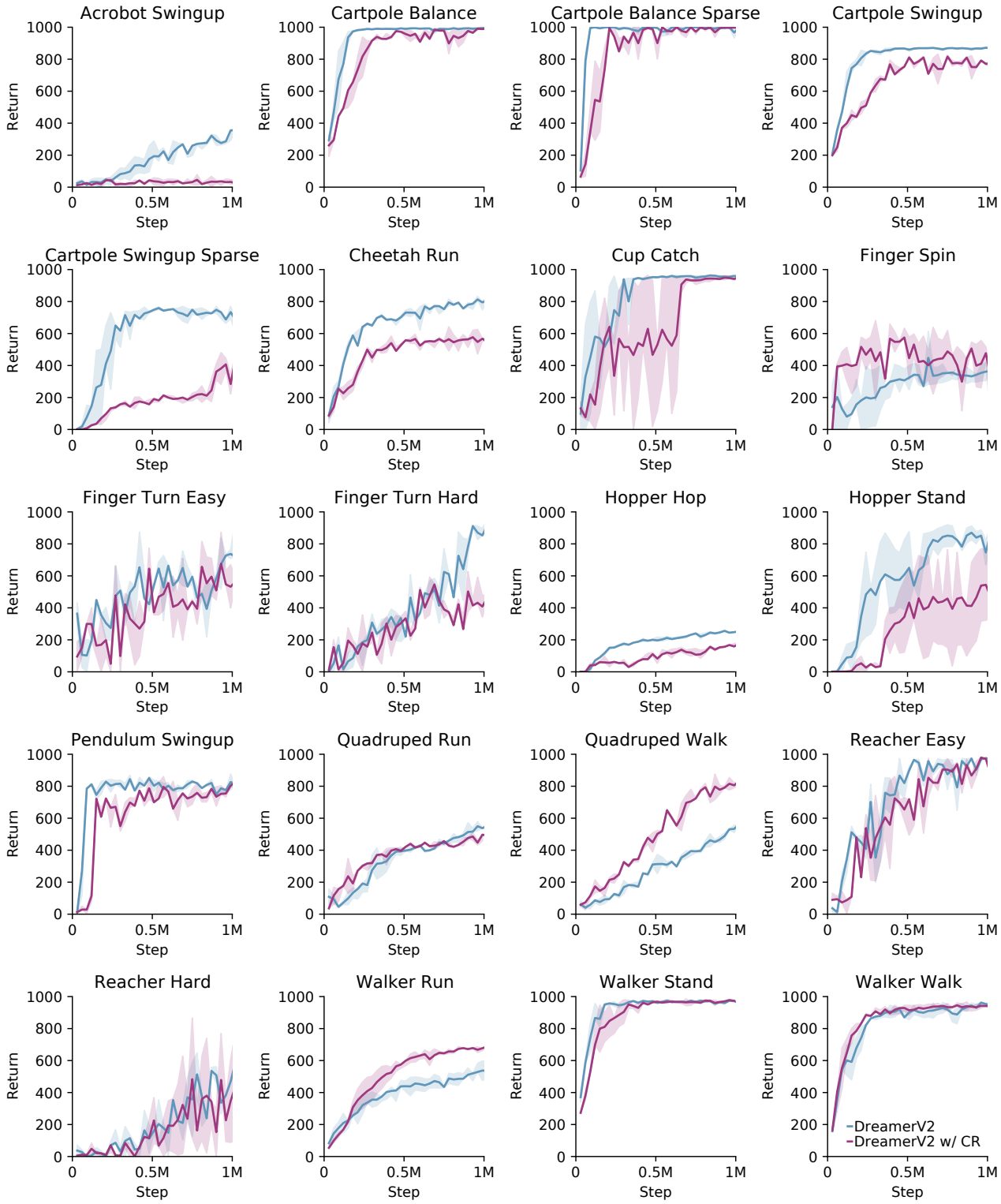


*Figure A18.* Deepmind Control Suite, comparing DreamerV2 and DreamerV2 with Curious Replay. Curious Replay improves performance on tasks such as Quadruped Walk and Walker Run, while decreasing performance on tasks such as Acrobot Swingup and Cartpole Swingup Sparse (n=2 per task, mean $\pm$ s.e.m.).

