Make the Pertinent Salient: Task-Relevant Reconstruction for Visual Control with Distractions

Anonymous authors
Paper under double-blind review

Keywords: Visual Control, Robust Representation Learning, Model-Based Reinforcement Learning.

Summary

Model-Based Reinforcement Learning (MBRL) has shown promise in visual control tasks due to its data efficiency. However, training MBRL agents to develop generalizable perception remains challenging, especially amid visual distractions that introduce noise in representation learning. We introduce Segmentation Dreamer (SD), a framework that facilitates representation learning in MBRL by incorporating a novel auxiliary task. Assuming that task-relevant components in images can be easily identified with prior knowledge in a given task, SD uses segmentation masks on image observations to reconstruct only task-relevant regions, reducing representation complexity. SD can leverage either ground-truth masks available in simulation or potentially imperfect segmentation foundation models. The latter is further improved by selectively applying the image reconstruction loss to mitigate misleading learning signals from mask prediction errors. In modified DeepMind Control suite and Meta-World tasks with added visual distractions, SD achieves significantly better sample efficiency and greater final performance than prior work and is especially effective in sparse reward tasks that had been unsolvable by prior work. We also validate its effectiveness in a real-world robotic lane-following task when training with intentional distractions to achieve zero-shot transfer.

Contribution(s)

1. This paper introduces a novel auxiliary task in model-based reinforcement learning (MBRL) to enhance representation learning in visually distracting environments. Our approach reconstructs control-relevant components while filtering out distractions, ensuring that latent embeddings focus on essential features.
   Context: While our method requires prior knowledge of task-relevant components, identifying these components is typically straightforward for practitioners in many robotics applications. Prior work using reconstruction-free auxiliary tasks relies on large amounts of data to infer important features, making them less sample-efficient.
2. This paper integrates segmentation foundation models to guide feature learning in visual control through task-relevant reconstruction targets, without incurring extra test-time overhead and while improving robustness to segmentation errors. This demonstrates an effective way to harness advances in computer vision for visual control tasks.
   Context: Prior approaches typically use segmentation models for input preprocessing, which adds deployment overhead and increases sensitivity to segmentation errors.
3. Our method learns effective visual control policies in environments with distractions, demonstrating success in DMC, where locomotion control requires handling contact dynamics; Meta-World, which involves robotic manipulation, occlusions, and multi-object interactions; and DuckieTown, where transferring lane-following behavior from simulation to reality must account for diverse perturbations, including foreground distractions.
   Context: Our method is sample-efficient, achieves higher final performance, and is the only method capable of learning with sparse rewards in DMC.

Make the Pertinent Salient: Task-Relevant Reconstruction for Visual Control with Distractions

Anonymous authors
Paper under double-blind review

Abstract

Model-Based Reinforcement Learning (MBRL) has shown promise in visual control tasks due to its data efficiency. However, training MBRL agents to develop generalizable perception remains challenging, especially amid visual distractions that introduce noise in representation learning. We introduce Segmentation Dreamer (SD), a framework that facilitates representation learning in MBRL by incorporating a novel auxiliary task. Assuming that task-relevant components in images can be easily identified with prior knowledge in a given task, SD uses segmentation masks on image observations to reconstruct only task-relevant regions, reducing representation complexity. SD can leverage either ground-truth masks available in simulation or potentially imperfect segmentation foundation models. The latter is further improved by selectively applying the image reconstruction loss to mitigate misleading learning signals from mask prediction errors. In modified DeepMind Control suite and Meta-World tasks with added visual distractions, SD achieves significantly better sample efficiency and greater final performance than prior work and is especially effective in sparse reward tasks that had been unsolvable by prior work. We also validate its effectiveness in a real-world robotic lane-following task when training with intentional distractions to achieve zero-shot transfer.

1 Introduction

Recent advances in model-based reinforcement learning (MBRL) (Sutton, 1991; Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020; Hansen et al., 2022; 2023) have made it a powerful tool for learning control policies, achieving high sample efficiency. Among these advancements, the DREAMER family (Hafner et al., 2020; 2021; 2023) stands out as seminal work, demonstrating strong performance across diverse visual control environments. This success is driven by a close cooperation between a world model and an actor–critic agent. The world model learns to emulate the environment's forward dynamics and reward function in a latent state space, and the agent is trained by interacting with this world model in place of the original environment.

