HarmonyDream: Task Harmonization Inside World Models

Haoyu Ma [*1] **Jialong Wu** [*1] **Ningya Feng** [1] **Chenjun Xiao** [2] **Dong Li** [2] **Jianye Hao** [2 3] **Jianmin Wang** [1] **Mingsheng Long** [1]

Abstract

Model-based reinforcement learning (MBRL) holds the promise of sample-efficient learning by utilizing a world model, which models how the environment works and typically encompasses components for two tasks: observation modeling and reward modeling. In this paper, through a dedicated empirical investigation, we gain a deeper understanding of the role each task plays in world models and uncover the overlooked potential of sample-efficient MBRL by mitigating the domination of either observation or reward modeling. Our key insight is that while prevalent approaches of explicit MBRL attempt to restore abundant details of the environment via observation models, it is difficult due to the environment's complexity and limited model capacity. On the other hand, reward models, while dominating implicit MBRL and adept at learning compact task-centric dynamics, are inadequate for sample-efficient learning without richer learning signals. Motivated by these insights and discoveries, we propose a simple yet effective approach, *HarmonyDream*, which automatically adjusts loss coefficients to maintain *task harmonization*, i.e. a dynamic equilibrium between the two tasks in world model learning. Our experiments show that the base MBRL method equipped with HarmonyDream gains 10%$-$69% absolute performance boosts on visual robotic tasks and sets a new state-of-the-art result on the Atari 100K benchmark. Code is available at https://github.com/thuml/HarmonyDream.

[*]Equal contribution [1]School of Software, BNRist, Tsinghua University. [2]Huawei Noah's Ark Lab. [3]College of Intelligence and Computing, Tianjin University.

Haoyu Ma <mhy22@mails.tsinghua.edu.cn>. Jialong Wu <wujialong0229@gmail.com>. Correspondence to: Mingsheng Long <mingsheng@tsinghua.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

1. Introduction

Learning efficiently to operate in environments with complex observations requires generalizing from past experiences. Model-based reinforcement learning (MBRL, Sutton (1990)) utilizing world models (Ha & Schmidhuber, 2018; LeCun, 2022) offers a promising approach. In MBRL, the agent learns behaviors by simulating trajectories based on world model predictions. These imaginary rollouts can reduce the need for real-environment interactions, thus improving the sample efficiency of model-based agents.

Concretely, world models are designed to learn and predict two key components of dynamics (formally defined in Sec. 2.1): how the environment transits and is observed (i.e. the *observation modeling* task) and how the task has been progressed (i.e. the *reward modeling* task) (Kaiser et al., 2020; Hafner et al., 2020; 2021). While observation transitions and rewards in low-dimensional spaces can be classically learned by separate models, for environments with high-dimensional and partial observations, it is favorable for world models to learn both tasks from a shared representation, a form of *multi-task learning*[1] (Caruana, 1997), aiming to improve learning efficiency and generalization performance (Jaderberg et al., 2017; Laskin et al., 2020; Yarats et al., 2021). However, to best exploit the benefits of multi-task learning, it demands careful designs to weigh the contribution of each task without allowing either one to dominate (Misra et al., 2016; Kendall et al., 2018), which naturally leads to the following question:

How do MBRL methods properly exploit the intrinsic multi-task benefits within world model learning?

In this work, we take a unified multi-task view to revisit world model learning in MBRL literature (Moerland et al., 2023): Prevalent *explicit* MBRL approaches (Kaiser et al., 2020; Hafner et al., 2021; Seo et al., 2022b), which is also our primary focus, aim to learn an exact duplicate of the environment by predicting each element (e.g., observations, rewards, and terminal signals), which gives the agent access to accurately learned transitions. However, learning to predict future observations can be difficult and inefficient since

[1]Here we refer to intrinsic multi-task learning inside world models rather than multi-task policy learning for different rewards.

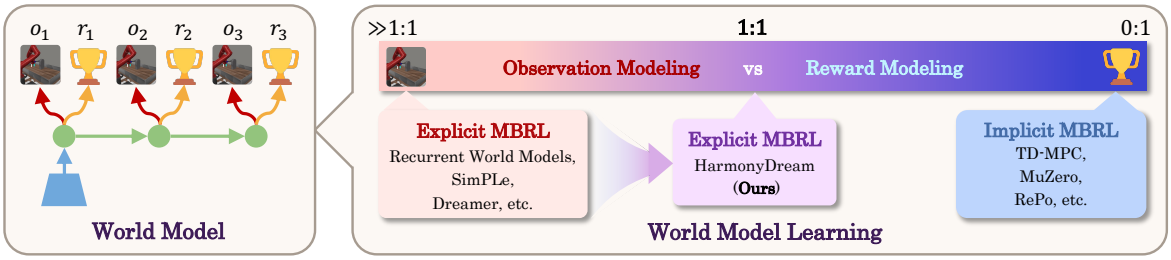


Figure 1. A multi-task view of world models. (*Left*) World models typically consist of components for two tasks: **observation modeling** and **reward modeling**. (*Right*) A spectrum of world model learning in MBRL. Explicit MBRL learns models dominated by observation modeling, while implicit MBRL relies solely on reward modeling. In the spirit of implicit MBRL, our proposed HarmonyDream enables explicit MBRL to maintain a dynamic equilibrium between them to unleash the multi-task benefits of world model learning, thus boosting the sample efficiency of MBRL.

it encourages the world model to capture everything in the environment, including task-irrelevant nuances (Okada & Taniguchi, 2021; Deng et al., 2022). Consequently, world model learning in explicit MBRL is typically dominated by observation modeling to capture complex observations and their associated dynamics but still suffers from model inaccuracies and compounding errors. This can be overcome by the spirit of *implicit* MBRL, which learns task-centric world models solely from reward modeling (Oh et al., 2017; Schrittwieser et al., 2020; Hansen et al., 2022) to realize the value equivalence principle, i.e., the predicted rewards along a trajectory of the world model matches that of the real environment (Grimm et al., 2020). This approach builds world models directly useful for MBRL to identify the optimal policy or value, and tends to perform better in tasks where the complete dynamics related to observations are too complicated to be perfectly modeled. Nevertheless, as the reward signals in RL are known to be sparser than signals in self-supervised learning, potentially leading to representation learning challenges, it is more practical to incorporate auxiliary tasks that provide richer learning signals beyond rewards (Jaderberg et al., 2017; Anand et al., 2022).

To support the above insights, we conduct a dedicated empirical investigation and reveal surprising deficiencies in sample efficiency within the default practice of a state-of-the-art model-based method (Dreamer, Hafner et al. (2020; 2021; 2023)). Notably, *increasing the coefficient of reward loss in world model learning leads to dramatically boosted sample efficiency* (see Sec. 2.3). Our analysis identifies the root cause as the domination of observation models in explicit world model learning: due to an overload of redundant observation signals, the model may establish spurious correlations in observations without realizing incorrect reward predictions, which ultimately hinders the learning process of the agent. On the other hand, a pure implicit version of Dreamer, which learns world models solely exploiting reward modeling, is also proven to be inefficient. In summary, domination of either task cannot properly exploit the multi-task benefits within world model learning.

As shown in Fig. 1, we propose to address the problem with **HarmonyDream**, a simple approach for explicit world model learning that exploits the advantages of both sides. By automatically adjusting loss coefficients through lightweight harmonizers, HarmonyDream seeks task harmonization inside world models, i.e., it maintains a dynamic equilibrium between reward and observation modeling during world model learning. We evaluate our approach on various challenging visual control domains, including Meta-world (Yu et al., 2020b), RLBench (James et al., 2020), distracted DMC variants (Grigsby & Qi, 2020; Zhang et al., 2018), the Atari 100K benchmark (Kaiser et al., 2020), and a challenging task from Minecraft (Fan et al., 2022), demonstrating consistent improvements in sample efficiency and generality to different base MBRL approaches (Deng et al., 2022).

The main contributions of this work are three-fold:

- To the best of our knowledge, our work, for the first time, systematically identifies the multi-task essence of world models and analyzes the deficiencies caused by the domination of a particular task, which is unexpectedly overlooked by most previous works.
- We propose HarmonyDream, a simple yet effective world model learning approach to mitigate the domination of either observation or reward modeling, without the need for exhaustive hyperparameter tuning.
- Our experiments show that HarmonyDream improves Dreamer with 10%−69% higher success rates or episode returns (up to 90% more success on Meta-world Assembly) in visual robotic tasks. Moreover, our method reaches a new state of the art, 136.5% mean human performance, on the Atari 100K benchmark.

2. A Multi-task Analysis in World Models

In this paper, we focus on vision-based RL tasks, formulated as partially observable Markov decision processes (POMDP). A POMDP is defined as a tuple

(a) Learning curves of different loss coefficients (b) Loss scales (c) State regression

Figure 2. Analysis experiments revealing the (b) imbalanced nature of world model learning and potential multi-task benefits yet to be properly exploited. Simply adjusting the coefficient of reward loss leads to (a) dramatically boosted sample efficiency of DreamerV2 agents and (c) better representations with lower environment state regression errors.

$(\mathcal{O}, \mathcal{A}, p, r, \gamma)$, where actions $a_t \sim \pi(a_t \mid o_{\leq t}, a_{<t})$ generated by the agent receive high-dimensional observations $o_t \sim p\left(o_t \mid o_{<t}, a_{<t}\right)$ and scalar rewards $r_t = r(o_{\leq t}, a_{<t})$ generated by the unknown transition dynamics p and reward function r. The goal of MBRL is to learn an agent that maximizes the γ-discounted cumulative rewards $\mathbb{E}_{p,\pi}\left[\sum_{t=1}^{T} \gamma^{t-1} r_t\right]$, leveraging a learned world model which approximates the underlying environment (p, r).

2.1. Two tasks in World Models

Two key tasks can be formally identified in world models, namely observation modeling and reward modeling.

Definition 2.1. The *observation modeling* task in world models is to predict consequent observations $p(o_{t+1:T} \mid o_{1:t}, a_{1:T})$ of a trajectory, given future actions. Similarly, the *reward modeling* task in world models is to predict future rewards $p(r_{t+1:T} \mid o_{1:t}, a_{1:T})$.

As mentioned before, these two tasks provide a unified view of MBRL: while explicit MBRL learns world models for both observations and rewards to mirror the complete dynamics of the environment, implicit MBRL only learns from reward modeling to capture task-centric dynamics.

2.2. Overview of World Model Learning

We conduct detailed analysis and build our method primarily upon DreamerV2[2] (Hafner et al., 2021), but we also demonstrate the generality of our method to various base MBRL algorithms, including DreamerV3 (Hafner et al., 2023) and DreamerPro (Deng et al., 2022) (see Sec. 4.4).

The world model in Dreamer (left in Fig. 1) consists of the following four components:

$$\begin{aligned} &\text{Representation model:} && z_t \sim q_\theta(z_t \mid z_{t-1}, a_{t-1}, o_t) \\ &\text{Observation model:} && \hat{o}_t \sim p_\theta(\hat{o}_t \mid z_t) \\ &\text{Transition model:} && \hat{z}_t \sim p_\theta(\hat{z}_t \mid z_{t-1}, a_{t-1}) \\ &\text{Reward model:} && \hat{r}_t \sim p_\theta\left(\hat{r}_t \mid z_t\right). \end{aligned} \tag{1}$$

The latent representation z_t is generated by the representation model using the previous latent state z_{t-1}, the current action a_{t-1}, and the current visual observation o_t. The latent prediction $\hat{z}_t$, meanwhile, is generated by the transition model using only the previous state and current action. All model parameters θ are trained to learn the observations, rewards, and transitions of the environment by jointly minimizing the following objectives:

$$\begin{aligned} &\text{Observation loss:} && \mathcal{L}_o(\theta) = -\log p_\theta(o_t \mid z_t) \\ &\text{Reward loss:} && \mathcal{L}_r(\theta) = -\log p_\theta(r_t \mid z_t) \\ &\text{Dynamics loss:} && \mathcal{L}_d(\theta) = \mathrm{KL}[q_\theta(z_t \mid z_{t-1}, a_{t-1}, o_t) \\ & && \qquad \| \, p_\theta(\hat{z}_t \mid z_{t-1}, a_{t-1})], \end{aligned} \tag{2}$$

where the dynamics loss simultaneously trains the latent predictions toward the representations, and regularizes the representations to be predictable. In practice, the observation model and reward model typically leverage Gaussian distributions, and both losses take the form of a simple L_2 loss between prediction $\hat{o}_t, \hat{r}_t$ and ground truth o_t, r_t respectively, excluding irrelevant constants.

Taking our multi-task view, the observation model and reward model with their associated losses account for the aforementioned two tasks in the world model of Dreamer. However, they do not operate in isolation and instead interact with and regularize each other upon the shared representation and transition model, in pursuit of either complete or task-centric latent dynamics, respectively.

The overall objective can be formulated as follows:

$$\mathcal{L}(\theta) = w_o \mathcal{L}_o(\theta) + w_r \mathcal{L}_r(\theta) + w_d \mathcal{L}_d(\theta). \tag{3}$$

By default, w_o, w_r, and w_d are typically set to approximately equal weights (namely, $w_o = w_r = w_d = 1.0$) (Hafner et al., 2020; 2021; Seo et al., 2022b; Wu et al.,

[2]When we started this work, DreamerV3 had not been released. A detailed discussion with DreamerV3 is included in later sections.

