Coupled Local and Global World Models for Efficient First Order RL

Author Names Omitted for Anonymous Review. Paper-ID 942

***Abstract*—World models offer a promising avenue for more faithfully capturing complex dynamics, including contacts and non-rigidity, as well as complex sensory information, such as visual perception, in situations where standard simulators struggle. However, these models are computationally complex to evaluate, posing a challenge for popular RL approaches that have been successfully used with simulators to solve complex locomotion tasks but yet struggle with manipulation. This paper introduces a method that bypasses simulators entirely, training RL policies inside world models learned from robots' interactions with real environments. At its core, our approach enables policy training with large-scale diffusion models via a novel decoupled first-order gradient (FoG) method: a full-scale world model generates accurate forward trajectories, while a lightweight latent-space surrogate approximates its local dynamics for efficient gradient computation. This coupling of a local and global world model ensures high-fidelity unrolling alongside computationally tractable differentiation. We demonstrate the efficacy of our method on the Push-T manipulation task, where it significantly outperforms PPO in sample efficiency. We further evaluate our approach through an ego-centric object manipulation task with a quadruped. Together, these results demonstrate that learning inside data-driven world models is a promising pathway for solving hard-to-model RL tasks in image space without reliance on hand-crafted physics simulators.**

I. Introduction

Recent breakthroughs in reinforcement learning (RL) for locomotion [7, 21, 40, 5, 4] have yielded highly agile policies, trained in parallelized GPU simulators and seamlessly deployed on real robots for acrobatic tasks. Despite these gains, RL confronts substantial barriers in robotics. Its sample inefficiency blocks learning from pixel-level inputs or direct real-robot interactions. While RL shines in locomotion through GPU-accelerated, highly parallel simulations, achieving sim-to-real alignment in manipulation often proves laborious and error-prone [39]. Many environments are inherently hard to model, including granular dynamics of gravel bags, pressurized deformations of an aluminum can that risk explosion under excessive force and the precise mechanics of its pull-tab opening, or the intricacies of folding a T-shirt, each requiring custom physics engines in an endless array of scenarios. Moreover, in everyday unstructured settings the robot typically does not have access to ground-truth states, and must instead rely on high-dimensional observations such as RGB, depth, or point clouds (instrumenting every scene with motion capture is not scalable). Modeling these observation spaces further amplifies the difficulty of building accurate simulators. Additionally, crafting effective reward signals for these problems is notoriously challenging, as exemplified by T-shirt manipulation.

Differentiable world models, when paired with first-order gradient (FoG) methods [14, 15, 22, 9], offer a pathway to resolve three of the aforementioned prominent issues: bridging the gap between simulation and reality by learning from real-world data to replace simulators; simplifying the modeling of varied environments through data acquisition rather than physics-based simulation; and alleviating sample inefficiency in RL via gradient signals with reduced variance [9] relative to score-function based algorithms such as PPO [33] or SAC [12, 13]. Despite this potential, their application in robotics is severely constrained. These models frequently exhibit trajectory drift during forward unrolls [28, 35] or resist efficient backpropagation, as seen in diffusion world models operating on images. For these, inefficiencies arise from the repeated application of denoising steps, which inflate backpropagation through time (BPTT) lengths; the inefficiency of backpropagating through pixels; and the growing scale of realistic, generalizable world models, which dramatically increases the time and memory required for differentiation.

This paper aims to address these issues. It builds on Decoupled forward-backward Model-based policy Optimization (DMO) [2], which separates forward simulation from backward differentiation, to make high-fidelity world models usable in FoG model-based RL (FoG-MBRL). Concretely, we propose a framework that learns a diffusion-based world model from images of real robot interactions, deploying it as a simulator surrogate, while simultaneously learning a secondary more light-weight Recurrent State Space Model (RSSM)—acting as a "local" world model in a low-dimensional latent space—to supply stable, low-variance gradients without backpropagating through diffusion. This decoupling resolves key obstacles of FoG-MBRL with pixel-space models, thereby opening the door to practical FoG-MBRL with powerful image-based world models. We empirically evaluate the capabilities of our approach on a manipulation and a loco-manipulation tasks learned directly from real robot data.

II. Related Work

A. World Models and Policy Learning.

Model-Based Reinforcement Learning (MBRL) approaches are generally distinguished by their control strategy. A prevalent line of work employs the learned model for online planning via Model Predictive Control (MPC) [8, 18, 19] or Monte-Carlo Tree Search [32, 30, 31, 25]. While effective, these methods incur high computational costs at inference time. Conversely, policy optimization methods encode control

into a neural network learned inside the world model. However, seminal works in this category, like DreamerV3 [15], typically rely on online interaction with a simulator. Simulator-free approaches like DayDreamer [34] learn directly from real-world interaction, but rely on Variational Auto-Encoders (VAEs) [26], which often exhibit limited expressivity and struggle to model complex distributions with high fidelity. In contrast, we target simulator-free learning using Diffusion Models, which offer superior modeling power but present unique policy optimization challenges.

B. Gradient Estimation for Policy Learning.

Optimization in MBRL generally falls into zeroth-order or first-order methods. Zeroth-order algorithms [22, 27, 3, 17, 29, 23] treat the dynamics as a black box. While robust to model inaccuracies, these methods are notably sample-inefficient. First-order Gradient (FoG) methods [9, 11] calculate analytic gradients via backpropagation, offering high data efficiency. However, standard FoG requires the dynamics derivatives to be computationally tractable, which excludes heavy-weight models. A promising solution is Decoupled Model Optimization (DMO) [2]: utilizing a high-fidelity model for trajectory generation (forward) and a lightweight proxy model for gradient estimation (backward). While prior DMO works relied on the simulator for the forward pass, we extend this paradigm to the simulator-free setting by using a "global" Diffusion model. This allows us to leverage high-fidelity generation without incurring the prohibitive cost of backpropagating through the heavy diffusion network.