Under this framework, accurate reward prediction is all we should sufficiently require for agent training. However, learning representations solely from reward signals is inherently challenging due to their limited expressiveness and high variance (Hafner et al., 2020; Jaderberg et al., 2017). To address this, DREAMER employs image reconstruction as an auxiliary task in world model training to facilitate representation learning. In environments with little distraction, image reconstruction proves effective by delivering rich feature-learning signals derived from pixels. However, in the presence of distractions, the image reconstruction task pushes the encoder to retain all image information, regardless of its task relevance. Including such information in the latent space complicates dynamics modeling and degrades sample efficiency by wasting model capacity and drowning the relevant signal in noise.

Distractions are prevalent in real-world visual control tasks. A robot operating in a cluttered environment such as a warehouse may perceive much task-irrelevant information that it needs to ignore.

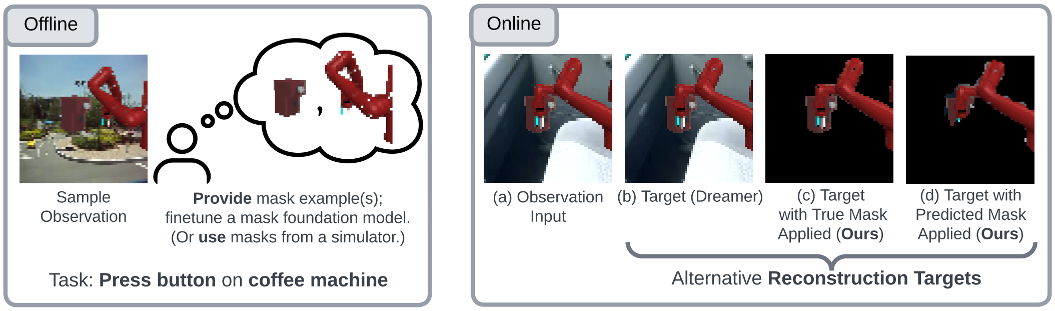

Offline
Sample Observation
Provide mask example(s); finetune a mask foundation model. (Or use masks from a simulator.)
Task: Press button on coffee machine
Online
(a) Observation Input
(b) Target (Dreamer)
(c) Target with True Mask Applied (Ours)
(d) Target with Predicted Mask Applied (Ours)
Alternative Reconstruction Targets

Figure 1: *Left*: Providing mask example(s) and fine-tuning a mask model, or instrumenting a simulator, to obtain masks. *Right*: An input observation in a distracting Meta-World with three alternative auxiliary task targets. Moving scenes in the background are considered distractions. (b) Observations including task-irrelevant information, disturbing world-model training. (c) and (d) Segmentation of task-relevant components using, respectively, a ground-truth mask and an approximate mask generated by segmentation models.

When training with domain randomization for added policy robustness, task-irrelevant information is actively added and must be denoised. Prior approaches (Zhang et al., 2021; Nguyen et al., 2021; Deng et al., 2022; Fu et al., 2021; Bharadhwaj et al., 2022) address the noisy reconstruction problem by devising reconstruction-free auxiliary tasks, such as contrastive learning (Chen et al., 2020). However, they often suffer from sample inefficiency, requiring many trajectories to isolate the task-relevant information that needs to be encoded. This challenge is exacerbated in sparse-reward environments, where the signal for task relevance is very weak. Additionally, working with small objects, which is common in object manipulation tasks, poses difficulties for these methods because those objects contribute less to loss functions and are easily overlooked without special attention (Seo et al., 2022).

Inspired by these problems, we address the following question in this paper: *How can we help world models learn task-relevant representations more efficiently?* Our proposed solution takes advantage of the observation that identifying task-relevant components within images is often straightforward with some domain knowledge. For instance, in a robotic manipulation task, the objects to manipulate and the robot arm are such task-relevant components, as shown in Fig. 1 (Left). Given this assumption, we introduce a simple yet effective alternative auxiliary task to reconstruct only the task-related components of image observations.