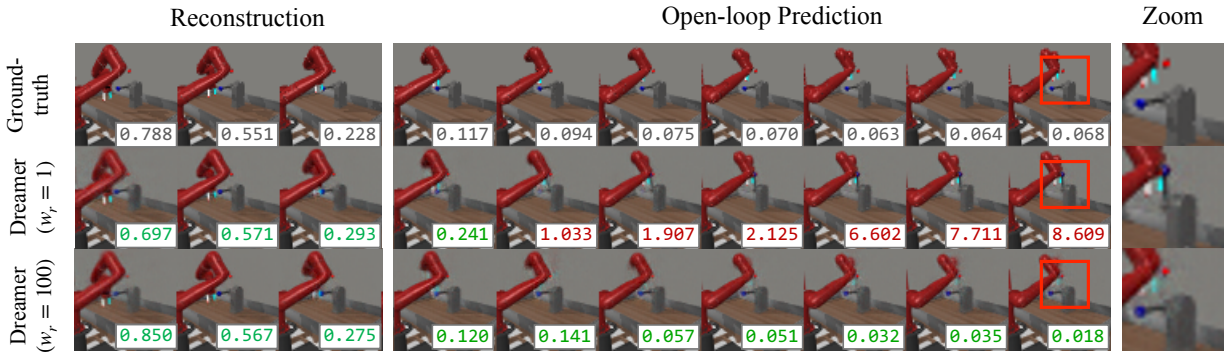


Figure 3. Analysis of world models learned with different reward loss coefficients. Rewards are labeled at the bottom right corner, with predictions marked as **correct** or **incorrect**. Dominating observation modeling in world models incurs spurious correlations between actions, observations, and rewards, which can be dissolved by properly emphasizing reward modeling.

2022), overlooking the potential domination of a particular task. In contrast, we conduct a careful empirical investigation to understand the role each task plays in world models and reveal the deficiency of the default weighting strategy.

2.3. Dive into World Model Learning

We consider the tasks of *pulling a lever up*, *pulling a handle up sideways*, and *hammering a screw on the wall*, from the Meta-world domain (Yu et al., 2020b), as our testbed to investigate world model learning. The prominent improvements of the derived approach in our benchmark experiments (see Sec. 4) prove that our discoveries can be generalized to various domains and tasks.

First of all, we experiment with simply adjusting the coefficient of the reward loss in Eq. (3). Results in Fig. 2a reveal a surprising fact that by simply tuning the reward loss weights ($w_r \in \{1, 10, 100\}$), the agent can achieve considerable improvements in terms of sample efficiency.

> **Finding 1.** *Leveraging the reward loss by adjusting its coefficient in world model learning has a great impact on the sample efficiency of model-based agents.*

One obvious reason for this is that the reward loss only accounts for a tiny proportion of the learning signals, actually a single scalar r_t. As shown in Fig. 2b, the scale of $\mathcal{L}_r$ is two orders of magnitude smaller than that of $\mathcal{L}_o$, which usually aggregates $H \times W \times C$ dimensions: $\log p_\theta(o_t \mid z_t) = \sum_{h,w,c} \log p_\theta(o_t^{(h,w,c)} \mid z_t)$. As discussed before, reward modeling is crucial for extracting task-relevant representations and driving behavior learning of the agents. Dominated by observation modeling, the world model fails to learn a task-centric latent space and predict accurate rewards, which hinders the agent learning process.

We then explore further to demonstrate how the observation modeling task dominating world models can specifically hurt behavior learning. To isolate distracting factors, we consider an offline setting (Levine et al., 2020). Concretely, we use a fixed replay buffer on the task of Lever Pull and offline train DreamerV2 agents with different reward loss coefficients on it (see details in Appendix C.4). In Fig. 3, we showcase a trajectory where the default Dreamer agent ($w_r = 1$) fails to lift the lever. It is evident that it learns a spurious correlation (Geirhos et al., 2020) between the actions of the robot and that of the lever and predicts inaccurate transitions and rewards, which misleads the agents to unfavorable behaviors. Properly balancing the reward loss ($w_r = 100$) can emphasize task-relevant information, such as whether the lever is actually lifted, to correct hallucinations by world models. Quantitative analysis in Fig. 2c measuring the ability of world models' representations to predict the ground truth states also suggests emphasizing reward modeling learns better task-centric representations.

> **Finding 2.** *Observation modeling as a dominating task can result in world models establishing spurious correlations without realizing incorrect reward predictions.*

Although we have shown above that exploiting reward modeling can bring benefits to world models and MBRL, learning world models depending solely on scarce reward signals, as implicit MBRL, has limited capability to learn meaningful representations, and thus can encounter optimization challenges and hinder sample-efficient learning (Yarats et al., 2021). Our experiment results in Fig. 2a show that a pure implicit version of DreamerV2 without the observation loss ($w_o = 0$) produces inferior results with a high variance.

> **Finding 3.** *Learning signal of world models from rewards alone without observations is inadequate for sample-efficient model-based learning.*

Discussion. We are not the first to adjust loss coefficients in world model learning, but we dedicatedly investigate this. Here we discuss the differences between our findings and previous literature. Our Finding 1 coincides with high re-

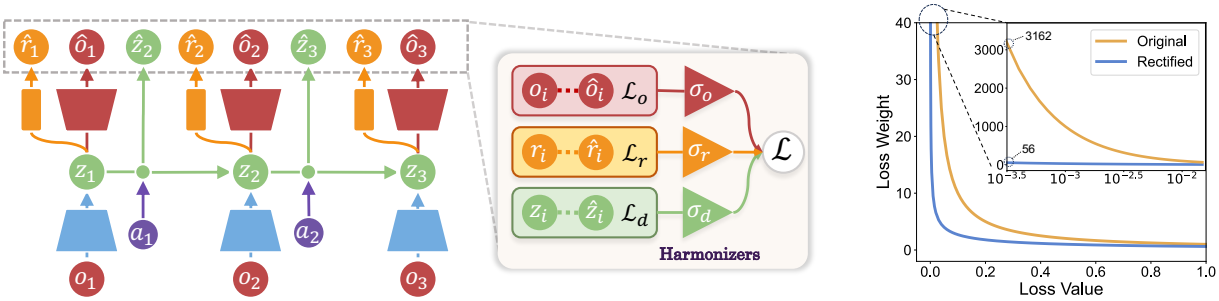


Figure 4. Overview of HarmonyDream. (*Left*) Built upon Dreamer, we introduce lightweight harmonizers to maintain a dynamic equilibrium between tasks. (*Right*) Comparison between the original harmonious loss (Eq. (4)) and the rectified one (Eq. (5)). The latter prevents an extremely large loss weight.

ward loss weights manually tuned (typically 100 or 1000) in decoder-free model-based RL (Nguyen et al., 2021; Deng et al., 2022). Our analysis differs from theirs in two significant ways: 1) We focus on a decoder-based world model, where the observations are learned from explicit reconstructions. 2) We discovered that emphasizing reward modeling is also beneficial for visually simple tasks (e.g. Meta-world tasks), in addition to visually demanding tasks with noisy backgrounds. Our Finding 3 is similar to the reward-only ablation in Dreamer (Hafner et al., 2020), but we prove that even if given higher loss weights, learning a world model purely from rewards is less sample-efficient than properly exploiting both observation and reward modeling.

3. HarmonyDream

In light of the discoveries and insights, we propose a simple yet effective method, HarmonyDream, as the first step towards exploiting the multi-task essence of explicit world model learning. Instead of task domination, we aim to dynamically maintain a harmonious interaction between the two tasks in world models: while observation modeling facilitates representation learning and prevents information loss, reward modeling enhances task-centric representations to inform behavior learning of the agents.

HarmonyDream mitigates the potential domination of a particular task in world models by introducing lightweight harmonizers, as shown in Fig. 4. Specifically, to maintain a dynamic equilibrium and avoid task domination, losses associated with different tasks are scaled to the same constant. A straightforward but suboptimal way is to set each loss weight to the reciprocal of the corresponding loss, i.e., $w_i = \mathrm{sg}(\frac{1}{\mathcal{L}_i}), i \in \{o, r, d\}$, where sg is a stop gradient function. Technically, as the loss is only calculated from a mini-batch of data and fluctuates throughout training, these weights are sensitive to outlier values and thus may further aggravate training instability. Instead, we adopt a variational method to learn the weights of different losses by the following *harmonious loss* for world model learning:

$$\begin{aligned}\mathcal{L}(\theta, \sigma_o, \sigma_r, \sigma_d) &= \sum_{i \in \{o,r,d\}} \mathcal{H}(\mathcal{L}_i(\theta), \sigma_i) \\ &= \sum_{i \in \{o,r,d\}} \frac{1}{\sigma_i}\mathcal{L}_i(\theta) + \log \sigma_i.\end{aligned} \tag{4}$$

The variational formulation $\mathcal{H}(\mathcal{L}_i(\theta), \sigma_i) = \sigma_i^{-1}\mathcal{L}_i(\theta) + \log \sigma_i$ serves as *harmonizers* to dynamically but smoothly rescale different losses, where the weight σ_i^{-1} with a learnable parameter $\sigma_i > 0$ approximates a "global" reciprocal of the loss scale, as stated in the following proposition:

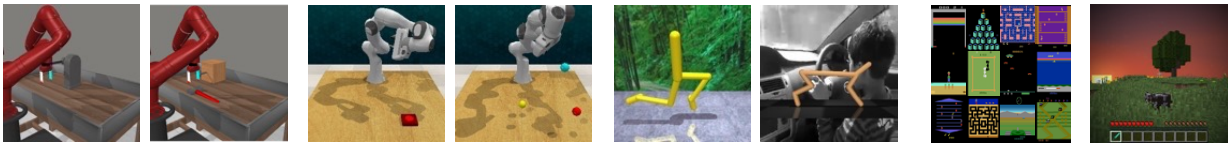
Proposition 3.1. *The optimal solution σ^* that minimizes the expected loss $\mathbb{E}[\mathcal{H}(\mathcal{L}, \sigma)]$, or equivalently $\nabla_\sigma \mathbb{E}[\mathcal{H}(\mathcal{L}, \sigma)] = 0$, is $\sigma^* = \mathbb{E}[\mathcal{L}]$. In other words, the harmonized loss scale is $\mathbb{E}[\mathcal{L}/\sigma^*] = 1$.*

In practice, σ_i is parameterized as $\sigma_i = \exp(s_i) > 0$, in order to optimize parameters s_i free of sign constraint. More essentially, we propose a rectification on Eq. (4), since a loss $\mathcal{L}$ with small values, such as the reward loss, can lead to extremely large coefficient $1/\sigma \approx \mathcal{L}^{-1} \gg 1$, which potentially hurt training stability. Specifically, we simply add a constant in regularization terms:

$$\begin{aligned}\mathcal{L}(\theta, \sigma_o, \sigma_r, \sigma_d) &= \sum_{i \in \{o,r,d\}} \hat{\mathcal{H}}(\mathcal{L}_i(\theta), \sigma_i) \\ &= \sum_{i \in \{o,r,d\}} \frac{1}{\sigma_i}\mathcal{L}_i(\theta) + \log(1 + \sigma_i).\end{aligned} \tag{5}$$

The harmonized loss scale by the *rectified harmonious loss* is equal to $\frac{2}{1+\sqrt{1+4/\mathbb{E}[\mathcal{L}]}} < 1$ (derived in Appendix B). We illustrate the corresponding loss weights learned with different loss scales in the right of Fig. 4, showing that the rectified loss effectively mitigates extremely large weights.

Discussion. Our harmonious loss is related in spirit to uncertainty weighting (Kendall et al., 2018) but has several key differences. Uncertainty weighting is derived from maximum likelihood estimation, which parameterizes noises



(a) Meta-world (b) RLBench (c) Distracted DMC variants (d) Atari (e) Minecraft

Figure 5. Visual control domains for evaluation, including robotic manipulation (a-b), distracted locomotion (c), and video games (d-e).

of Gaussian-distributed outputs of each task, known as homoscedastic uncertainty. In contrast, our motivation is to balance loss scales among tasks. More specifically, measuring the uncertainty of observations and rewards results in putting each observation pixel on equal footing as the scalar reward, still overlooking the large disparity in dimension sizes. However, we take high-dimensional observations as a whole and directly balance the two losses. Furthermore, we do not make assumptions on the distributions behind losses, which makes it possible for us to balance the KL loss, while uncertainty weighting has no theoretical basis for doing so.

4. Experiments

We evaluate the ability of HarmonyDream to boost sample efficiency of base MBRL methods on diverse and challenging visual control domains as shown in Figure 5, including robotic manipulation and locomotion, and video game tasks. We conduct most experiments for HarmonyDream based on DreamerV2 but also demonstrate its generality to other base MBRL methods, including DreamerV3 (Hafner et al., 2023) and DreamerPro (Deng et al., 2022). Experimental details and additional results can be found in Appendix C and E.