C. Diffusion for Efficient Control.

Recent works have successfully scaled World Models using diffusion parameterizations, such as DreamerV4 [17] and DIAMOND [1]. However, these methods typically rely on zeroth-order optimization, which requires a large number of samples. We argue that the combination of heavy-weight diffusion models and sample-inefficient zeroth-order optimization is computationally prohibitive, as it requires generating a vast number of expensive diffusion samples. Our method addresses this by employing sample-efficient FoG. By leveraging the decoupled architecture described above, we effectively distill the global diffusion priors into a local latent model, enabling fast analytic gradient updates and preventing the sampling bottleneck of diffusion-based zeroth-order RL methods.

III. Method

A. Background

a) Notations: We consider an infinite horizon, discounted Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, r, \gamma, f)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, γ is the discount factor, and $f : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ is the dynamics function. Here, $\mathcal{P}(\mathcal{S})$ denotes the space of probability distributions over states. We denote π_θ the parametrized policy.

b) Model-based RL with first order gradients: We need the partial derivatives of the dynamics with respect to state and action to compute the gradients of the RL objective $G(\theta)$ with respect to the policy parameters. In FoG-MBRL, the policy gradient is estimated from the gradient of the episodic return:

$$\begin{aligned}\nabla_\theta G(\theta) &= \nabla_\theta \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \\ &= \sum_{t=0}^{\infty} \gamma^t \left[\left.\frac{\partial r(s,a)}{\partial s}\right|_{(s_t,a_t)} \frac{ds_t}{d\theta} \right. \\ &\quad \left. + \left.\frac{\partial r(s,a)}{\partial a}\right|_{(s_t,a_t)} \frac{da_t}{d\theta} \right],\end{aligned} \tag{1}$$

with:

$$\begin{cases} \dfrac{ds_{t+1}}{d\theta} = \left.\dfrac{\partial f(s,a)}{\partial s}\right|_{(s_t,a_t)} \dfrac{ds_t}{d\theta} + \left.\dfrac{\partial f(s,a)}{\partial a}\right|_{(s_t,a_t)} \dfrac{da_t}{d\theta} \\ \dfrac{da_{t+1}}{d\theta} = \left.\dfrac{\partial \pi_\theta(s)}{\partial \theta}\right|_{(\tilde{\theta},s_t)} + \left.\dfrac{\partial \pi_\theta(s)}{\partial s}\right|_{(\tilde{\theta},s_t)} \dfrac{ds_t}{d\theta} \end{cases} \tag{2}$$

For $\left.\frac{\partial f(s,a)}{\partial s}\right|_{(s_t,a_t)}$ and $\left.\frac{\partial f(s,a)}{\partial a}\right|_{(s_t,a_t)}$, FoG-MBRL learns an approximation of the dynamics, $\hat{f}_\phi$, and uses it to predict $\hat{f}_\phi(\hat{s}_t, a_t) = \hat{s}_{t+1} \approx s_{t+1}$ and approximate $\left.\frac{\partial f(s,a)}{\partial s}\right|_{(s_t,a_t)} \approx \left.\frac{\partial \hat{f}_\phi(s,a)}{\partial s}\right|_{(\hat{s}_t,a_t)}$ and $\left.\frac{\partial f(s,a)}{\partial a}\right|_{(s_t,a_t)} \approx \left.\frac{\partial \hat{f}_\phi(s,a)}{\partial a}\right|_{(\hat{s}_t,a_t)}$. Importantly, the gradients are computed at the rollouts induced by the learned dynamics itself (the sequence of $\hat{s}_t$), so any compounding model error directly changes the states at which derivatives are taken. This observation motivates our decoupling framework.

B. Decoupling

We adopt DMO [2] for FoG-MBRL, which decouples forward rollouts from backward gradient computation by using distinct models: a forward model $f^{\rightarrow}$ to generate high-fidelity trajectories and the backward model $f_\phi^{\leftarrow} \equiv \hat{f}_\phi$ (the learned dynamics approximator from the FoG-MBRL framework) for gradient evaluation. Concretely, rather than evaluating the Jacobians of $f_\phi^{\leftarrow}$ at its own next state $s_{t+1}^{\leftarrow} = f_\phi^{\leftarrow}(s_t^{\leftarrow}, a_t)$, we evaluate them at the forward next state $s_{t+1}^{\rightarrow} = f^{\rightarrow}(s_t^{\rightarrow}, a_t)$, i.e., $\left.\frac{\partial f_\phi^{\leftarrow}(s,a)}{\partial s}\right|_{(s_{t+1}^{\rightarrow},a_{t+1})}$ (and analogously for $\partial/\partial a$), to approximate the true dynamics derivatives in the policy gradient of (1). This decoupling enables forward accuracy and backward tractability to be optimized independently.