We accomplish this by using segmentation masks of task-related objects which are easily accessible in simulations. Specifically, we replace Dreamer's auxiliary task to reconstruct raw RGB image observations (Fig. 1b) with an alternative task to reconstruct images with a *task-relevant mask* applied to them (Fig. 1c). By doing this, the world model can learn features from a rich pixel-reconstruction loss signal without being hindered by the noise of visual distractions. As long as task-relevance can feasibly be encoded in segmentation-mask format, which is common in many object-centric and robotics domains, our method can be used to improve the efficiency of world model training in distracting environments.

Unlike previous work that incorporates segmentation masks as inputs in reinforcement learning (RL) (James et al., 2019; So et al., 2022; Zhang et al., 2025), we use masks solely in an auxiliary task to improve representation learning. This approach offers two advantages. First, segmentation masks are only required during training. Our method still operates on the original (potentially distracting) images, so masks are unnecessary at test time, improving computational efficiency during deployment. Second, the masks do not need to be perfect; as long as they guide feature learning to be informative for the downstream task, approximate masks can replace ground-truth masks, increasing practicality.

To this end, we propose training with our auxiliary task using segmentation estimates, enabling learning in scenarios where no ground-truth (GT) masks are available. Building on recent advances in segmentation foundation models (Kirillov et al., 2023; Zhang et al., 2023; Xie et al., 2021), we fine-tune these models with a small amount of annotated training data to generate pseudo-labels for the auxiliary task (Fig. 1d). While the performance with segmentation estimates is strong without further modification, we find that the training can sometimes be destabilized due to incorrect learning signals from segmentation prediction errors. To enhance robustness, we identify pixels where the foundation model's mask prediction disagrees with a second mask prediction given by our world model. We then exclude these pixels from the RGB reconstruction loss, preventing training on potentially incorrect targets (Section 4.3).

We additionally demonstrate our method's effectiveness in cases where ground-truth mask are available but only during training, such as when training in simulation for zero-shot deployment on a real robot. In such cases, methods like domain randomization (Tobin et al., 2017) can be employed during training to introduce visual distractions and promote test-time generalization to unseen real environment appearances. Using ground-truth masks provided by the simulation, we show that decoding only task-relevant information dramatically improves the world model's training efficiency and generalization on a real-robot lane-following task.

We evaluate our method on various robotics benchmarks, including DeepMind Control Suite (Tassa et al., 2018) and Meta-World (Yu et al., 2019), perturbing both with visual distractions. We show that our method for reconstructing masked RGB targets using the ground-truth masks in the presence of distractions can reach the same level of performance as training in *original* environment with no distractions added. Our method for training with approximate masks also shows impressive performance, often matching the performance of the ground-truth mask variant. In both benchmarks, our approximate-mask method achieves higher sample efficiency and superior test returns compared to previous approaches. Notably, this is accomplished with very few task-specific mask example data points (1, 5, or 10 used for fine-tuning), with much of its strength coming from the power of segmentation foundation models. Furthermore, our method proves particularly effective in sparse reward environments and those involving small objects, where prior approaches often struggle.

Finally, in the robot lane-following task, we demonstrate our method's effectiveness in simulation-to-real training by decoding only task-relevant components of image observations, promoting more efficient simulation training and better zero-shot generalization to the real world environment.

2 Related Work

Model-Based RL for Distracting Visual Environments. Recent advances in MBRL have enabled efficient learning from image observations (Finn & Levine, 2017; Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020; 2021; 2023; Schrittwieser et al., 2020; Hansen et al., 2022; 2023). However, learning robust perceptual representations in the presence of distractions remains challenging. Some approaches use non-reconstructive representation learning methods (Nguyen et al., 2021; Deng et al., 2022), such as contrastive (Chen et al., 2020) and prototypical learning (Caron et al., 2020). However, features learned with these methods do not necessarily involve task-related content since they do not explicitly consider task-relevance in feature learning. Other works introduce auxiliary objectives to explicitly incorporate downstream task information, such as DBC (Zhang et al., 2021), which uses a bisimulation metric (Ferns et al., 2011), and TIA (Fu et al., 2021), which explicitly separates task-relevant and irrelevant branches to distinguish reward-correlated visual features from distractions. More recent methods exploit inductive biases like predictability (Zhu et al., 2023) and controllability (Wang et al., 2022; Bharadhwaj et al., 2022) but often require extensive sampling to infer task-relevant content. Notably, solving sparse reward environments with distractions remains an open problem. In contrast, our work proposes to leverage domain knowledge via image masks to directly guide task-relevant representation learning and improve sample efficiency by reducing the complexity of learned representations. While model-free RL has explored robust representation learning (Laskin et al., 2020; Kostrikov et al., 2021; Yarats et al., 2021; Hansen et al., 2021; Hansen & Wang, 2021; Nair et al., 2022; Zhang et al., 2019), MBRL remains superior in sample efficiency and performance for visual control, making it our primary focus for comparison.