4.1. Meta-world Experiments

Environment details. Meta-world is a benchmark of 50 robotic manipulation tasks with fine-grained observation details, such as small target objects. Due to our limited computational resources, we choose a set of representative tasks according to the categories of task difficulty by Seo et al. (2022a): two from the *easy* category (Lever Pull and Handle Pull Side), two from the *medium* category (Hammer and Sweep Into), and two from the *hard* category (Push and Assembly). These tasks are run over different numbers of environment steps: *easy* tasks and Hammer over 250K steps, Sweep Into over 500K steps, the else over 1M steps.

Results. In Fig. 6a, we report the performance of HarmonyDream on six Meta-world tasks, in comparison with our base MBRL method DreamerV2. By simply adding harmonizers to the original DreamerV2 method, our HarmonyDream demonstrates superior performance in terms of both sample efficiency and final success rate. In particular, HarmonyDream achieves over 75% and 90% success rates on the challenging Push and Assembly tasks, respectively, while DreamerV2 fails to learn a meaningful policy.

4.2. RLBench Experiments

Environment details. To assess our method on more complex visual robotic manipulation tasks, we perform evaluations on the RLBench (James et al., 2020) domain. Most tasks in RLBench have high intrinsic difficulty and only offer sparse rewards. Learning these tasks requires expert demonstrations, dedicated network structure, and additional inputs (James & Davison, 2022; James et al., 2022), which is out of our scope. Therefore, following Seo et al. (2022a), we conduct experiments on two relatively easy tasks (Push Button and Reach Target) with dense rewards.

Results. In Fig. 6b, we show the superiority of our approach on the RLBench domain. HarmonyDream offers 28% of absolute final performance gain on the Push Button task and 50% on the more difficult Reach Target tasks. The results presented above prove the ability of HarmonyDream to promote sample efficiency of model-based RL on robotic manipulation domains for both easy and difficult tasks.

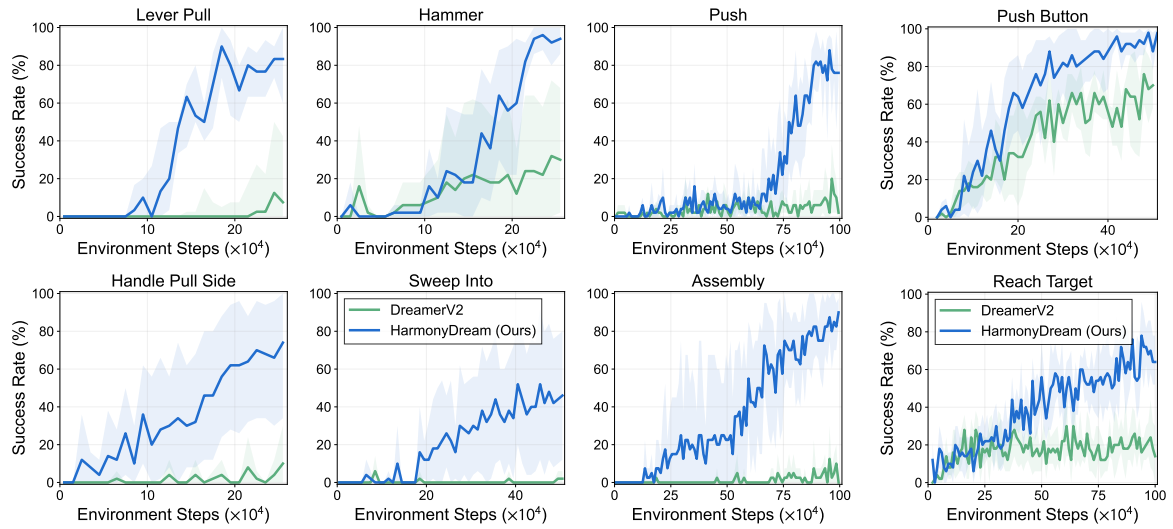
4.3. DMC Remastered Experiments

Environment details. DMC Remastered (Grigsby & Qi, 2020) is a challenging extension of the widely used robotic locomotion benchmark, DeepMind Control Suite (Tassa et al., 2018) with randomly generated graphics emphasizing visual diversity. We train and evaluate our agents on three tasks: Cheetah Run, Walker Run, and Cartpole Balance.

Results. Fig. 7a demonstrates the effectiveness of HarmonyDream on three DMCR tasks. Our method greatly enhances the base DreamerV2 method to unleash its potential. Fig. 7b shows different learning curves of the dynamics loss between HarmonyDream and DreamerV2. It is worth noting that DMCR tasks contain distracting visual factors, such as background and robot body color, which may hinder the learning process of observation modeling. DreamerV2 diverges in learning loss on this task, but by leveraging the importance of reward modeling, HarmonyDream bypasses distractors in observations and can learn task-centric transitions more easily, indicated by converged dynamics loss.

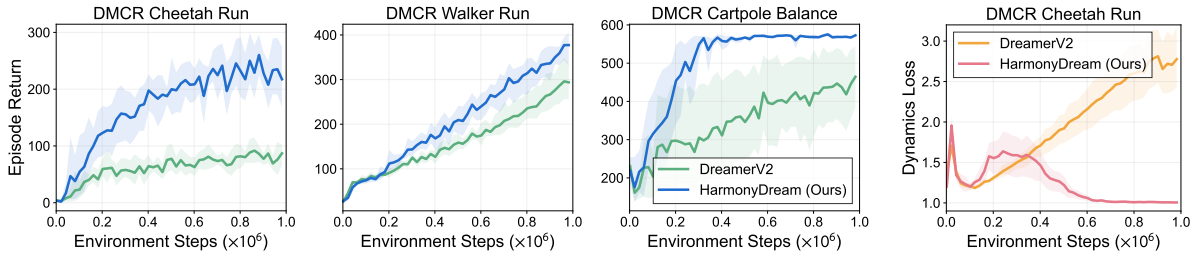
4.4. Generality to Model-based RL Methods

DreamerV3. DreamerV3 (Hafner et al., 2023) improves DreamerV2 to master diverse domains. Notably, our method is orthogonal to the various modifications in DreamerV3.



(a) Meta-world (b) RLBench

Figure 6. Learning curves on visual manipulation tasks from (a) Meta-World and (b) RLBench benchmarks, measured on the success rate. We report the mean and 95% confidence interval across five runs.



(a) Learning curves (b) Dynamics loss

Figure 7. Learning curves (a) on three DMC Remastered visual locomotion tasks and (b) one dynamics loss curve shown on Cheetah Run. We report the mean and 95% confidence interval across five runs.

DreamerV3 introduces a static symlog transformation to mitigate the problem of different per-dimension scales across environment domains, while HarmonyDream dynamically balances the overall loss scales across tasks in world model learning, considering together per-dimension scales, dimensions, and training dynamics. We refer to a detailed discussion in Appendix D.1. Experiments on Meta-world and RLBench, as shown in Fig. 8, illustrate that our method can combine with DreamerV3 to further improve performance. To further illustrate the applicability of our method, we also evaluate our Harmony DreamerV3 on two video game domains: Minecraft and Atari. For the Minecraft domain, we choose a challenging task of learning a basic skill, Hunt Cow, from the MineDojo benchmark (Fan et al., 2022). As shown in Fig. 9, Harmony DreamerV3 exhibits great improvement in the Minecraft domain. For the Atari 100K benchmark (Kaiser et al., 2020), we improve DreamerV3 to achieve 136.5% of mean human performance, setting a new state of the art among methods without lookahead search.

DreamerPro. DreamerPro (Deng et al., 2022) is a model-based RL method that "reconstructs" the cluster assignment of the observation. We conduct DreamerPro experiments on the DMCR domain. By default, DreamerPro uses a manually tuned reward loss weight $w_r = 1000$. We demonstrate in Fig. 8 that our method can still achieve higher sample efficiency and, on average, outperform manually tuned loss weights that are computationally costly.

4.5. Analysis

Comparison to implicit MBRL. As shown in Sec. 2.3, learning from reward modeling alone lacks sample efficiency. However, one may argue that purposefully designed implicit MBRL methods can be more effective. In Fig. 10a, we show comparisons with an implicit MBRL method, TD-MPC (Hansen et al., 2022) on three tasks of Meta-world. We observe that TD-MPC has difficulty in efficient learning as it lacks observation modeling to guide representation learning. In contrast, our method achieves superior performance. We

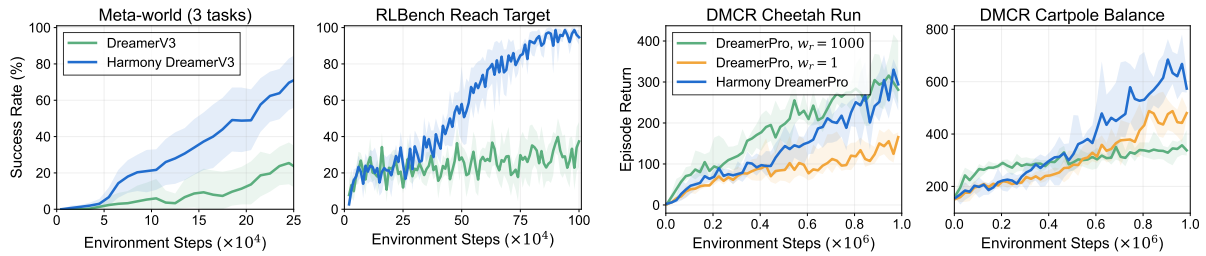


Figure 8. Performance of HarmonyDream applied to DreamerV3 (*left*) and DreamerPro (*right*).

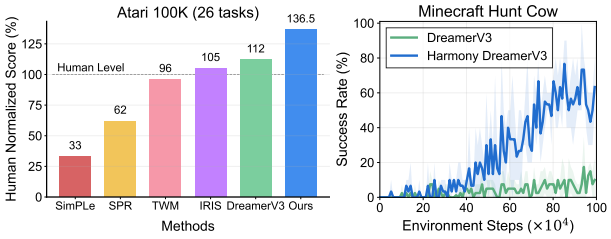


Figure 9. Performance of HarmonyDream based on DreamerV3 on the Atari 100K benchmark (*left*) and the Hunt Cow task from the MineDojo benchmark (*right*).

also compare with another implicit MBRL method, RePo (Zhu et al., 2023), as shown in the following paragraph.

Comparison to Dreamer-based task-centric methods. Denoised MDP (Wang et al., 2022) and RePo (Zhu et al., 2023) represent modifications to the Dreamer architecture that share a similar point with our approach in enhancing task-centric representations. We compare our method to these two methods on Meta-world, DMC Remastered, and additionally, natural background DMC (Zhang et al., 2018), which is also a distracted DMC variant used originally in the RePo paper. Fig. 11 shows that our HarmonyDream has a higher sample efficiency than Denoised MDP and RePo. Detailed discussion and comparison to these methods can be found in Appendix E.9 and E.10, respectively.

Comparison to multi-task learning methods. While our focus is not on developing a new multi-task learning method, we compare HarmonyDream with advanced methods in this area, including Uncertainty Weighting (UW, Kendall et al. (2018)), Dynamics Weight Average (DWA, Liu et al. (2019)), and NashMTL (Navon et al., 2022). Fig. 10b illustrates that our straightforward method is the most effective one among these methods, which also has the advantage of extreme simplicity. In-depth discussions on the differences between methods and why these methods can hardly make more improvements are included in Appendix D.2.

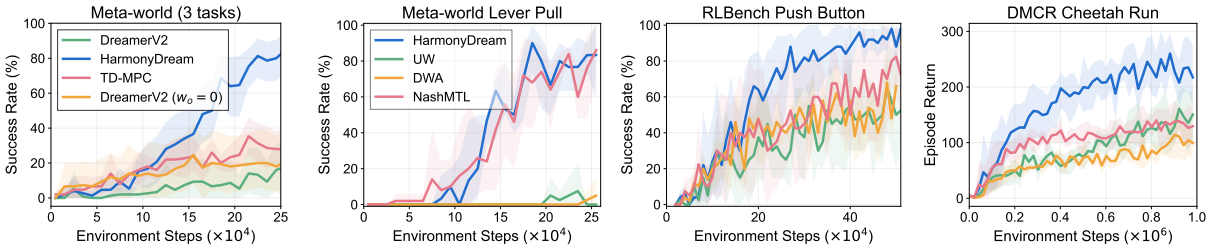
Ablation on rectified loss. We illustrate, through Fig. 17 in Appendix, the effectiveness of our rectified loss (Eq. (5)) in enhancing training stability and final performance.

5. Related Work

World models for visual RL. There exist several approaches to learning world models that explicitly model observations, transitions, and rewards. They can be widely utilized to boost sample efficiency in visual RL. In world models, visual representation can be learned via image reconstruction (Ha & Schmidhuber, 2018; Kaiser et al., 2020; Hafner et al., 2019; Seo et al., 2022a;b), or reconstruction-free contrastive learning (Okada & Taniguchi, 2021; Deng et al., 2022). Dreamer (Hafner et al., 2020; 2021; 2023) represents a series of methods that learn latent dynamics models from observations and learn behaviors by latent imagination. These methods have proven their effectiveness in tasks like video games (Hafner et al., 2021) and real robot control (Wu et al., 2022). Regardless, the problem of task domination is general for world models, and our findings and approach are not limited to our focused Dreamer architecture.