C. Offline Data Collection

We collect "play data" on the real robot in image space only. By "play data" we mean data that was collected by a teleoperator or automatically by exploring the real environment, rather than trying to provide demonstrations to solve specific tasks. Since such data does not require clean temporal segmentation of the sequence into clearly defined tasks and does not mandate frequent environment resets during data collection, it

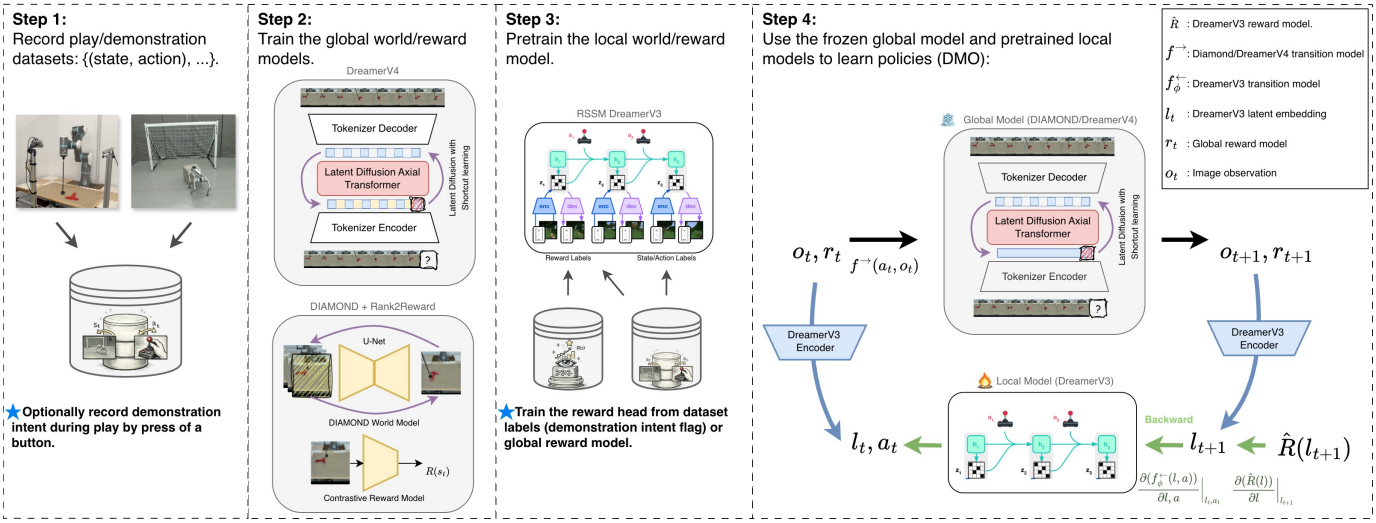


Fig. 1: Overview of the proposed approach. Global world/reward models are learned from play/demonstration data (Steps 1 and 2). Then, local world/reward models are pre-trained (Step 3). Policy is optimized with the DMO algorithm (Step 4): forward simulation uses the diffusion-based global world model in pixel space, trained on real robot data, to produce high-fidelity rollouts. The backward pass uses gradients computed from a local world model operating in a low-dimensional latent space. The local world/reward model is further fine-tuned during policy optimization.

is less expensive to collect compared to task-specific episodic datasets typically used for Behavior Cloning (BC).

D. Forward Models (Global)

1) Dynamics Models: As the forward model $f^{\rightarrow}$, we employ two distinct instantiations of diffusion world models. For the globally observable pushT task (where the environment's state is fully known from the current image observation), we adopt the DIAMOND [1] world model, a Convolutional Neural Network (CNN)-based architecture originally trained on Atari games. Through optimized noise scheduling and pre/post-conditioning schemes [24], DIAMOND enables rapid inference with minimal diffusion steps (three in our case).

While DIAMOND is able to generate highly photo-realistic scenes, it lacks the scalability needed for complex, partially observable tasks where a long temporal context length is essential. For instance, in our quadruped experiment involving ego-centric manipulation of a cube, the object may occasionally fall outside the robot's view, and maintaining a history of observations is necessary to ensure correct localization. To address this issue, we implement a transformer-based latent diffusion world model, based on the recent DreamerV4 [16] architecture (originally demonstrated for learning agents inside the Minecraft game). This implementation enables fast inference through shortcut learning [10] (few diffusion steps) and causal diffusion forcing [6] (KV-cache-friendly real-time recurrent generation). Furthermore, the architecture employs Axial Attention [20] to decouple temporal and spatial processing, significantly reducing the computational complexity of attention over long sequences. More importantly, because it is based on transformers, it exhibits the scalability needed for modeling complex environments and accommodates the need for temporally long context lengths. Both models are trained offline solely on the collected play data in image space.

2) Reward Models: To avoid the challenges of handcrafting reward functions in pixel space, we learn the reward function from data. In this paper, we adopt two different approaches for each of our tasks: For the Push-T task, we adopt an unsupervised reward learning approach based on the Bradley-Terry contrastive loss and trained from passive videos (play and optional demonstration data without the action signal) [37]. Specifically, given the start and goal states s_{start} and s_{goal} sampled from a temporal sequence, we train an energy function $f_\psi(s_t \mid s_{start}, s_{goal})$ to assign higher values to states s_t that are temporally closer to the goal:

$$\mathcal{L}(\psi) = \frac{e^{f_\psi(s_i)}}{e^{f_\psi(s_i)} + e^{f_\psi(s_j)}} \mathbf{1}\{i > j\} + \frac{e^{f_\psi(s_j)}}{e^{f_\psi(s_i)} + e^{f_\psi(s_j)}} \mathbf{1}\{i \leq j\}, \tag{3}$$

where s_i and s_j are randomly sampled frames between s_{start} and s_{goal}.

While this reward formulation based on goal images works very well for simple tasks with relatively short execution time, it breaks down for problems where the objective is hard to define as a sequence of goal images or requires multi-step, temporally long solutions. To account for such cases, we formulate another reward function that encodes whether a frame belongs to an intended demonstration or to a play behavior not intended to contribute to the task. Specifically, during data collection, the user turns on a label whenever they intend to perform the task and removes it otherwise. We show that this implicit signal can serve as a task-reward that does not rely on an explicit goal image or other conditioning signals. This reward may also be augmented by other helpful clues about the task progress (e.g., a key with a higher reward value whenever a milestone is met).