Segmentation for RL. Segmentation models (He et al., 2017; Redmon et al., 2016) have been widely used across many downstream tasks, including RL (Kirillov et al., 2023; Anantharaman et al., 2018; Yuan et al., 2018; James et al., 2019; So et al., 2022). Recent advances in segmentation foundation models (Zhang et al., 2023; Xie et al., 2021) enable streamlined and accelerated adaptation to new domains with one/few-shot learning. In RL, a common approach to leveraging segmentation models involves converting input RGB images into segmentation masks (James et al., 2019; So et al., 2022) or latent representations (Zhang et al., 2025), improving robustness to complex scenes and domain randomization. However, this increases computational overhead and the risk of failure if segmentation models malfunction. Our method instead leverages segmentation masks as an auxiliary task, removing the reliance on segmentation models at deployment while improving test-time performance. While FOCUS (Ferraro et al., 2023) also uses masked input as an auxiliary target, it focuses on learning disentangled representations rather than handling distractions. Moreover, it provides only preliminary results with segmentation models without analyzing their impact on downstream RL tasks.

3 Preliminaries

We consider a partially observable Markov decision process (POMDP) formalized as a tuple $(\mathcal{S}, \Omega, \mathcal{A}, \mathcal{T}, \mathcal{O}, p_0, \mathcal{R}, \gamma)$, consisting of states $s \in \mathcal{S}$, observations $o \in \Omega$, actions $a \in \mathcal{A}$, state transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, observation function $\mathcal{O} : \mathcal{S} \to \Omega$, initial state distribution p_0, reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and discount factor γ. At time t, the agent does not have access to actual world state s_t, but to the observation $o_t = \mathcal{O}(s_t)$, which in this paper we consider to be a high-dimensional image. Our objective is to learn a policy $\pi(a_t|o_{\leq t}, a_{<t})$ that achieves high expected discounted cumulative rewards $\mathbb{E}[\sum_t \gamma^t r_t]$, with $r_t = \mathcal{R}(s_t, a_t)$ and the expectation over the joint stochastic process induced by the environment and the policy.

Dreamer (Hafner et al., 2020; 2021; 2023) is a broadly applicable MBRL method in which a world model learns to represent environment dynamics in a latent state space $(h, z) \in \mathcal{H} \times \mathcal{Z}$, consisting of deterministic and stochastic components respectively, from which rewards, observations, and future latent states can be decoded. The components of the world model are:

$$\begin{aligned}
&\text{Sequence model:} && h_t = f_\phi(h_{t-1}, z_{t-1}, a_{t-1}) \\
&\text{Observation encoder:} && z_t \sim q_\phi(z_t|h_t, o_t) \\
&\text{Dynamics predictor:} && \hat{z}_t \sim p_\phi(\hat{z}_t|h_t) \\
&\text{Reward predictor:} && \hat{r}_t \sim p_\phi(\hat{r}_t|h_t, z_t) \\
&\text{Continuation predictor:} && \hat{c}_t \sim p_\phi(\hat{c}_t|h_t, z_t) \\
&\text{Observation decoder:} && \hat{o}_t \sim p_\phi(\hat{o}_t|h_t, z_t),
\end{aligned} \tag{1}$$

where the encoder maps observations o_t into a latent representation, the dynamics model emulates the transition distribution in latent state space, the reward and continuation models respectively predict rewards and episode termination, and the observation decoder reconstructs the input. The concatenation of h_t and z_t, *i.e.* $x_t = [h_t; z_t]$, serves as the model state. Given a starting state, an actor–critic agent is trained inside the world model by rolling out latent-state trajectories. The world model itself is trained by optimizing a weighted combination of three losses:

$$\mathcal{L}(\phi) \doteq \mathbb{E}_{q_\phi}\left[\sum_{t=1}^{T}(\beta_{\text{pred}}\mathcal{L}_{\text{pred}}(\phi) + \beta_{\text{dyn}}\mathcal{L}_{\text{dyn}}(\phi) + \beta_{\text{rep}}\mathcal{L}_{\text{rep}}(\phi))\right] \tag{2}$$

$$\mathcal{L}_{\text{pred}}(\phi) \doteq -\ln p_\phi(o_t|z_t, h_t) - \ln p_\phi(r_t|z_t, h_t) - \ln p_\phi(c_t|z_t, h_t) \tag{3}$$

$$\mathcal{L}_{\text{dyn}}(\phi) \doteq \max(1, \text{KL}[\llbracket q_\phi(z_t|h_t, o_t)\rrbracket \| \quad p_\phi(\hat{z}_t|h_t))]) \tag{4}$$

$$\mathcal{L}_{\text{rep}}(\phi) \doteq \max(1, \text{KL}[q_\phi(z_t|h_t, o_t) \quad \| \quad \llbracket p_\phi(\hat{z}_t|h_t))\rrbracket]), \tag{5}$$

where $\llbracket \cdot \rrbracket$ denotes where gradients are stopped from backpropagating to the expression in brackets.

Critically, the first component of $\mathcal{L}_{\text{pred}}$ for reconstructing observations from world model states is leveraged as a powerful heuristic to shape latent features. Under the assumption that observations primarily contain task-relevant information, this objective is likely to encourage the latent state to retain information critical for the RL agent. However, the opposite can also be true. If observations are dominated by task-irrelevant information, the latent dynamics may become more complex by incorporating features impertinent to decision-making. This can lead to wasted capacity in the latent state representation (Lambert et al., 2020), drown the supervision signal in noise, and reduce the sample efficiency.

Problem Setup. We consider environments where the latter case is true and observations contain a large number of spurious variations (Zhu et al., 2023). Concretely, we consider some features of states $s_t \in \mathcal{S}$ to be irrelevant for the control task. We assume that states s_t can be decomposed into task-relevant components $s_t^+ \in S^+$ and task-irrelevant components $s_t^- \in S^-$ such that $s_t = (s_t^+, s_t^-) \in \mathcal{S} = \mathcal{S}^+ \times \mathcal{S}^-$. We follow prior work (Zhu et al., 2023; Fu et al., 2021; Bharadhwaj et al., 2022) in visual control under distraction and assume that (1) the reward is a function only of the task-relevant component, *i.e.* $\mathcal{R} : \mathcal{S}^+ \times \mathcal{A} \rightarrow \mathbb{R}$; and (2) the forward dynamics of the task-relevant part only depends on itself, $s_{t+1}^+ \sim \mathcal{T}(s_{t+1}^+ | s_t^+, a_t)$. Note that observations o_t are a function of both s_t^+ and s_t^-, thus we have $\mathcal{O} : \mathcal{S}^+ \times \mathcal{S}^- \rightarrow \Omega$.

Our goal is to learn effective latent representations $[h_t; z_t]$ for task control. Ideally, this would mean that the world model will only encode and simulate task-relevant state components s_t^+ in its latent space without modeling unnecessary information in s_t^-. To learn features pertaining to s_t^+, image reconstruction can provide a rich and direct learning signal, but only when observation information about s_t^+ is not drowned out by other information from s_t^-. To overcome this pitfall, we propose to apply a heuristic filter to reconstruction targets o_t with the criteria that it minimizes irrelevant information pertaining to s_t^- while keeping task-relevant information about s_t^+.

4 Method

We build on DREAMER-V3 (Hafner et al., 2023) to explicitly model s_t^+ while attempting to avoid encoding information about s_t^-. In Section 4.1, we describe how we accomplish this by using domain knowledge to apply a task-relevance mask to observation reconstruction targets. In Section 4.2 we describe how we leverage segmentation mask foundation models to provide approximate masks over task-relevant observation components. Finally, in Section 4.3, we propose a modified decoder architecture and objective to mitigate noisy learning signals from incorrect mask predictions.