Implicit model-based RL. Implicit MBRL (Moerland et al., 2023) is a more abstract approach and aims to learn value equivalence models (Grimm et al., 2020) that focus on task-centric characteristics of the environment. This approach mitigates the objective mismatch (Lambert et al., 2020) between maximum likelihood estimation for world models and maximizing returns for policies. A typical success is MuZero (Schrittwieser et al., 2020; Ye et al., 2021), which learns a world model by predicting task-specific rewards, values, and policies, without observation reconstruction. Similarly, TD-MPC (Hansen et al., 2022) learns implicit world models for continuous control. While focusing on Dreamer, our analysis is consistent with those of MuZero showing that the potential efficiency of task-centric models can be better released when properly leveraging richer information from observation models (Anand et al., 2022).

Multi-task learning. Multi-task learning (Caruana, 1997; Ruder, 2017) aims to improve different tasks by jointly learning from a shared representation. A common approach is to aggregate task losses, where the loss or gradient of each task is manipulated by criteria like uncertainty (Kendall et al., 2018), performance metric (Guo et al., 2018), gradient norm (Chen et al., 2018) or gradient direction (Yu et al., 2020a; Wang et al., 2021; Navon et al., 2022), to avoid negative



(a) Comparison to TD-MPC (b) Comparison to multi-task learning methods applied to DreamerV2

Figure 10. Comparison of HarmonyDream to implicit MBRL methods and multi-task learning methods.

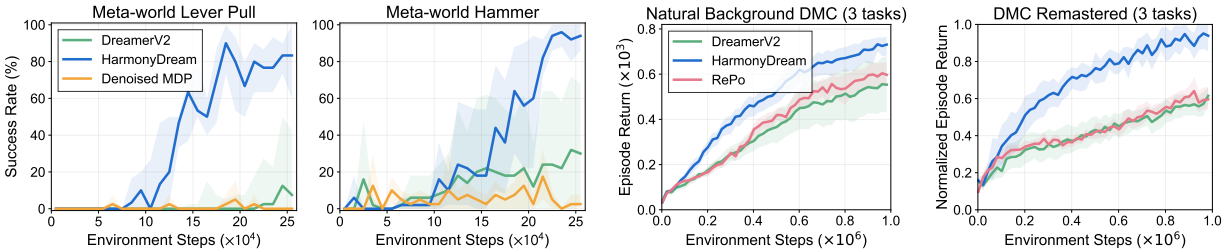


Figure 11. Comparison of HarmonyDream with Dreamer-based task-centric methods, Denoised MDP (*left*) and RePo (*right*).

transfer (Jiang et al., 2023). Previous works on multi-task learning in RL typically considered different policy learning tasks defined by different reward functions or environment dynamics (Rusu et al., 2016; Teh et al., 2017; Yu et al., 2020a). In contrast, we innovatively depict world model learning as multi-task learning, composed of reward and observation modeling, and HarmonyDream learns to maintain a delicate equilibrium between them to mitigate domination.

6. Discussion

We identify two tasks inside world models—observation and reward modeling—and interpret different MBRL methods as different task weighting. Our empirical study reveals that domination of a particular task can dramatically deteriorate the sample efficiency of MBRL. We thus introduce HarmonyDream, a simple world model learning approach that dynamically balances these tasks, thereby substantially improving sample efficiency.

HarmonyDream is particularly effective for scenarios where observation models are necessary for better representation learning, but the default weighting strategy of explicit world model learning causes negative effects due to observation modeling domination. These scenarios are mainly vision-based RL tasks, typically with complicated observations, including but not limited to:

- **Fine-grained task-relevant observations**: Robotics manipulation tasks (e.g., Meta-world and RLBench) and video games (e.g., Atari games, particularly Breakout, Qbert, and Gopher) require accurately modeling interactions with small objects.
- **Highly varied task-irrelevant observations**: Redundant visual components such as backgrounds (e.g., natural background DMC) and body color (e.g., DMCR) can easily distract visual agents if task-relevant information is not emphasized correctly.
- **Hybrid of both**: More difficult open-world tasks (e.g. Minecraft) can encounter both, including small target entities and abundant visual details.

These environment features are ubiquitous in realistic applications, and simply emphasizing reward modeling through HarmonyDream without any architecture modifications or hyperparameter tuning can make remarkable improvements.

Benchmark environments featuring clean observations with prominent target objects, such as standard DMC and Crafter (Hafner, 2022), do not encounter significant domination of observation modeling and are expected to gain marginal improvements with HarmonyDream, as shown in Fig. 16 (for DMC) and Fig. 22 (for Crafter) in Appendix. Nevertheless, we do not observe any negative performance change with HarmonyDream on these clean benchmarks.

The development of our method is primarily based on empirical and intuitive observations. A future direction is to explain our method theoretically, or to better measure and balance the contributions of world model tasks empirically, beyond simply considering loss scales. We hope our work can offer valuable insights and help pave the way for exploiting the multi-task nature of world models.

Acknowledgements

We would like to thank many colleagues, in particular Haixu Wu, Baixu Chen, Yuhong Yang, Chaoyi Deng, and Jincheng Zhong, who provided us with valuable discussions. This work was supported by the National Natural Science Foundation of China (62022050 and U2342217), the BNRist Innovation Fund (BNR2024RC01010), the Huawei Innovation Fund, and the National Engineering Research Center for Big Data Software.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Anand, A., Walker, J., Li, Y., Vértes, E., Schrittwieser, J., Ozair, S., Weber, T., and Hamrick, J. B. Procedural generalization by planning with self-supervised world models. In *ICLR*, 2022.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2021.

Caruana, R. Multitask learning. *Machine learning*, 28: 41–75, 1997.

Chen, Z., Badrinarayanan, V., Lee, C.-Y., and Rabinovich, A. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, 2018.

Deng, F., Jang, I., and Ahn, S. Dreamerpro: Reconstruction-free model-based reinforcement learning with prototypical representations. In *ICML*, 2022.

Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.-A., Zhu, Y., and Anandkumar, A. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *NeurIPS*, 2022.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Grigsby, J. and Qi, Y. Measuring visual generalization in continuous control from pixels. *arXiv preprint arXiv:2010.06740*, 2020.

Grimm, C., Barreto, A., Singh, S., and Silver, D. The value equivalence principle for model-based reinforcement learning. In *NeurIPS*, 2020.

Guo, M., Haque, A., Huang, D.-A., Yeung, S., and Fei-Fei, L. Dynamic task prioritization for multitask learning. In *ECCV*, 2018.

Ha, D. and Schmidhuber, J. Recurrent world models facilitate policy evolution. In *NeurIPS*, 2018.

Hafner, D. Benchmarking the spectrum of agent capabilities. In *ICLR*, 2022.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *ICML*, 2019.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *ICLR*, 2020.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering atari with discrete world models. In *ICLR*, 2021.

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

Hansen, N., Wang, X., and Su, H. Temporal difference learning for model predictive control. In *ICML*, 2022.

Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. Reinforcement learning with unsupervised auxiliary tasks. In *ICLR*, 2017.

James, S. and Davison, A. J. Q-attention: Enabling efficient learning for vision-based robotic manipulation. *IEEE Robotics and Automation Letters*, 2022.

James, S., Ma, Z., Arrojo, D. R., and Davison, A. J. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

James, S., Wada, K., Laidlow, T., and Davison, A. J. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. In *CVPR*, 2022.

Jiang, J., Chen, B., Pan, J., Wang, X., Dapeng, L., Jiang, J., and Long, M. Forkmerge: Overcoming negative transfer in multi-task learning. In *NeurIPS*, 2023.

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. In *ICLR*, 2020.

Kendall, A., Gal, Y., and Cipolla, R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018.

Lambert, N., Amos, B., Yadan, O., and Calandra, R. Objective mismatch in model-based reinforcement learning. In *L4DC*, 2020.

Laskin, M., Srinivas, A., and Abbeel, P. Curl: Contrastive unsupervised representations for reinforcement learning. In *ICML*, 2020.

LeCun, Y. A path towards autonomous machine intelligence. *preprint posted on openreview*, 2022.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Liu, B., Liu, X., Jin, X., Stone, P., and Liu, Q. Conflict-averse gradient descent for multi-task learning. In *NeurIPS*, 2021.

Liu, S., Johns, E., and Davison, A. J. End-to-end multi-task learning with attention. In *CVPR*, 2019.

Micheli, V., Alonso, E., and Fleuret, F. Transformers are sample efficient world models. In *ICLR*, 2023.

Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., and Wu, H. Mixed precision training. In *ICLR*, 2018.

Misra, I., Shrivastava, A., Gupta, A., and Hebert, M. Cross-stitch networks for multi-task learning. In *CVPR*, 2016.

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

Navon, A., Shamsian, A., Achituve, I., Maron, H., Kawaguchi, K., Chechik, G., and Fetaya, E. Multi-task learning as a bargaining game. In *ICML*, 2022.

Nguyen, T., Shu, R., Pham, T., Bui, H., and Ermon, S. Temporal predictive coding for model-based planning in latent space. In *ICML*, 2021.

Oh, J., Singh, S., and Lee, H. Value prediction network. In *NeurIPS*, 2017.

Okada, M. and Taniguchi, T. Dreaming: Model-based reinforcement learning by latent imagination without reconstruction. In *ICRA*, 2021.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Robine, J., Höftmann, M., Uelwer, T., and Harmeling, S. Transformer-based world models are happy with 100k interactions. In *ICLR*, 2023.

Ruder, S. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

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.

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.

Seo, Y., Hafner, D., Liu, H., Liu, F., James, S., Lee, K., and Abbeel, P. Masked world models for visual control. In *CoRL*, 2022a.

Seo, Y., Lee, K., James, S. L., and Abbeel, P. Reinforcement learning with action-free pre-training from videos. In *ICML*, 2022b.

Sutton, R. S. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pp. 216–224. Elsevier, 1990.

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.

Teh, Y., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., Heess, N., and Pascanu, R. Distral: Robust multitask reinforcement learning. In *NeurIPS*, 2017.

Wang, T., Du, S. S., Torralba, A., Isola, P., Zhang, A., and Tian, Y. Denoised mdps: Learning world models better than the world itself. In *ICML*, 2022.

Wang, Z., Tsvetkov, Y., Firat, O., and Cao, Y. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *ICLR*, 2021.

Wu, J., Ma, H., Deng, C., and Long, M. Pre-training contextualized world models with in-the-wild videos for reinforcement learning. In *NeurIPS*, 2023.

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

Yarats, D., Zhang, A., Kostrikov, I., Amos, B., Pineau, J., and Fergus, R. Improving sample efficiency in model-free reinforcement learning from images. In *AAAI*, 2021.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *ICLR*, 2022.

Ye, W., Liu, S., Kurutach, T., Abbeel, P., and Gao, Y. Mastering atari games with limited data. In *NeurIPS*, 2021.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. In *NeurIPS*, 2020a.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *CoRL*, 2020b.

Zhang, A., Wu, Y., and Pineau, J. Natural environment benchmarks for reinforcement learning. *arXiv preprint arXiv:1811.06032*, 2018.

Zhou, B., Li, K., Jiang, J., and Lu, Z. Learning from visual observation via offline pretrained state-to-go transformer. In *NeurIPS*, 2023.

Zhu, C., Simchowitz, M., Gadipudi, S., and Gupta, A. Repo: Resilient model-based reinforcement learning by regularizing posterior predictability. In *NeurIPS*, 2023.

A. Behavior Learning

HarmonyDream does not change the behavior learning procedure of its base MBRL methods (Hafner et al., 2021; 2023; Deng et al., 2022), and we briefly describe the actor-critic learning scheme shared with these base methods.

Specifically, we leverage a stochastic actor and a deterministic critic parameterized by ψ and ξ, respectively, as shown below:

$$\text{Actor: } \hat{a}_t \sim \pi_\psi\left(\hat{a}_t \mid \hat{z}_t\right) \qquad \text{Critic: } v_\xi\left(\hat{z}_t\right) \approx \mathrm{E}_{p_\theta,\pi_\psi}\left[\sum\nolimits_{\tau\geq t}\gamma^{\tau-t}\hat{r}_\tau\right], \tag{6}$$

where p_θ is the world model. The actor and critic are jointly trained on the same imagined trajectories $\{\hat{z}_\tau, \hat{a}_\tau, \hat{r}_\tau\}$ with horizon H, generated by the transition model and reward model in Eq. (1) and the actor in Eq. (6). The critic is trained to regress the λ-target:

$$\mathcal{L}_{\text{critic}}(\xi) \doteq \mathbb{E}_{p_\theta,\pi_\psi}\left[\sum_{\tau=t}^{t+H}\frac{1}{2}\left(v_\xi(\hat{z}_\tau) - \text{sg}(V_\tau^\lambda)\right)^2\right], \tag{7}$$

$$V_\tau^\lambda \doteq \hat{r}_\tau + \gamma\begin{cases}(1-\lambda)v_\xi(\hat{z}_{\tau+1}) + \lambda V_{\tau+1}^\lambda & \text{if } \tau < t+H\\ v_\xi(\hat{z}_{\tau+1}) & \text{if } \tau = t+H.\end{cases} \tag{8}$$

The actor, meanwhile, is trained to output actions that maximize the critic output by backpropagating value gradients through the learned world model. The actor loss is defined as follows:

$$\mathcal{L}_{\text{actor}}(\psi) \doteq \mathbb{E}_{p_\theta,\pi_\psi}\left[\sum_{\tau=t}^{t+H}\left(-V_\tau^\lambda - \eta\,\mathrm{H}\left[\pi_\psi(\hat{a}_\tau|\hat{z}_\tau)\right]\right)\right], \tag{9}$$

where $\mathrm{H}\left[\pi_\psi(\hat{a}_\tau|\hat{z}_\tau)\right]$ is an entropy regularization which encourages exploration, and η is the hyperparameter that adjusts the regularization strength. For more details, we refer to Hafner et al. (2020).