Unlike the contrastive reward function introduced earlier for

the Push-T task, we do not train an independent model here. Instead, we leverage the world model's learned knowledge by adding an extra token per frame for reward computation, which causally attends to the previous tokens while being masked during the world model's latent token computation. We post-train this world model on the binary-labeled play data and use the joint model as the final world+reward model for RL training.

Throughout the paper, we refer to the world and reward models described in this section as "global world model" and "global reward model," respectively, in contrast to the backward "local world model" and "local reward model." The global models are designed for scalability and long-horizon generalization across the state space, whereas the local models are optimized for precision strictly within the vicinity of the current policy's trajectory and require only single-step accuracy.

E. Backward Models (Local)

For the backward model $f_\phi^{\leftarrow}$, we adopt the Recurrent State-Space Model (RSSM) architecture from DreamerV3 [15], which acts as a local latent dynamics and reward model. The model is first pretrained offline on the play data, with the global reward model predicting the reward ground truth. During online RL training, we continue fine-tuning $f_\phi^{\leftarrow}$ and its reward head using trajectories generated by the forward global model $f^{\rightarrow}$, ensuring that the RSSM maintains high local accuracy with respect to the current policy's conditional action distribution. To avoid differentiating through pixels (memory efficiency), an image encoder projects observations into a compact latent space, where $f_\phi^{\leftarrow}$ operates exclusively; notably, the policy gradients do not flow through the encoder. In the DMO framework, unlike DreamerV3 or other MBRL frameworks, the transition model in $f_\phi^{\leftarrow}$ requires only single-step accuracy, as Jacobians are evaluated at the encodings of precise forward-simulated images from the global model $f^{\rightarrow}$.

F. The Algorithm

Given the described global and local models, we present our complete procedure in Algorithm 1. To ensure stable policy updates, we extend the original DMO algorithm to incorporate the Soft Analytic Policy Optimization (SAPO) objective [36]. Specifically, we optimize the policy parameters $\boldsymbol{\theta}$ to maximize the entropy-regularized expected return:

$$\mathcal{L}_\pi^{\text{DMO-SAPO}}(\boldsymbol{\theta}) := \mathbb{E}_{\tau\sim\pi_\theta,f}\Bigg[\sum_{h=1}^{H-1}\gamma^h\big(r\left(s_h,a_h\right)+\alpha\mathcal{H}_\pi\left[a_h\mid s_h\right]\big) + \gamma^H V_{\boldsymbol{\psi}}^{\pi_\theta}\left(s_H\right)\Bigg], \tag{4}$$

where $\mathcal{H}_\pi[a_h \mid s_h]$ denotes the continuous Shannon entropy of the action distribution and α is the temperature parameter.

The critic network $V_{\boldsymbol{\psi}}^{\pi_\theta}$ is learned via SGD by minimizing the temporal difference error against λ-returns:

$$\mathcal{L}_V(\boldsymbol{\psi}) := \sum_{h=1}^{H-1}\left\|V_{\boldsymbol{\psi}}^{\pi_\theta}\left(s_h\right)-\hat{V}\left(s_h\right)\right\|_2^2, \tag{5}$$

where:

$$\begin{aligned} V_h\left(s_t\right) &:= \sum_{n=t}^{t+h-1}\gamma^{n-t}r\left(s_n,a_n\right)+\gamma^{t+h}V_{\boldsymbol{\psi}}^{\pi_\theta}\left(s_{t+h}\right) \\ \hat{V}\left(s_t\right) &:= (1-\lambda)\left[\sum_{h=1}^{H-t-1}\lambda^{h-1}V_h\left(s_t\right)\right] \\ &\quad +\lambda^{H-t-1}V_H\left(s_t\right) \end{aligned} \tag{6}$$

When unrolling the DreamerV4 global world model, we employ KV Caching to significantly reduce inference time. To initialize the context of the DreamerV4 world model during RL training, we do not start from a single frame. Instead, we sample a short history window of length $L_{\text{init}} = 32$ from the real trajectory dataset and pre-fill the key-value (KV) caches of the Transformer. This provides the model with sufficient temporal context to resolve potential ambiguities (e.g., momentary occlusions of the object) before the policy begins its interaction.

IV. Experiments

A. Setup

We conduct our study using two representative real-world examples: 1) a table-top manipulator (Flexiv Rizon-10S) with a fixed camera on the base overseeing the scene, and 2) a mobile robot (Unitree Go2 quadruped) with an onboard camera. The manipulator is controlled by a low-level end-effector pose controller that receives delta poses at 5Hz and applies them to the robot at 100Hz after interpolation. For the quadruped, the action is the 2D body linear and angular velocity fed at 5Hz into a low-level RL locomotion policy running at 50Hz ([2]).

B. Tasks

1) Push-T*:* This task requires pushing a T-shaped tool from a random pose to the center of the board in a predefined orientation. The contrastive reward model and the DIAMOND world model are used for this task and are both trained on a 4-hour dataset of unstructured play interactions collected via human teleoperation. Due to the global and unobstructed view of the scene, the world model for this task achieves accurate long-horizon rollouts with only 4 past frames as context (0.8s).