4.1 Using Segmentation Masks to Filter Image Targets

We first introduce our main assumption, that the task-relevant components of image observations are easily identifiable with domain knowledge. In many real scenarios, it is often straightforward for a practitioner to know what the task-related parts of an image are, *e.g.* objects necessary for achieving a goal in object manipulation tasks. With this assumption, we propose a new reconstruction-based auxiliary task that leverages domain knowledge of task-relevant regions. Instead of reconstructing the raw image observations (Fig. 1b) which may contain task-irrelevant distractions, we apply a heuristic task-relevance segmentation mask over the image observation (Fig. 1c) to exclusively reconstruct components of the image that are pertinent to control.

Since our new masked reconstruction target should contain only image regions relevant for achieving the downstream task, our world model learns latent representations where a larger portion of the features are useful to the RL agent. By explicitly excluding task-irrelevant observation components, the latent dynamics also becomes simpler and more sample-efficient to learn than the original (more complex, higher variance) dynamics on unfiltered observations. In simulations, ground-truth masks of relevant observation components are often easily accessible, e.g., in MuJoCo (Todorov et al., 2012) through added calls to the simulator API. We term the method trained with our proposed replacement auxiliary task as Segmentation Dreamer (SD) and call the version trained with ground-truth masks SD$^{\text{GT}}$.

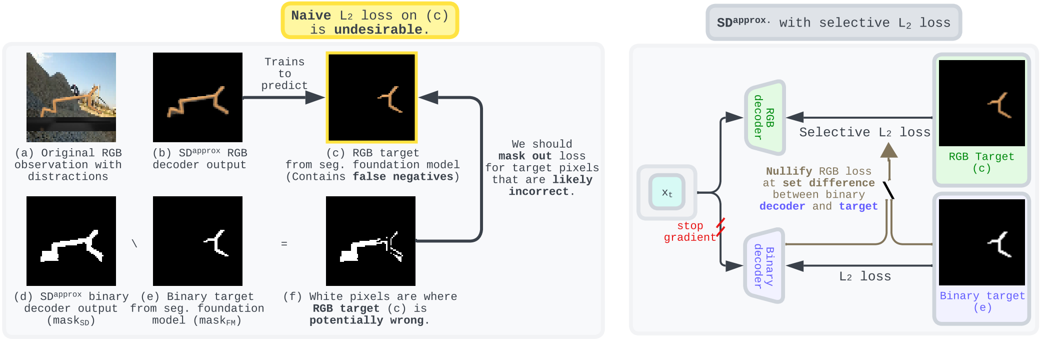

Naive L₂ loss on (c) is **undesirable**.
(a) Original RGB observation with distractions
(b) SDapprox RGB decoder output
Trains to predict
(c) RGB target from seg. foundation model (Contains **false negatives**)
We should **mask out** loss for target pixels that are **likely incorrect**.
(d) SDapprox binary decoder output (maskSD)
\
(e) Binary target from seg. foundation model (maskFM)
=
(f) White pixels are where **RGB target** (c) is **potentially wrong**.
SDapprox. with selective L₂ loss
xt
stop gradient
RGB decoder
Selective L₂ loss
RGB Target (c)
Nullify RGB loss at **set difference** between binary **decoder** and **target**
Binary decoder
L₂ loss
Binary target (e)

Figure 2: **Filtering L_2 loss to avoid training on false negatives in RGB labels.** *Left*: Estimated pixel locations (f) where the RGB target (c) is likely incorrectly masked out by the segmentation model (e). *Right*: A world model equipped with two decoders, one for reconstructing task-relevant masked RGB images and the other for binary masks, the targets for which are generated by a segmentation model. RGB L_2 loss is selectively masked by the set difference between (d) and (e). Latent representations (x_t) in the world model are subjected to the training signal only from the RGB branch. The binary branch is only utilized for selective L_2 loss.