B. Derivations

Proof of Proposition 3.1. To minimize $\mathbb{E}[\mathcal{H}(\mathcal{L},\sigma)]$, we force the the partial derivative w.r.t. σ to 0:

$$\nabla_\sigma\mathbb{E}[\mathcal{H}(\mathcal{L},\sigma)] = \nabla_\sigma\mathbb{E}\left[\frac{1}{\sigma}\mathcal{L} + \log\sigma\right] = \mathbb{E}\left[\nabla_\sigma\left(\frac{1}{\sigma}\mathcal{L} + \log\sigma\right)\right] \tag{10}$$

$$= \mathbb{E}\left[-\frac{1}{\sigma^2}\mathcal{L} + \frac{1}{\sigma}\right] = \frac{1}{\sigma} - \frac{1}{\sigma^2}\mathbb{E}[\mathcal{L}] = 0. \tag{11}$$

This results in the solution $\sigma^* = \mathbb{E}[\mathcal{L}]$, and equivalently, the harmonized loss scale is $\mathbb{E}[\mathcal{L}/\sigma^*] = 1$.

Analytic solution of rectified loss. Similarly, minimizing $\mathbb{E}\left[\hat{\mathcal{H}}(\mathcal{L},\sigma)\right]$ yields

$$\begin{aligned}\nabla_\sigma\mathbb{E}\left[\hat{\mathcal{H}}(\mathcal{L},\sigma)\right] = \nabla_\sigma\left(\frac{1}{\sigma}\mathbb{E}[\mathcal{L}] + \log\left(1+\sigma\right)\right) &= -\frac{1}{\sigma^2}\mathbb{E}[\mathcal{L}] + \frac{1}{1+\sigma} = 0\\ \sigma &= \frac{\mathbb{E}[\mathcal{L}] + \sqrt{\mathbb{E}[\mathcal{L}]^2 + 4\mathbb{E}[\mathcal{L}]}}{2}.\end{aligned} \tag{12}$$

Therefore the learnable loss weight, in our rectified harmonious loss, approximates the analytic loss weight:

$$\frac{1}{\sigma} = \frac{2}{\mathbb{E}[\mathcal{L}] + \sqrt{\mathbb{E}[\mathcal{L}]^2 + 4\mathbb{E}[\mathcal{L}]}}, \tag{13}$$

corresponding to a loss scale $\mathbb{E}[\mathcal{L}]$, which is less than the unrectified $1/\mathbb{E}[\mathcal{L}]$. Adding a constant in the regularization term $\log(1+\sigma)$ results in the $4\mathbb{E}[\mathcal{L}]$ in the $\sqrt{\mathbb{E}[\mathcal{L}]^2 + 4\mathbb{E}[\mathcal{L}]}$ term, which prevents the loss weight from getting extremely large when faced with a small $\mathbb{E}[\mathcal{L}]$.

C. Experimental Details

C.1. Benchmark Environments

Meta-world. Meta-world (Yu et al., 2020b) is a benchmark of 50 distinct robotic manipulation tasks. We choose six tasks in all according to the difficulty criterion *(easy, medium, hard, and very hard)* proposed by Seo et al. (2022a). Specifically, we choose Handle Pull Side and Lever Pull from the *easy* category, Hammer and Sweep Into from the *medium* category, and Push and Assembly from the *hard* category. We observe that although the Hammer task belongs to the *medium* category, it is relatively easy for the DreamerV2 agent to learn, and our HarmonyDream can already achieve high success with 250K environment steps. Therefore, we train our agents over 250K environment steps on Hammer, along with the two *easy* tasks. For the remaining tasks, we train our agents over 500K environment steps for Sweep Into, and 1M environment steps for Push and Assembly, according to their various difficulties. In all tasks, the episode length is 500 environment steps with no action repeat.

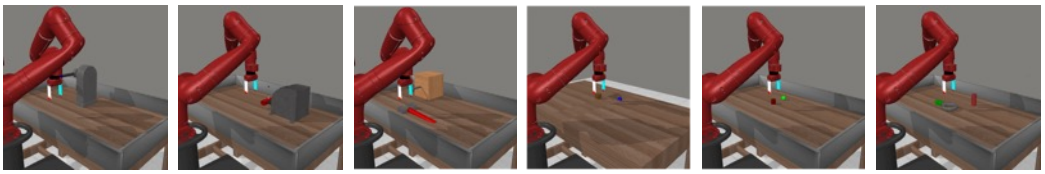


Figure 12. Example observations of Meta-world tasks: Lever Pull, Handle Pull Side, Hammer, Sweep Into, Push, and Assembly (left to right).

RLBench. RLBench (James et al., 2020) is a challenging benchmark for robot learning. Most tasks in RLBench are overchallenging for DreamerV2, even equipped with HarmonyDream. Therefore, following Seo et al. (2022a), we choose two relatively easy tasks (i.e. Push Button, Reach Target) and use an action mode that specifies the delta of joint positions. Because the original RLBench benchmark does not provide dense rewards for the Push Button task, we assign a dense reward following Seo et al. (2022a), which is defined as the sum of the L2 distance of the gripper to the button and the magnitude of the button being pushed. In our experiments, we found that the original convolutional encoder and decoder of DreamerV2 can be insufficient for learning the RLBench task. Therefore, in this domain, we adopt the ResNet-style encoder and decoder from Wu et al. (2023) for both DreamerV2 and our HarmonyDream. Note here that changes in the encoder and decoder architecture are completely orthogonal to our method and contributions. For tasks in the RLBench domain, the maximum episode length is set to 400 environment steps with an action repeat of 2.

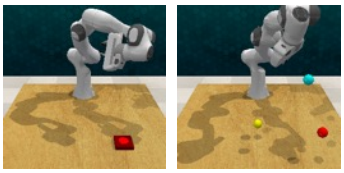


Figure 13. Example observations of RLBench tasks: Push Button and Reach Target.

DMC Remastered. The DMC Remastered (DMCR) (Grigsby & Qi, 2020) benchmark is a challenging extension of the widely used robotic locomotion benchmark, DeepMind Control Suite (Tassa et al., 2018), by expanding a complicated graphical variety. On initialization of each episode for both training and evaluation, the DMCR environment randomly resets 7 factors affecting visual conditions, including floor texture, background, robot body color, target color, reflectance, camera position, and lighting. Our agents are trained and evaluated on three tasks: Cheetah Run, Walker Run, and Cartpole Balance. We use all variation factors in all of our experiments and train our agents over 1M environment steps. Following the common setup of DeepMind Control Suite (Hafner et al., 2020; Yarats et al., 2022), we set the episode length to 1000 environment steps with an action repeat of 2.

Figure 14. Example observations of DMC Remastered tasks: Cheetah Run, Walker Run, and Cartpole Balance.

Atari 100K Benchmark. The Atari 100K benchmark contains 26 video games with up to 18 discrete actions. On this benchmark, the agent is

allowed to interact with each game environment for 100K steps, equivalent to 400K frames due to a frameskip of 4. This number of interaction steps, roughly two hours of real-time gameplay, has become a widely adopted standard in the realm of sample-efficient reinforcement learning. Human players are evaluated after two hours to get familiar with the game. Following the established protocol, we report the raw performance for each game, and the mean and median of human normalized scores: $(\text{score}_{\text{agent}} - \text{score}_{\text{random}}) \,/\, (\text{score}_{\text{human}} - \text{score}_{\text{random}})$. For this benchmark, we keep all implementation details the same as DreamerV3.

Natural Background DMC. Natural background DMC (Zhang et al., 2018) modifies the DeepMind Control Suite by substituting its static background with natural videos. In our paper, this environment is implemented using the RePo (Zhu et al., 2023) codebase[3]. Following RePo, we train and evaluate our agent on three tasks: Cheetah Run, Walker Run and Cartpole Swingup. We adopt the standard configuration of DMC for natural background DMC, with a maximum episode length of 1000 environment steps and an action repeat of 2.

Minecraft. Minecraft is a popular open-world game where a player explores a procedurally generated 3D world with diverse types of terrains to roam, materials to mine, tools to craft, structures to build, and wonders to discover. We leverage MineDojo (Fan et al., 2022), an massive simulation suite developed on Minecraft, encompassing over 3000 distinct tasks. Our focus was to master a fundamental skill, *Hunt Cow*, utilizing the manual dense reward provided by MineDojo. We prune the action space of MineDojo to Table 1, following the practice of STG-Transformer (Zhou et al., 2023). For this benchmark, we employ the *Large* model size variant of DreamerV3, comprising approximately 77M parameters. To ensure the terrain diversity of the environment, we hard reset the environment to generate a new world every 5 episodes. Observations for our agents consist solely of RGB frames, with a resolution of $128 \times 128 \times 3$ pixels. The maximum episode length is 500 environment steps, with no action repeat.

Table 1. Pruned Action Space of the MineDojo Environment

Index	Descriptions	Num of Actions
0	Forward and backward	3
1	Move left and right	3
2	Jump, sneak, and sprint	4
3	Camera delta pitch/yaw ($\pm 15^\circ$ for each action)	5
4	Use and Attack	3

C.2. Base MBRL Methods

DreamerV2. Unless otherwise specified, *HarmonyDream (Ours)* in the experiment section refers to the HarmonyDream method based on DreamerV2 (Hafner et al., 2021). Details about DreamerV2 have been elaborated on in the main text, and we refer readers to Sec. 2.2 and Hafner et al. (2020; 2021).

DreamerV3. DreamerV3 (Hafner et al., 2023) is a general and scalable algorithm that builds upon DreamerV2. In order to master a wide range of domains with fixed hyperparameters, DreamerV3 made many changes to DreamerV2, including using symlog predictions, utilizing world model regularization by combining KL balancing and free bits, modifying the network architecture, and so forth. A main modification relevant to our method is that DreamerV3 explicitly partitions the dynamics loss in Eq. (2) into a dynamics loss and a representation loss as follows:

$$\begin{aligned} &\text{Dynamics loss:} && \mathcal{L}_{\text{dyn}}(\theta) = \max(1, \mathrm{KL}\left[\mathrm{sg}(q_\theta(z_t \mid z_{t-1}, a_{t-1}, o_t)) \,\|\, p_\theta(\hat{z}_t \mid z_{t-1}, a_{t-1})\right]), \\ &\text{Representation loss:} && \mathcal{L}_{\text{rep}}(\theta) = \max(1, \mathrm{KL}\left[q_\theta(z_t \mid z_{t-1}, a_{t-1}, o_t) \,\|\, \mathrm{sg}(p_\theta(\hat{z}_t \mid z_{t-1}, a_{t-1}))\right]). \end{aligned} \tag{14}$$

Since $\mathcal{L}_{\text{dyn}}(\theta)$ and $\mathcal{L}_{\text{rep}}(\theta)$ yield the same loss value, leading to identical learned coefficients, we implement Harmony DreamerV3 by recombining the two losses into $\mathcal{L}_d(\theta)$ as follows:

$$\mathcal{L}_d(\theta) \doteq \alpha \mathcal{L}_{\text{dyn}}(\theta) + (1-\alpha)\mathcal{L}_{\text{rep}}(\theta). \tag{15}$$

Here α is the KL balancing coefficient predefined by DreamerV3. In this way, we can use the same learning objective as Eq. (5) for Harmony DreamerV3.

[3] https://github.com/zchuning/repo

DreamerPro. DreamerPro (Deng et al., 2022) is a reconstruction-free model-based RL method that incorporates prototypical representations in the world model learning process. The overall learning objective of the DreamerPro method is defined as follows:

$$\mathcal{L}_{\text{DreamerPro}}(\theta) = \mathcal{L}_{\text{SwAV}}(\theta) + \mathcal{L}_{\text{Temp}}(\theta) + \mathcal{L}_{\text{R}}(\theta) + \mathcal{L}_{\text{KL}}(\theta). \tag{16}$$

The $\mathcal{L}_{\text{SwAV}}$ term stands for prototypical representation loss used in SwAV (Caron et al., 2021), which improves prediction from an augmented view and induces useful features for static images. $\mathcal{L}_{\text{Temp}}$ stands for temporal loss that considers temporal structure and reconstructs the cluster assignment of the observation instead of the visual observation itself. As $\mathcal{L}_{\text{SwAV}} + \mathcal{L}_{\text{Temp}}$ replaces $\mathcal{L}_o$ in Eq. (2), we build our Harmony DreamerPro by substituting the overall learning objective into the following:

$$\mathcal{L}_{\text{Harmony DreamerPro}}(\theta) = \sum_{i \in \{\text{SwAV,Temp,R,KL}\}} \frac{1}{\sigma_i} \mathcal{L}_i(\theta) + \log(1 + \sigma_i). \tag{17}$$

C.3. Hyperparameters

Our proposed HarmonyDream only involves adding lightweight harmonizers, each corresponding to a single learnable parameter, and thus **does not introduce any additional hyperparameters**. For Harmony DreamerV3 and Harmony DreamerPro, we use the default hyperparameters of DreamerV3 and DreamerPro, respectively. For our HarmonyDream based on DreamerV2, we use the same set of hyperparameters as DreamerV2 (Hafner et al., 2021). Important hyperparameters for HarmonyDream are listed in Table 2.