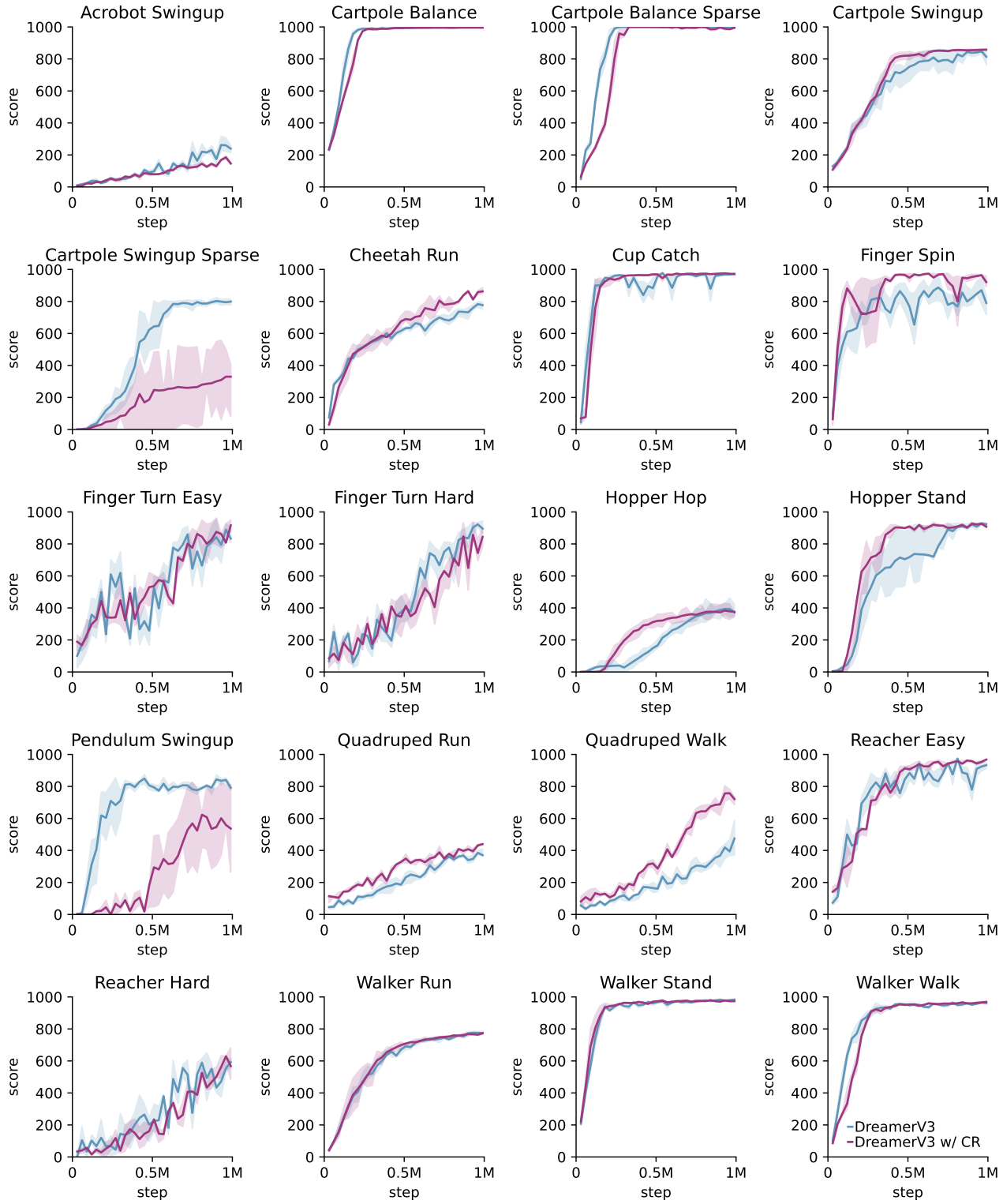


*Figure A19.* Deepmind Control Suite, comparing DreamerV3 and DreamerV3 with Curious Replay. Curious Replay improves performance on tasks such as Quadruped Walk, while decreasing performance on tasks such as Cartpole Swingup Sparse (n=3 per task, mean ± s.e.m.).