2) Ego-Centric Push Cube: Our second task aims to remove the global view (table-top camera) and push a cubical object into a goal as observed by a forward-looking camera. During interaction, the object can partially block the view or fall outside of the field of view, making the problem more complex to solve. In this example, we use the DreamerV4-inspired global world model, which is provided with a much longer context (96 frames, equivalent to 19.2 seconds), enabling it to handle occlusions and scenarios where the object

Algorithm 1 DMO-SAPO with global and local world models

Offline training of the diamond or dreamerV4 diffusion world model $f^{\rightarrow}$ and global reward world model r
Offline pretraining of local dreamerV3 world model $f_{\phi}^{\leftarrow}$ (including its reward head $\hat{R}_{\phi}$)
for epoch = 1 to N **do**
  # Backward model learning
  for model mini epoch **do**
    $(o, a, o') \sim \mathcal{B}$, where $\mathcal{B}$ is the replay buffer
    $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \alpha_{\boldsymbol{\phi}} \nabla_{\phi} \mathcal{L}_{f\leftarrow}(\boldsymbol{\phi})$, where $\mathcal{L}_{f\leftarrow}$ is the dreamerV3 model learning loss
  end for
  # Actor and critic learning
  $total\_reward \leftarrow 0$
  for h = 1 to H **do**
    $l_h \leftarrow \text{encoder}(o_h)$
    $a_h \leftarrow \pi_{\theta}(l_h)$
    $o_{h+1} \leftarrow f^{\rightarrow}(o_h, a_h)$ (where $f^{\rightarrow}$ is the diamond or the dreamerV4 diffusion world model)
    $r_h \leftarrow r(o_{h+1}, a_h)$ (where r is the global reward model, only used to learn the local reward and the value function)
    $\hat{R}_h \leftarrow \hat{R}_{\phi}(o_{h+1}, a_h)$
    $total\_reward \leftarrow total\_reward - \hat{R}_h$
  end for
  Compute $\mathcal{L}_{\pi}^{\text{DMO-SAPO}}(\boldsymbol{\theta})$ from $total\_reward$ and $V_{\boldsymbol{\psi}}^{\pi_{\theta}}(l_{H+1})$
  Compute $\mathcal{L}_V(\boldsymbol{\psi})$ from $(l_1, \ldots, l_H)$
  Use $\frac{\partial f_{\phi}^{\leftarrow}(l,a)}{\partial l}\Big|_{(l_{t+1}, a_{t+1})}$ and $\frac{\partial f_{\phi}^{\leftarrow}(l,a)}{\partial a}\Big|_{(l_{t+1}, a_{t+1})}$ to approximate $\frac{\partial f(o,a)}{\partial o}\Big|_{(o_{t+1}, a_{t+1})}$ and $\frac{\partial f(o,a)}{\partial a}\Big|_{(o_{t+1}, a_{t+1})}$ during the following backward pass.
  $\nabla_{\theta} \mathcal{L}_{\pi}^{\text{DMO-SAPO}}(\boldsymbol{\theta}) \leftarrow \text{backward}(\mathcal{L}_{\pi}^{\text{DMO-SAPO}}(\boldsymbol{\theta}))$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_{\boldsymbol{\theta}} \nabla_{\theta} \mathcal{L}_{\pi}^{\text{DMO-SAPO}}(\boldsymbol{\theta})$
  $\boldsymbol{\psi} \leftarrow \boldsymbol{\psi} + \alpha_{\boldsymbol{\psi}} \nabla_{\psi} \mathcal{L}_V(\boldsymbol{\psi})$
end for

exits the field of view. Both the world model and the reward head are trained on 12 hours of random interaction data, with demo/play intent labeled by the user via a button press.

C. Baselines and Ablations

We compare our method against two baselines: (1) PPO trained directly on image observations for both tasks, serving as a model-free RL benchmark; and (2) Action Chunking Transformer (ACT) [38], a state-of-the-art Behavior Cloning (BC) policy, trained on the demonstration subset of the Push Cube dataset to demonstrate the benefits of RL over pure behavior cloning.

To our knowledge, existing FoG-MBRL methods, such as the original DreamerV3 algorithm, are not suitable for working with diffusion-based learned models, as it is numerically problematic to take gradients through diffusion, so we did not include such methods as baselines. Indeed, relying solely on direct backpropagation through the global model is infeasible due to the model's much larger scale and the deep computation graph imposed by the multi-step diffusion process. Nevertheless, we test whether a DreamerV3-only world model would suffice to learn a good policy in the baseline called "No Diffusion".

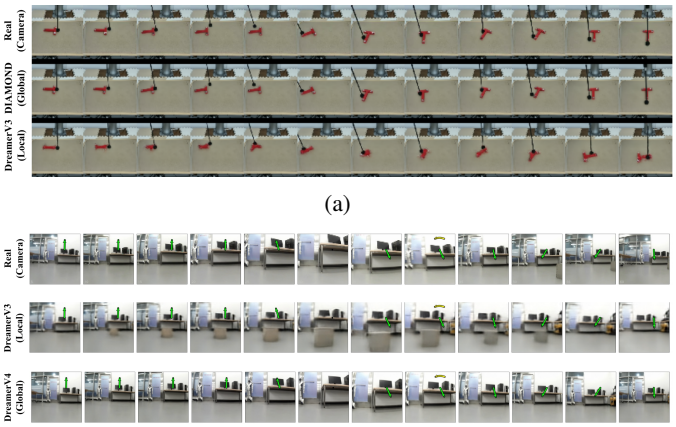


Fig. 2: (a) Unrolling of real and model-predicted trajectories comparing the real trajectory (top), the diamond diffusion trajectory (middle), and the DreamerV3 trajectory (bottom) at time steps $0, 5, 10, \ldots, 60$ (12 s). (b) Unrolling of real and model-predicted trajectories comparing the real trajectory (top), the DreamerV3 trajectory (middle), and the DreamerV4 diffusion trajectory (bottom) at time steps $0, 5, 10, \ldots, 60$ (12 s), with both models initialized with zero context, and the cube initially occluded. Note that the local model violates object permanency by spawning the cube throughout the rollout.