Table 2. Hyperparameters in our HarmonyDream based on DreamerV2. We use the same hyperparameters as DreamerV2.

Hyperparameter	Value
Observation size	$64 \times 64 \times 3$
Observation preprocess	Linearly rescale from $[0, 255]$ to $[-0.5, 0.5]$
Action Repeat	1 for Meta-world 2 for RLBench, DMCR and Natural Background DMC
Max episode length	500 for Meta-world, DMCR and Natural Background DMC 200 for RLBench
Early episode termination	True for RLBench, False otherwise
Trajectory segment length T	50
Random exploration	5000 environment steps for Meta-world and RLBench 1000 environment steps for DMCR and Natural Background DMC
Replay buffer capacity	10^6
Training frequency	Every 5 environment steps
Imagination horizon H	15
Discount γ	0.99
λ-target discount	0.95
Entropy regularization η	1×10^{-4}
Batch size	50 for Meta-world and RLBench 16 for DMCR and Natural Background DMC
RSSM hidden size	1024
World model optimizer	Adam
World model learning rate	3×10^{-4}
Actor optimizer	Adam
Actor learning rate	8×10^{-5}
Critic optimizer	Adam
Critic learning rate	8×10^{-5}
Evaluation episodes	10

C.4. Analysis Experiment Details (Fig. 2c and 3)

For the analysis in Sec. 2.3, namely Fig. 2c and 3, we conduct our experiments on a fixed training buffer to better ablate distracting factors. We first train a separate DreamerV2 agent and use training trajectories collected during its whole training process as our fixed buffer. The fixed buffer comprises 250K environment steps and covers data from low-return to high-return trajectories (Levine et al., 2020). We then offline train our DreamerV2 agents with different reward loss coefficients on this buffer. All other hyperparameters, such as training frequency, training steps, and evaluation episodes, are the same as in Table 2.

Details for Fig. 2c We denote the agent trained using $w_r = 1$ as *original weight* and trained using $w_r = 100$ for Lever Pull, $w_r = 10$ for Handle Pull Side and Hammer as *balanced weight*. To build the state regression dataset, first, we gather 10,000 segments of trajectories, each with a length of 50, from the evaluation episodes of both the agent trained using *original weight* and the agent trained using *balanced weight*. These segments are then combined into a dataset comprising 20,000 segments. This dataset is subsequently divided into a training set and a validation set at a ratio of 90% to 10%. Each data point in the dataset consists of a ground truth state and a predicted state representation, where the ground truth state is made up of the actual positions of task-relevant objects. We use a 4-layer MLP with a hidden size of 400 and an MSE loss to regress the representation to the ground-truth state. We report regression loss results on the validation set.

Details for Fig. 3 In the Lever Pull task, the robot needs to reach the end of a lever (marked in blue in the observation) and pull it to the designated position (marked in red in the observation). We utilize a trajectory where the default DreamerV2 with $w_r = 1$ fails to lift the lever to analyze the reason behind its poor performance. Both agents use 15 frames for observation and reconstruction and predict 35 frames open-loop. We plot each image with an interval of 5 frames in Fig. 3.

C.5. Computational Resources

We implement our HarmonyDream based on DreamerV2 using PyTorch (Paszke et al., 2019). Training is conducted with automatic mixed precision (Micikevicius et al., 2018) on Meta-world and RLBench and full precision on DMCR. In terms of training time, it takes ∼24 hours for each run of Meta-world experiments over 250K environment steps, ∼24 hours for RLBench over 500K environment steps, and ∼23 hours for DMCR over 1M environment steps, respectively. The lightweight harmonizers introduced by HarmonyDream do not affect the training time. In terms of memory usage, Meta-world and RLBench experiments require ∼10GB GPU memory, and DMCR requires ∼5GB GPU memory, thus, the experiments can be done using typical 12GB GPUs.

D. Extended Discussions

D.1. Differences with DreamerV3

When we started this work, DreamerV3 had not been released. Thus, we primarily conduct experiments based on DreamerV2, as mentioned in the main paper. We state here that the modifications introduced by DreamerV3 do not fully address the problem of task domination inside world models, which is the problem HarmonyDream intends to solve. As shown in Appendix E.1 and E.6, HarmonyDream applied to DreamerV3 can further unleash the potentials of this base method.

There are mainly two changes of DreamerV3 relevant to improving world model learning: KL balancing and symlog predictions. We have already shown in Appendix C.2 that KL balancing is orthogonal to our method and that we can easily incorporate this modification into our approach. On the other hand, symlog predictions also do not solve our problem of seeking a balance between reward modeling and observation modeling. First of all, the symlog transformation only shrinks extremely large values but is unable to rescale various values into exactly the same magnitude, while our harmonious loss properly addresses this by dynamically approximating the scales of the values. More importantly, the primary reason why L_r has a significantly smaller loss scale is the difference in dimension: as we have stated in Sec 2.3, the observation loss L_o usually aggregates $H \times W \times C$ dimensions, while the reward loss L_r is derived from only a scalar. In summary, using symlog predictions as DreamerV3 only mitigates the problem of differing per-dimension scales (typically across environment domains) by a static transformation, while our method aims to dynamically balance the overall loss scales across tasks in world model learning, considering together per-dimension scales, dimensions, and training dynamics.

In practice, DreamerV3 uses twohot symlog predictions for the reward predictor to replace the MSE loss in DreamerV2. This approach increases the scale of the reward loss, but is insufficient to mitigate the domination of the image loss. We observe that the reward loss in DreamerV3 is still significantly smaller than the observation loss, especially for visually demanding domains such as RLBench, where the reward loss is still two orders of magnitude smaller.

D.2. Comparisons with Multi-task Learning Methods

In this paper, we understand world model learning from a multi-task or multi-objective view. Methods in the field of multi-task learning or multi-objective learning can be roughly categorized into loss-based and gradient-based. Since gradient-based methods mainly address the problem of gradient conflicts (Yu et al., 2020a; Liu et al., 2021), which is not the main case in world model learning, we focus our discussion on loss-based methods, which assigns different weights to task

losses by various criteria. We choose the following methods as our baselines to discuss differences and conduct comparison experiments. The experiment results can be found in Fig. 10b of the main paper.

Uncertainty Weighting (UW, Kendall et al. (2018)) balances tasks with different scales of targets, which is measured as uncertainty of outputs. As pointed out in Section 2.2, in world model learning, observation loss $\mathcal{L}_o(\theta) = -\log p_\theta\left(o_t \mid z_t\right) = -\sum_{h,w,c} \log p_\theta(o_t^{(h,w,c)} \mid z_t)$ and reward loss $\mathcal{L}_r(\theta) = -\log p_\theta\left(r_t \mid z_t\right)$ differs not only in scales but also in dimensions. To implement UW, we opt for depicting the uncertainty of each pixel. By assuming all pixel values share a common standard deviation σ_o for Gaussian distributions, the uncertainty-weighted image loss takes the following form: $\mathcal{L}(\theta, \sigma_o) = \sum_{h,w,c}(\hat{o}_t^{(h,w,c)} - o_t^{(h,w,c)})^2/2\sigma_o + \log \sigma_o = {\sigma_o}^{-1}\mathcal{L}_o(\theta) + HWC \log \sigma_o$. A detailed explanation of the differences between our harmonious loss and UW is provided in the discussion section in Section 3.

Dynamics Weight Average (DWA, Liu et al. (2019)) balances tasks according to their learning progress, illustrating the various task difficulties. However, in world model learning, since the data in the replay buffer is growing and non-stationary, the relative descending rate of losses may not accurately measure task difficulties and learning progress.

NashMTL (Navon et al., 2022) is the most similar to our method, whose optimization direction has balanced projections of individual gradient directions. However, its implementation is far more complex than our method, as it introduces an optimization procedure to determine loss weights on each iteration. In our experiments, we also find this optimization is prone to deteriorate to produce near-zero weights without careful tuning of optimization parameters.

In Fig 10b, we compare against the multi-task methods we mentioned above. Experiments are conducted on *Lever Pull* from Meta-world, *Push Button* from RLBench, and *Cheetah Run* from DMCR, respectively. Our method is the most effective among multi-task methods and has the advantage of simplicity. Although NashMTL produces similar results on the Lever Pull task, it outputs extreme loss weights on the other two tasks, which accounts for its low performance. Our HarmonyDream, on the other hand, uses a rectified loss that effectively mitigates extremely large loss weights.

E. Extended Experiment Results

E.1. Atari 100K Experiments

We based our implementation of HarmonyDream applied to DreamerV3 (denoted as *Harmony DreamerV3*) on the official DreamerV3 codebase[4]. To ensure the fairness and quality of our results, we also reproduced DreamerV3 results using the official code and configurations. Fig 15 shows Atari learning curves of the reproduced DreamerV3 and our Harmony DreamerV3 on all 26 environments. Note here that our learning curves are plotted using **evaluation scores**, rather than averaged training scores as in DreamerV3, which may account for part of the differences between our curves and that reported by Hafner et al. (2023). Both DreamerV3 and our Harmony DreamerV3 are evaluated for 100 episodes every 20K environment steps. In each curve, the solid line represents the average evaluation score across 5 seeds, while the shaded region indicates the standard deviation. This is consistent with the figure representation in DreamerV3.

Table 3 shows the mean score and aggregated human normalized scores of our Harmony DreamerV3 on Atari tasks, compared to other methods. The scores in the *SimPLe, TWM, IRIS*, and *DreamerV3 (Original)* columns correspond to the scores reported in their papers, respectively. The *DreamerV3 (Reproduced)* column contains scores reproduced using the official codebase. The reproduced results exhibit performance comparable to the reported results. The slight discrepancy in the human-normalized score is primarily attributed to the subpar performance in the Crazy Climber game. Our Harmony DreamerV3 significantly improves upon the base method's performance. It either matches or surpasses DreamerV3 in 23 of the 26 tested environments, thereby setting a new state-of-the-art benchmark with a human mean score of 136.5%. It's noteworthy that this enhancement is achieved without the addition of any hyperparameters or alterations to any network structures. By harmonizing tasks in world model learning, we fully exploit the inherent potential of our base model, further highlighting the value of our work.

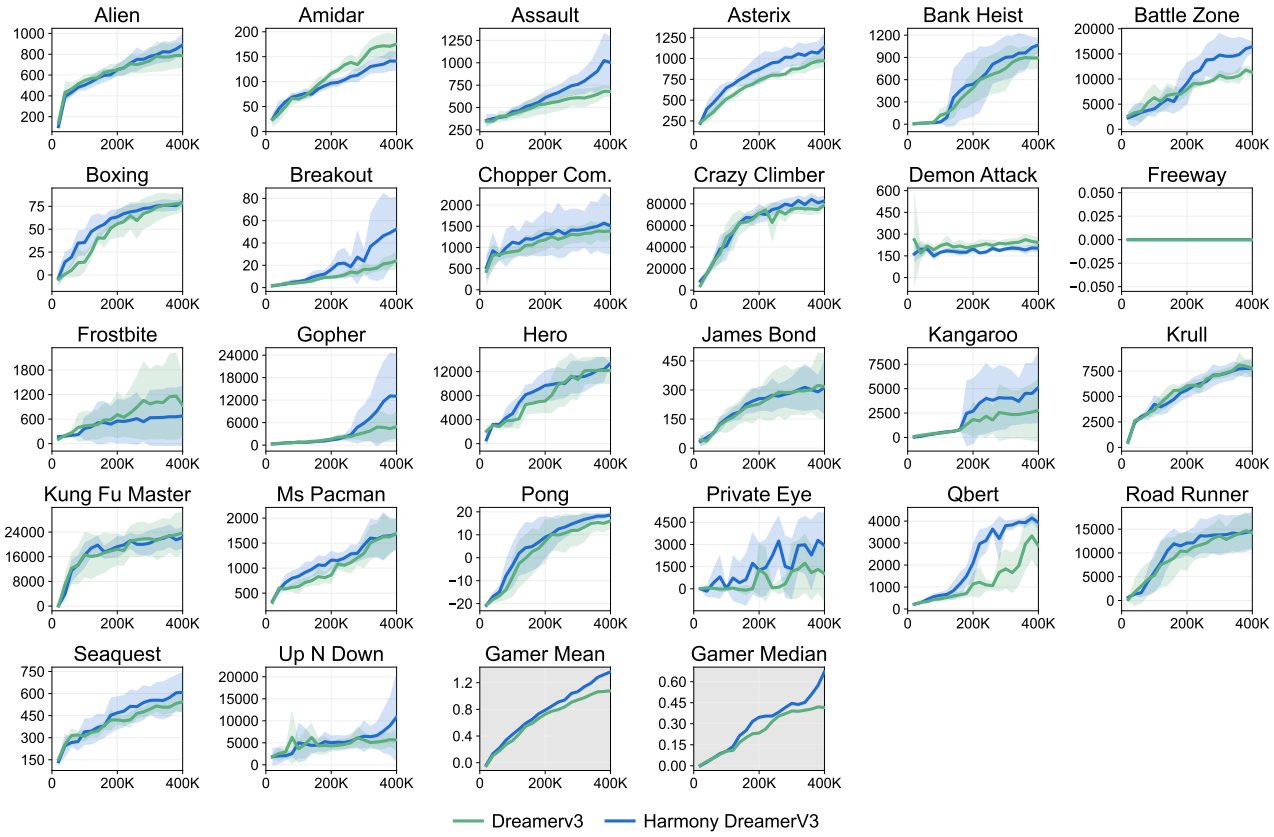


Figure 15. Atari learning curves of DreamerV3 (reproduced) and Harmony DreamerV3 with a budget of 400K frames, amounting to 100K policy steps.