4.2 Leveraging Approximate Masks

A simulator with ground-truth masks for task-relevant regions is not always available. For such cases where only RGB images are available from the environment, we propose to fine-tune a segmentation mask foundation model to our domain and integrate its predictions into the SD training pipeline. Below, we describe our method for training with approximate task-relevance masks, termed SD$^{\text{approx.}}$.

As an offline process before training the world model, we fine-tune a segmentation model with a small number of example RGB images and their mask annotations that indicate task-relevant image regions. Recent advances in segmentation foundation models allow us to adapt a domain-specific mask model with very few examples. For our experiments, we use the Personalized SAM (PerSAM) (Zhang et al., 2023) using one-shot adaptation and SegFormer (Xie et al., 2021) fine-tuned with 5 and 10 examples. For the sake of controlled and reproducible evaluation, we extract these RGB and mask training pairs from simulators, however, the sample size is small enough for expert annotation. Also, although we use these specific foundation models, our method should also be compatible with *any* semantic masking method. Additional details on fine-tuning these models are provided in Appendix K. Once fine-tuning is complete, we incorporate the segmentation model into the SD pipeline to create pseudo-labels for our proposed auxiliary task.

4.3 Learning in the Presence of Mask Errors

Although foundation segmentation models generalize well to new scenarios (e.g., different poses, occlusions), prediction errors are inevitable (Fig. 1d). Since each frame is processed independently, segmentation predictions may flicker across trajectories. False negatives in task relevance are particularly detrimental when using naive L_2 loss for image reconstruction, as missing relevant scene elements in reconstruction targets can lead the encoder to learn incomplete representation, discarding essential task-related information. This variability disrupts the learning of accurate representations and dynamics in the world model.

Despite noisy targets, neural networks can self-correct if most labels are accurate (Han et al., 2018). Additionally, DREAMER's use of GRUs (Cho et al., 2014) provides temporal consistency even with flickering targets. However, as shown in Fig. 2 (b)&(c), it's undesirable to propagate gradients from regions where the observation has been incorrectly masked out. Allowing gradients from these regions provides misleading signals and reinforces errors rather than correcting them.. If we could identify the incorrect regions in the reconstruction target, we could nullify the decoder's L_2 loss there—a technique we call selective L_2 loss.

Since we cannot directly identify regions where the RGB target is incorrectly masked due to false negatives, we estimate them. Preliminary experiments show that a binary mask decoder from world model states (as an added auxiliary task) can be less prone to transient false negatives, unlike RGB prediction, which tends to memorize noisy labels. Therefore, we propose training a world model with two reconstruction tasks (Fig. 2, right): one decoding masked RGB images and the other predicting task-relevance binary masks. Both use the foundation model's binary mask, mask_{FM}, to construct targets. The RGB branch decodes masked RGB images, while the binary branch predicts mask_{FM}. We denote the binary masks produced by the world model as mask_{SD}, where pixels labeled *true* (or white) indicate task relevance.

To avoid training on incorrectly masked-out regions, we estimate where mask_{FM} may be falsely negative by finding disagreements with mask_{SD}. Specifically, we selectively nullify RGB decoder L_2 loss for regions marked false in mask_{FM} but predicted true in mask_{SD}. This prevents training on potentially falsely masked-out pixels still considered task-relevant by a second predictor. Formally, the mask for selective L_2 loss is the set difference between true pixel locations in mask_{SD} and mask_{FM}:

$$\text{pixel}_{\text{MaskOut}} = \text{pixel}_{\text{SD}} \setminus \text{pixel}_{\text{FM}} \tag{6}$$

where $\text{pixel}_{\text{MaskOut}}$ indicates pixels to nullify loss at, and pixel_{SD} and pixel_{FM} are pixels marked true in mask_{SD} and mask_{FM}, respectively.

Fig. 2 (d–f) shows examples of mask_{SD}, mask_{FM}, and $\text{pixel}_{\text{MaskOut}}$. See Appendix L for details on obtaining mask_{SD}. Our experiments indicate that selective L_2 loss effectively overcomes noisy segmentation labels and improves downstream agent performance.

Lastly, we observe better performance by blocking gradients from the binary mask decoding objective from propagating into the world model, so we apply a stop gradient to the mask decoder head inputs (see Appendix F for ablations).