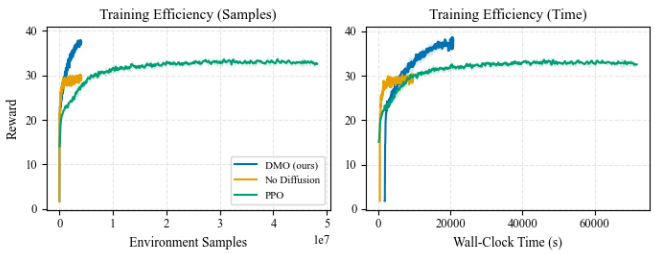
D. Quantitative Results

The results in Fig. 3 demonstrate that DMO achieves superior sample and time efficiency compared to PPO across both tasks. Notably, the time efficiency gains shown in Fig. 3 validate our hypothesis that first-order policy gradients are more suitable than zeroth-order policy gradients for computationally heavy world models (DMO vs PPO).

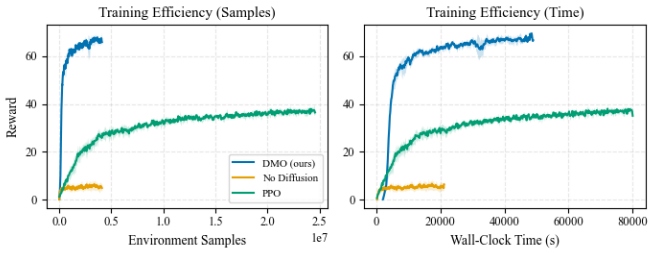
Furthermore, the poor performance of the "No Diffusion" ablation in Fig. 3 further underscores the necessity of a high-fidelity global world model for forward simulation of more complex tasks. As shown in Fig. 2a and Fig. 2b, DreamerV3 generalizes less effectively than the diffusion world model. Nevertheless, using the local model for tractable gradient computation and, by decoupling, exploiting the diffusion model's broad generalization, delivers significant improvements in both sample and time efficiency.

Table I reports the success rates for the Push-T task experiment (Fig. 4) and confirms the advantage of our approach compared to the baselines. The "No Diffusion" ablation did not lead to successful achievement of the task and PPO only managed to solve the task once.

It is worth noting that for the Push Cube task, the forward



(a)



(b)

Fig. 3: (a) Efficiency comparison on the Push-T task: sample efficiency of DMO (8M samples) versus PPO (40M samples) and the No Diffusion ablation (left), and corresponding time efficiency comparison (right). (b) Efficiency comparison on the Push Cube task: sample efficiency of DMO (4M samples) versus PPO (25M samples) and the No Diffusion ablation (left), and time efficiency comparison between DMO, PPO and No Diffusion (right).

simulation length for the PPO baseline had to be increased to 128 (as opposed to 64 for the DMO); otherwise, the PPO baseline falls into a suboptimal local solution where it repeatedly approaches and avoids the cube to collect rewards of 1 during each approach (approaching=demonstration behavior, thus +1 reward). In the meantime, DMO succeeded in both settings and demonstrates robustness to hyperparameter choices.

TABLE I: Success Rates Across Algorithms for Push-T (out of 10 Tries)

Algorithm	Successes / 10
DMO	9/10
PPO	1/10[1]
No Diffusion	0/10

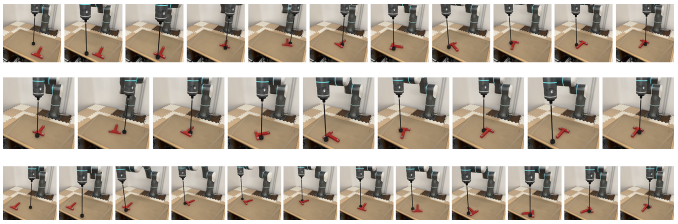


Fig. 4: Three real-robot Push-T trajectories executed by the policy learned with our approach.

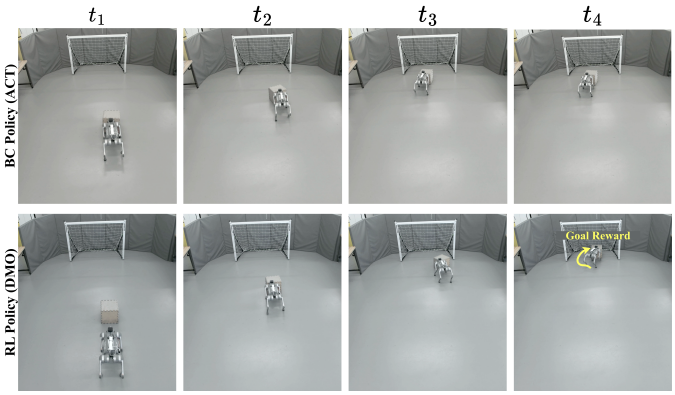


Fig. 5: Task completion comparison. **Top:** The Behavior Cloning (ACT) policy successfully approaches the cube but stops pushing before the object enters the goal, reflecting the sub-optimal demonstration distribution. **Bottom:** Our RL Policy (DMO) generalizes beyond the demonstrations, learning to push the cube fully into the net to maximize the task reward.