[4]We use this version of the DreamerV3 codebase: `https://github.com/danijar/dreamerv3/tree/8fa35f`. We notice that several changes have made to this codebase subsequent to our paper's initial release in February 2024.

Table 3. Mean scores on the Atari 100K benchmark per game as well as the aggregated human normalized mean and median. Bold numbers indicate scores within 5% of the best.

Game	Random	Human	SimPLe (2020)	TWM (2023)	IRIS (2023)	DreamerV3 (Original)	DreamerV3 (Reproduced)	Harmony DreamerV3
Alien	228	7128	617	675	420	**959**	786	890
Amidar	6	1720	74	122	143	139	**175**	141
Assault	222	742	527	683	**1524**	706	680	1003
Asterix	210	8503	**1128**	**1117**	854	932	974	**1140**
Bank Heist	14	753	34	467	53	649	894	**1069**
Battle Zone	2360	37188	4031	5068	13074	12250	11314	**16456**
Boxing	0	12	8	**78**	70	**78**	**78**	**80**
Breakout	2	30	16	20	**84**	31	24	53
Chopper Com.	811	7388	979	**1697**	1565	420	1390	1510
Crazy Climber	10780	35829	62584	71820	59234	**97190**	78969	82739
Demon Attack	152	1971	208	350	**2034**	303	241	203
Freeway	0	30	17	24	**31**	0	0	0
Frostbite	65	4335	237	**1476**	259	909	939	679
Gopher	258	2412	597	1675	2236	3730	4936	**13043**
Hero	1027	30826	2657	7254	7037	11161	12160	**13378**
James Bond	29	303	101	362	**463**	**445**	318	317
Kangaroo	52	3035	51	1240	838	4098	2773	**5118**
Krull	1598	2666	2205	6349	6616	**7782**	**7764**	**7754**
Kung Fu Master	258	22736	14862	**24555**	21760	21420	**23753**	22274
Ms Pacman	307	6952	1480	1588	999	1327	**1696**	**1681**
Pong	-21	15	13	**19**	15	**18**	**18**	**19**
Private Eye	25	69571	35	87	100	882	1036	**2932**
Qbert	164	13455	1289	3331	746	3405	2872	**3933**
Road Runner	12	7845	5641	9109	9615	**15565**	14248	14646
Seaquest	68	42055	683	**774**	661	618	544	665
Up N Down	533	11693	3350	**15982**	3546	7667	5636	10874
Human Mean	0%	100%	33%	96%	105%	112%	108%	**136.5%**
Human Median	0%	100%	13%	51%	29%	49%	42%	**67.1%**

E.2. DeepMind Control Suite Experiments

The DeepMind Control Suite (DMC, Tassa et al. (2018)) is a widely used benchmark for visual locomotion. We have conducted additional experiments on four tasks: Cheetah Run, Quadruped Run, Walker Run, and Finger Turn Hard. In Fig. 16, we present comparisons between our HarmonyDream and the base DreamerV2. We note that the performance of relatively easy DMC tasks has been almost saturated by recent literature (Yarats et al., 2021; Hafner et al., 2021), and we suppose that in this domain, current limitations of model-based methods are not rooted in the world model, but rather in behavior learning (Hafner et al., 2023), which falls outside the scope of our method and contributions. Nevertheless, our HarmonyDream is still able to obtain a noticeable gain in performance in the more difficult Quadruped Run task.

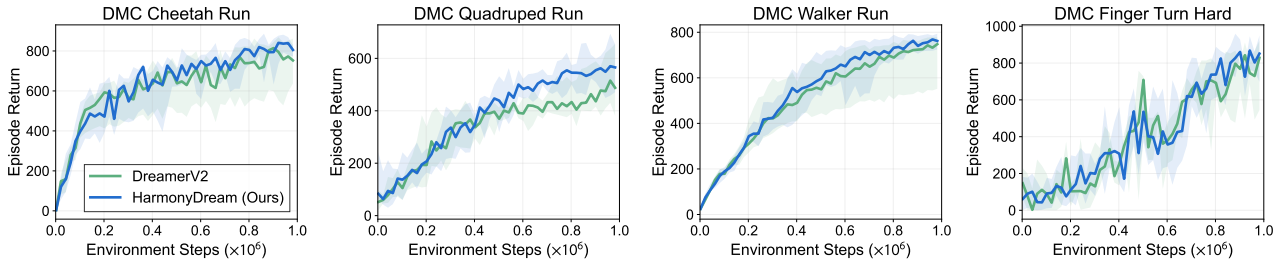
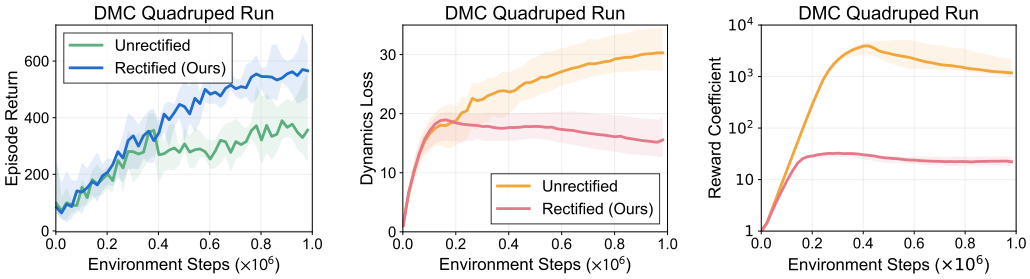


Figure 16. Learning curves of HarmonyDream and DreamerV2 on the DMC domain.

E.3. Ablation Study on Rectified Harmonious Loss

In Sec. 3, we have already presented a detailed explanation on the necessity of our *rectified harmonious loss*, changing the regularization term from $\log \sigma_i$ in Eq. (4) to $\log(1 + \sigma_i)$ in Eq. (5). Here, we present experimental results to support our claim. We use *Unrectified* to note our method trained using the objective in Eq. (4), and *Rectified (Ours)* to note our method trained using Eq. (5). As shown in Fig. 17 and Fig. 18, the excessively large reward coefficient (Fig. 17c) for *Unrectified* can lead to a divergence in the dynamics loss (Fig. 17b), which in turn negatively impacts performance (Fig. 17a and Fig. 18).



(a) Learning curves. (b) Dynamics loss. (c) Reward coefficient.

Figure 17. Training curves for *Unrectified HarmonyDream* (denoted as Unrectified) using Eq. (4) on the DMC Quadruped Run task, in comparison with our HarmonyDream (denoted as Rectified).

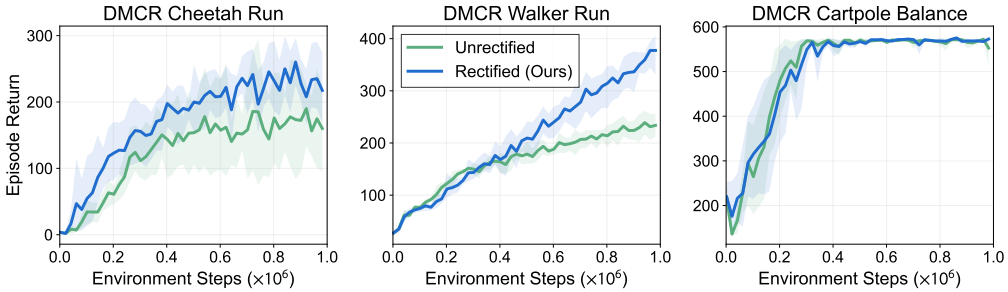


Figure 18. Learning curves for *Unrectified HarmonyDream* (denoted as Unrectified) using Eq. (4) on the DMCR domain, in comparison with our HarmonyDream (denoted as Rectified).

E.4. Ablation Study on Adjusting Dynamics Loss Weight w_d

Manually tuning the dynamics loss coefficient w_d (e.g. $w_d = 0.1$) is common in MBRL methods (Hafner et al., 2021; 2023; Seo et al., 2022a;b). We note that our HarmonyDream differs from these previous approaches as we treat the different losses in a multi-task view and balance loss scales between them, while previous approaches see w_d simply as a hyperparameter. Fig. 19 shows a comparison between fixing w_d to 1 in HarmonyDream (denoted as *HarmonyDream* $w_d = 1$) and using σ_d to balance w_d (denoted as *HarmonyDream (Ours)*), where our proposed HarmonyDream performs slightly better than the one fixing w_d, and both methods outperform DreamerV2 by a clear margin. This result highlights the importance of harmonizing two different modeling tasks in world models, instead of only tuning on the shared dynamics part of them.

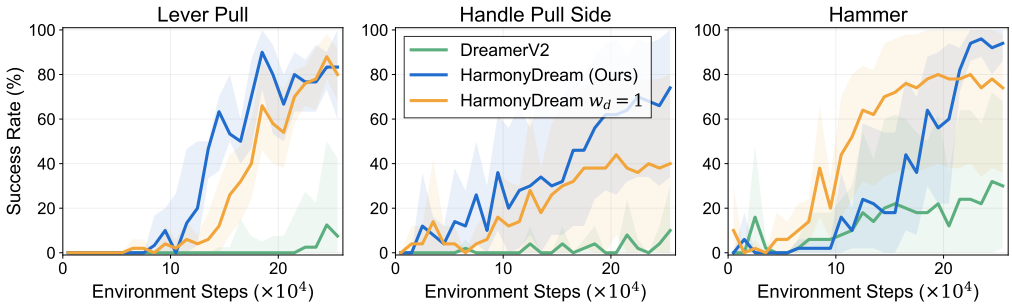
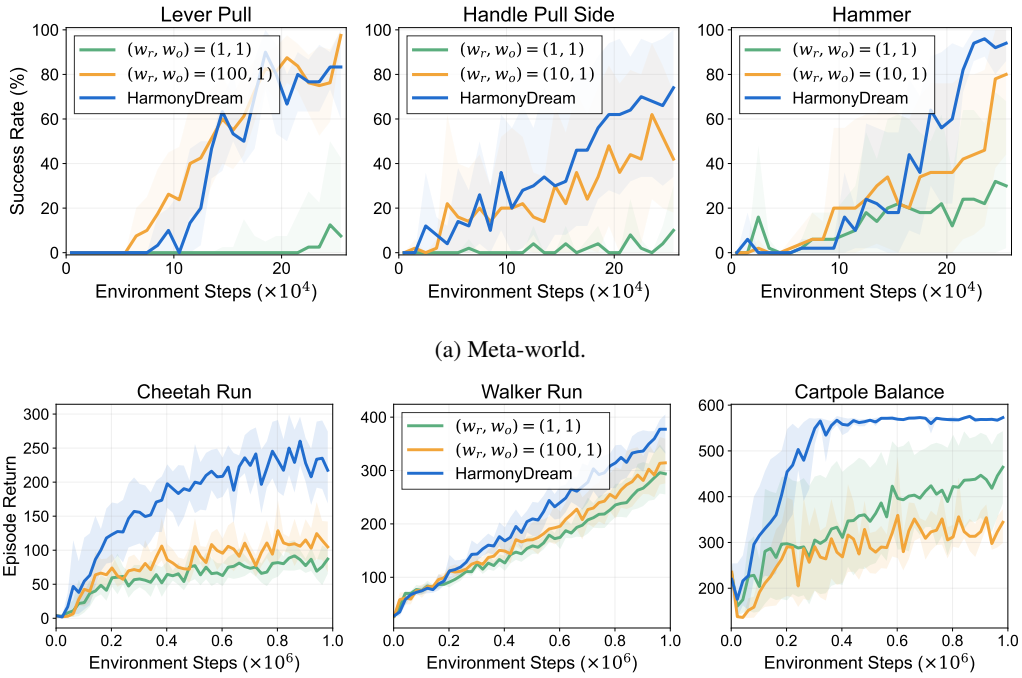


Figure 19. Ablation on adjusting w_d in HarmonyDream.

E.5. Comparison to Tuned Weights

We present a direct comparison between our HarmonyDream and manually tuned weights for DreamerV2. For the Meta-world domain, we plot the tuned better results from $w_r \in \{10, 100\}, w_o = 1$. For the DMCR domain, we plot tuned results using $w_r = 100, w_o = 1$. Results in Fig. 20 show that our HarmonyDream outperforms manually tuned weights in most tasks, which adds to the value of our method.



(a) Meta-world.

(b) DMCR.