Regarding the ACT baseline on the Push Cube task, while it successfully approaches and pushes the cube, it consistently falls into a spurious equilibrium due to its myopic tendency to mimic the action. Specifically, the policy learns to stop in front of the goal, where during the demonstration phases, the operator exercises more caution upon approaching the gate (Fig. 5). This leads to a 0% success rate for the BC baseline, whereas the RL policy exhibits future-reward-seeking behavior, pushing the cube all the way into the goal (receiving a large terminal reward upon reaching the goal).

E. Qualitative Results

Our analysis of the Push Cube experiment reveals distinct behavioral differences across methods, particularly in object localization, task completion, and control smoothness (Table II).

First, as shown in Fig. 6, DMO exhibits an emergent active search strategy when the object is out of view. Specifically, the agent retreats to the goal area and executes a full 360° yaw rotation to scan the environment. In contrast, the BC policy (first row of Fig. 6) simply moves backward when the cube is lost. This blind retraction is brittle (if the cube is located behind the robot, the agent drags it farther away) and reflects the poor search strategy in the demonstration subset of the dataset. However, RL leverages random exploration during

TABLE II: Trajectory Quality Analysis. We report the mean metrics over $N = 3$ evaluation episodes for the Push Cube task. *Steps*: lower is better; *Straightness*: displacement/path length, ↑ is straighter; *Curvature*: yaw change per meter, ↓ is smoother.

Method	Steps to Success (↓)	Straightness Index (↑)	Curvature (rad/m) (↓)
DMO (Ours)	**91.3**	**0.881**	**0.565**
PPO	134.3	0.597	0.775
BC (ACT)	–[2]	0.447	0.588

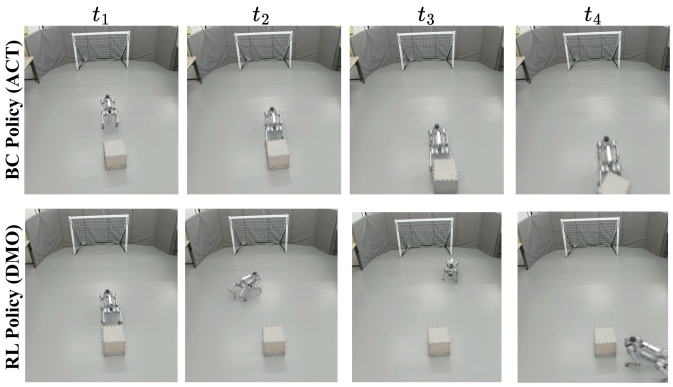


Fig. 6: Comparison of search strategies under partial observability. **Top:** When the cube is out of view, the BC policy (ACT) executes a naive backward retreat, which fails to locate the object if it is directly behind. **Bottom:** The DMO policy exhibits an emergent active search behavior, rotating to visually scan the environment and successfully localize the target.

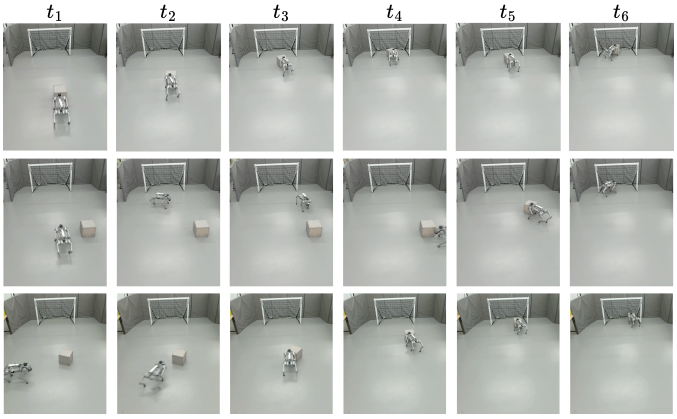


Fig. 7: Real-world execution of the DMO-learned policy for the Go2 Push Cube task when using a different low-level policy than seen during training. **Top:** Standard successful push trajectory. **Middle:** Emergent active search behavior; when the object is initially out of view (or lost), the agent retreats and scans the environment to localize the cube before pushing. **Bottom:** Successful approach and push from a different initial configuration.

training to discover more effective behaviors.

Second, as mentioned in the previous section and shown in Fig. 5, the BC (ACT) policy consistently stops before fully pushing the cube into the goal, whereas DMO successfully completes the task by leveraging sparse successful examples with high terminal rewards.

To further quantify the quality of the learned behaviors, we report two metrics: *Straightness Index* (the ratio of Euclidean displacement to total path length, where 1.0 is optimal) and *Curvature* (average yaw change per meter). Even in seeds where PPO successfully learns the task, the resulting policy exhibits highly oscillatory and erratic movement patterns, evidenced by a high curvature (0.775 rad/m) and low straightness (0.597). In contrast, DMO produces significantly smoother and more decisive trajectories, achieving a straightness index of $\mathbf{0.881}$ and the lowest curvature ($\mathbf{0.565}$ rad/m).

Finally, we evaluated robustness to dynamics distribution shifts by replacing the low-level locomotion controller with a variant with a different response behavior and gait style (no Raibert heuristics in reward). As shown in Fig. 7, without collecting new data or retraining the world models, the robot successfully pushes the cube into the goal. While PPO failed to generalize due to its higher entropy and less smooth action output, both DMO and BC maintained their performance. This result addresses a common critique of model-based methods—that models must be retrained for every specific robot configuration. Instead, this experiment demonstrates that the DMO framework correctly discovers stabilizing and smooth feedback strategies that are robust to variations in low-level dynamics. This finding implies that pretrained general-purpose world models may still produce useful policies for downstream tasks. A deeper investigation of this axis is the topic of our future research.

V. Conclusion

In summary, our framework bridges the limitations of simulator-centric RL by learning diffusion world models from real data, yielding visually realistic simulations that substantially reduce sim-to-real discrepancies in manipulation tasks. The decoupling mechanism leverages the comparative strengths of models—global diffusion for accurate, generalizable forward predictions and local latent models for tractable backward gradients—maintaining efficiency despite the high dimensionality of pixel inputs. Notably, our method enables learning policies directly from real robot data, using images as input and transfers zero-shot with a very high success rate on the real robot. Importantly, our experiments demonstrated practical task improvements when compared to pure behavior cloning. Our method provides a safe proxy for real-robot training, allowing scalable policy optimization without risking hardware damage or unsafe behaviors, paving the way for robust vision-based robotics in unstructured environments.

References

[1] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37, 2024.

[1] Under a relaxed success criterion (counting runs where the T briefly passes through the correct pose and position without stopping), PPO achieves 4/10 successes.

[2] ACT never reaches the terminal state and gets stuck at the goal entrance, and thus we cannot report on steps to success for it.

[2] Joseph Amigo, Rooholla Khorrambakht, Elliot Chane-Sane, Nicolas Mansard, and Ludovic Righetti. First order model-based rl through decoupled backpropagation. In *Conference on Robot Learning (CoRL)*, 2025.

[3] Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. *Advances in neural information processing systems*, 31, 2018.

[4] Elliot Chane-Sane, Joseph Amigo, Thomas Flayols, Ludovic Righetti, and Nicolas Mansard. Soloparkour: Constrained reinforcement learning for visual locomotion from privileged experience. In *Conference on Robot Learning (CoRL)*, 2024.

[5] Elliot Chane-Sane, Pierre-Alexandre Leziart, Thomas Flayols, Olivier Stasse, Philippe Souères, and Nicolas Mansard. Cat: Constraints as terminations for legged locomotion reinforcement learning. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13303–13310, 2024.

[6] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37, 2024.

[7] Xuxin Cheng, Kexin Shi, Ananye Agarwal, and Deepak Pathak. Extreme parkour with legged robots. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11443–11450, 2024.

[8] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

[9] Ignasi Clavera, Violet Fu, and Pieter Abbeel. Model-augmented actor-critic: Backpropagating through paths. In *International Conference on Learning Representations (ICLR)*, 2020.

[10] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.

[11] Ignat Georgiev, Varun Giridhar, Nicklas Hansen, and Animesh Garg. Pwm: Policy learning with multi-task world models. *arXiv preprint arXiv:2407.02466*, 2024.

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

[13] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2019.

[14] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations (ICLR)*, 2021.

[15] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025.

[16] Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527*, 2025.

[17] Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527*, 2025.

[18] Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning (ICML)*, 2022.

[19] Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations (ICLR)*, 2024.

[20] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. In *International Conference on Learning Representations (ICLR)*, 2020.

[21] David Hoeller, Nikita Rudin, Dhionis Sako, and Marco Hutter. Anymal parkour: Learning agile navigation for quadrupedal robots. *Science Robotics*, 9(88), 2024.

[22] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, 32, 2019.

[23] Armand Jordana, Jianghan Zhang, Joseph Amigo, and Ludovic Righetti. An introduction to zero-order optimization techniques for robotics. *arXiv preprint arXiv:2506.22087*, 2025.

[24] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35, 2022.

[25] Rooholla Khorrambakht, Joaquim Ortiz-Haro, Joseph Amigo, Omar Mostafa, Daniel Dugas, Franziska Meier, and Ludovic Righetti. Worldplanner: Monte carlo tree search and mpc with action-conditioned visual world models, 2025.

[26] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[27] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations (ICLR)*, 2018.

[28] Nathan Lambert, Kristofer Pister, and Roberto Calandra. Investigating compounding prediction errors in learned dynamics models. *arXiv preprint arXiv:2203.09637*, 2022.

[29] Chenhao Li, Andreas Krause, and Marco Hutter. Uncertainty-aware robotic world model makes offline

model-based reinforcement learning work on real robots, 2025.

[30] Yazhe Niu, Yuan Pu, Zhenjie Yang, Xueyan Li, Tong Zhou, Jiyuan Ren, Shuai Hu, Hongsheng Li, and Yu Liu. Lightzero: A unified benchmark for monte carlo tree search in general sequential decision scenarios. *Advances in Neural Information Processing Systems*, 36, 2024.

[31] Yuan Pu, Yazhe Niu, Jiyuan Ren, Zhenjie Yang, Hongsheng Li, and Yu Liu. Unizero: Generalized and efficient planning with scalable latent world models. *arXiv preprint arXiv:2406.10667*, 2024.

[32] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[33] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[34] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pages 2226–2240. PMLR, 2023.

[35] Chenjun Xiao, Yifan Wu, Chen Ma, Dale Schuurmans, and Martin Müller. Learning to combat compounding-error in model-based reinforcement learning, 2019.

[36] Eliot Xing, Vernon Luk, and Jean Oh. Stabilizing reinforcement learning in differentiable multiphysics simulation. *International Conference on Learning Representations (ICLR)*, 2025.

[37] Daniel Yang, Davin Tjia, Jacob Berg, Dima Damen, Pulkit Agrawal, and Abhishek Gupta. Rank2reward: Learning shaped reward functions from passive video. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2806–2813. IEEE, 2024.

[38] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023.

[39] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)*, pages 737–744. IEEE, 2020.

[40] Ziwen Zhuang, Zipeng Fu, Jianren Wang, Christopher Atkeson, Soeren Schwertfeger, Chelsea Finn, and Hang Zhao. Robot parkour learning. *arXiv preprint arXiv:2309.05665*, 2023.