Figure 20. Learning curves of HarmonyDream compared to tuned weights on Meta-world and DMCR.

E.6. Extended Results of Harmony DreamerV3 on Meta-world

In Fig. 8 of the main paper, we have presented the aggregated results of our HarmonyDream generalized to DreamerV3 (referred to as *Harmony DreamerV3*), on three Meta-world tasks: Lever Pull, Handle Pull Side, and Hammer. Here in Fig. 21, we provide individual results of these three tasks, along with the results of an additional task, Sweep Into. Our approach consistently improves the sample efficiency of our base method, proving excellent generality.

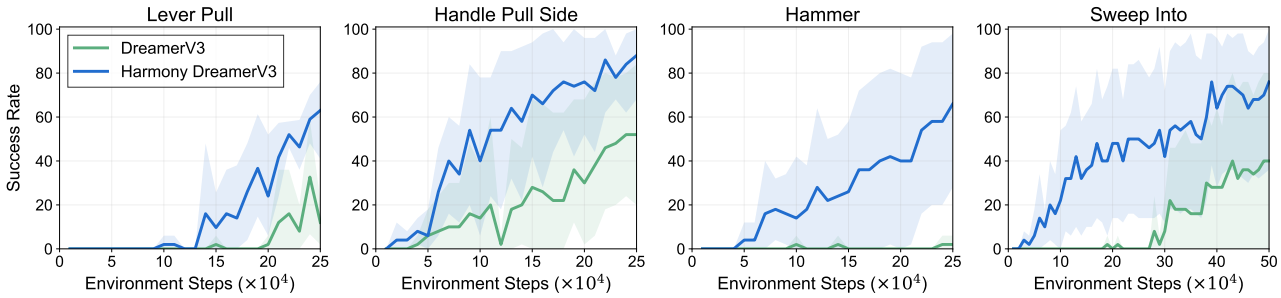


Figure 21. Detailed results of Harmony DreamerV3 on Meta-world.

E.7. Additional Results of Harmony DreamerV3 on Crafter

Crafter (Hafner, 2022) is a 2D open-world survival game benchmark where the agent needs to learn multiple skills within a single environment. High rewards in this benchmark demand robust generalization and representation capabilities from the agent. However, our method is not typically effective in the Crafter domain, which is characterized by clear observations and distinct target objects. As a result, Harmony DreamerV3 marginally outperforms DreamerV3, as shown in Fig. 22.

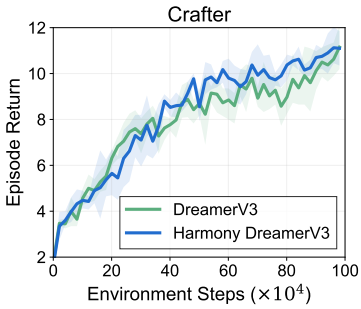
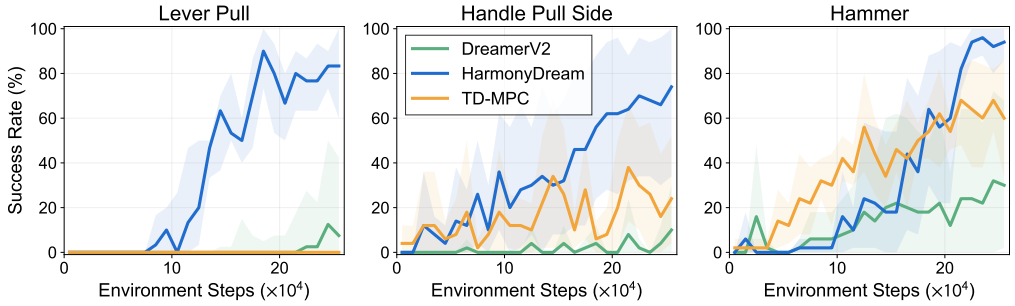


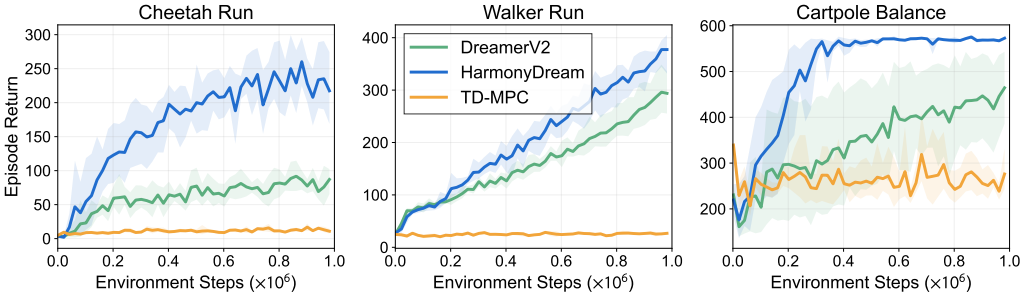
Figure 22. Results of Harmony DreamerV3 on Crafter.

E.8. Extended Results of Implicit MBRL Methods

We observe that the performance of TD-MPC (Hansen et al., 2022) is fairly low compared to our HarmonyDream. Due to our limited computational resources, we only conduct experiments on the Meta-world and DMCR domain. The Meta-world result in Fig. 10a aggregates over three tasks: Lever Pull, Handle Pull Side, and Hammer, which are the same three tasks as in Fig. 8. Full TD-MPC results in Fig. 23 show that TD-MPC is unable to learn a meaningful policy in some tasks.



(a) Meta-world.



(b) DMCR.

Figure 23. Learning curves of TD-MPC.

E.9. Comparison with Denoised MDP

HarmonyDream shares a similar point with Denoised MDP (Wang et al., 2022) in enhancing task-centric representations. However, the two approaches are orthogonal. In Fig. 24, we show a comparison of our method to Denoised MDP. Denoised MDP performs information decomposition by changing the MDP transition structure and utilizing the reward as a guide to separate task-relevant information. However, since Denoised MDP does not modify the weight for the reward modeling

task, the observation modeling task can still dominate the learning process. Consequently, the training signals from the reward modeling task may be inadequate to guide decomposition. It's also worth noting that Denoised MDP only added noise distractors to task-irrelevant factors in their DMC experiments. On the other hand, the benchmark adopted in our experiments, DMCR, adds visual distractors to both task-irrelevant and task-relevant factors, such as the color of the body and floor, which adds complexity to both factors and results in more challenging tasks. These two reasons above can account for the low performance of Denoised MDP in our benchmarks.

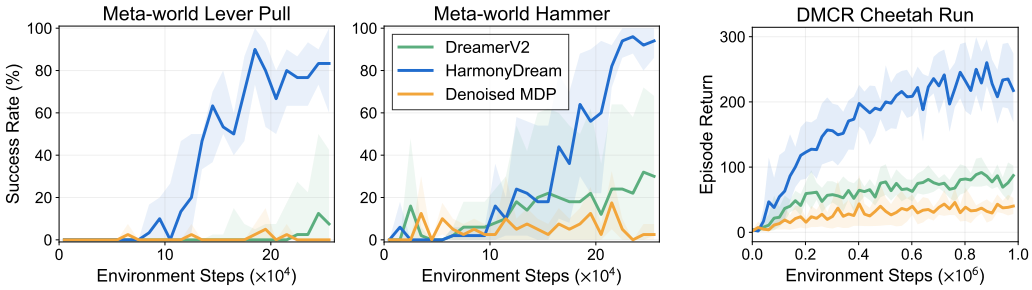
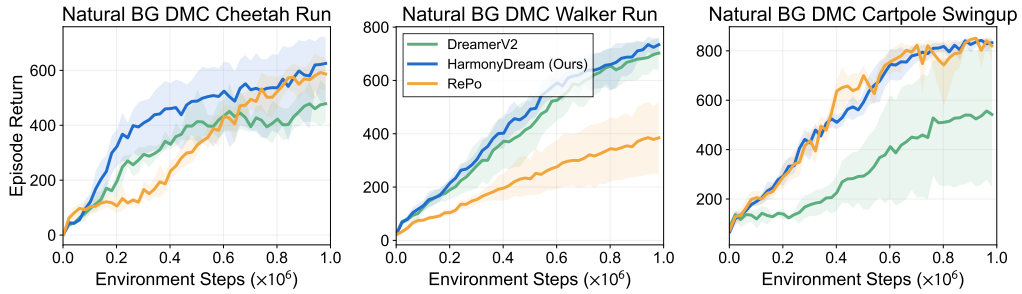


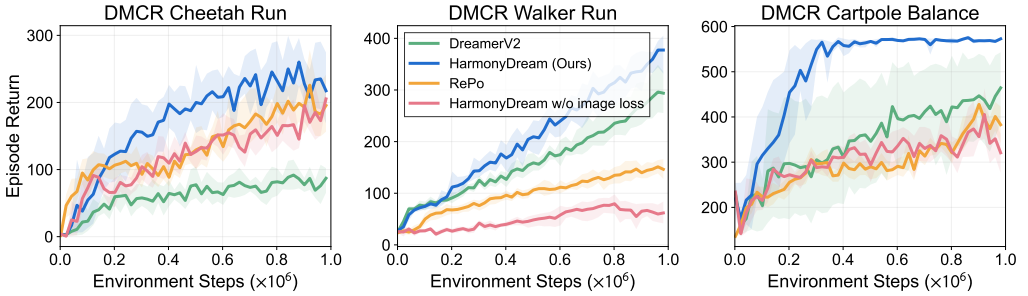
Figure 24. Comparison of HarmonyDream with Denoised MDP.

E.10. Comparison with RePo

RePo (Zhu et al., 2023) is a modification on DreamerV2 that removes the observation reconstruction loss while introducing a dynamically adjusted coefficient of dynamics loss. As shown in Fig. 25, our HarmonyDream shows a higher sample efficiency compared to RePo on both natural background DMC and DMCR. It is notable that RePo takes a similar form as HarmonyDream without observation loss (i.e. fixing $w_o = 0$). While the adjusted coefficient of RePo does not guarantee uniform loss scales, we observe in our experiments that it, in effect, makes dynamics loss and reward loss have more similar scales. We demonstrate on the DMCR domain that the two share similar learning curves, which to some extent enhances our Finding 3, that learning signals from rewards alone is inadequate for sample-efficient learning due to limited representation learning capability. We also note that RePo still needs to carefully tune a crucial hyperparameter, the information bottleneck ϵ, while HarmonyDream does not introduce any hyperparameters.



(a) Natural background (BG) DMC.



(b) DMCR.

Figure 25. Comparison of HarmonyDream with RePo.

E.11. Learned Coefficients

Fig. 27 illustrates the learned harmony loss coefficients for two Meta-world tasks: Lever Pull and Handle Pull Side. The harmonized reward loss coefficient for Lever Pull is observed to be higher than that for Handle Pull Side. This observation aligns with the fact that the coefficient pair $(w_r, w_o) = (100, 1)$ yields superior performance on Lever Pull, while the pair $(w_r, w_o) = (10, 1)$ facilitates better learning on Handle Pull Side, as depicted in Fig. 2a.

Additionally, we present the impact of varying loss coefficients for DreamerV2 on the Meta-world Hammer task in Fig. 26, supplementing the information in Fig. 2a.

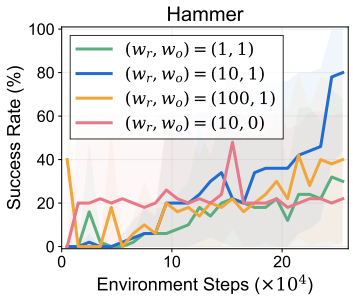


Figure 26. Effects of different loss coefficients on an additional task.

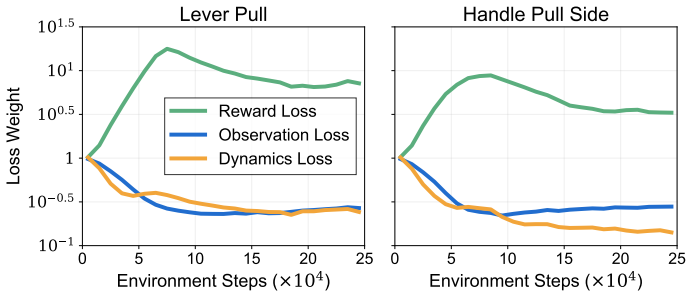


Figure 27. Learned harmony loss coefficients on Meta-world tasks.

E.12. Quantitative Evaluation of the Beneficial Impact of Observation Modeling on Reward Modeling

To explore the possible beneficial impact of observation modeling on reward modeling, we utilize the offline experimental setup in Fig 2c and 3, whose details are described in Appendix C.4. We offline train two DreamerV2 agents using task weights $(w_r = 100, w_o = 1)$ and $(w_r = 100, w_o = 0)$ and evaluate the ability to accurately predict rewards on a validation set with the same distribution as the offline training set. For this task, we gathered 20,000 segments of trajectories, each of length 50. We utilized 35 frames for observation and predicted the reward for the remaining 15 frames. Results are reported in the form of average MSE loss. We observe that the world model with observation modeling predicts the reward better than the world model that only models the reward. The prediction loss of $(w_r = 100, w_o = 1)$ is 0.379, while the loss of $(w_r = 100, w_o = 0)$ is 0.416. This result indicates that observation modeling has a positive effect on reward modeling.