5 Experiments

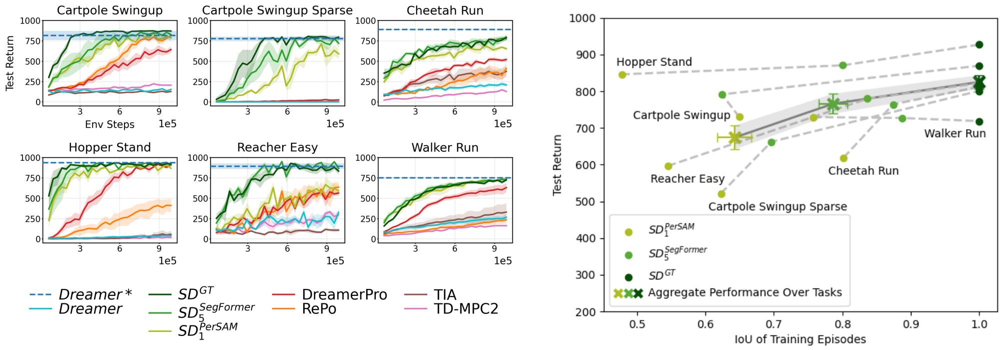
We evaluate our method on visual robotic control tasks from the DeepMind Control Suite (DMC) (Tassa et al., 2018) and Meta-World (Yu et al., 2019). Since these benchmarks feature simple backgrounds with minimal distractions, we introduce visual distractions by replacing the backgrounds with random videos from the 'driving car' class in the Kinetics 400 dataset (Kay et al., 2017), following prior work (Zhang et al., 2021; Nguyen et al., 2021; Deng et al., 2022). Details on the environment setup and task visualizations are provided in Appendices G and A. In evaluation, we roll out policies over 10 episodes and compute the average episode return. Unless otherwise specified, we report the mean and standard error of the mean (SEM) over four independent runs with different random seeds. All experiments use the default DREAMER-V3 hyperparameters. We also evaluate our method in a real-world lane-following task, demonstrating that SD can learn a policy that generalizes to unseen appearances at deployment.

5.1 DMC Experiments

We evaluate SD on six DMC tasks with varying contact dynamics, degrees of freedom, and reward sparsity. For each task, models are trained for 1M environment steps generated by 500K policy decision steps with an action repeat of 2.

5.1.1 Comparison with DREAMER

We compare our methods, SD^{GT} and $\text{SD}^{\text{approx.}}$, to the base DREAMER (Hafner et al., 2023) method. Here, $\text{SD}^{\text{approx.}}$ is denoted as $\text{SD}^{\text{FM}}_{N}$, specifying the segmentation model used (FM) and the number of fine-tuning examples (N). All methods are trained in distracting environments, except for the DREAMER* baseline, which is trained in the original environment without visual distractions. In



(a) Environment Steps vs. Expected Test Return

(b) IoU during Training vs. Expected Test Return

Figure 3: (a) **Learning curves on six visual control tasks from DMC.** Every method but DREAMER* is trained on distracting environments. All curves show the mean over 4 seeds with the standard error of the mean (SEM) shaded. (b) Segmentation quality during training vs. downstream task performance. Best viewed in color.

most cases, we consider DREAMER* as an upper bound for methods trained with distractions. Similarly, SD$^{\text{GT}}$ serves as an upper bound for SD$^{\text{approx.}}$, with the performance gap expected to decrease in the future as segmentation quality improves.

As shown in Fig. 3a, DREAMER fails across all tasks due to task-irrelevant information in RGB reconstruction targets, which wastes latent capacity and complicates dynamics learning. In contrast, SD$^{\text{GT}}$ achieves test returns comparable to DREAMER* by focusing on reconstructing essential features and ignoring irrelevant components. Interestingly, SD$^{\text{GT}}$ outperforms DREAMER* in Cartpole Swingup, possibly because the original environment still contains small distractions (e.g., moving dots) that DREAMER* has to model.

A limitation of SD is its reliance on acccurate and correct prior knowledge to select task-relevant components. In Cheetah Run, SD$^{\text{GT}}$ underperforms compared to DREAMER*, likely because we only include the cheetah's body in the mask, excluding the ground plate, which may be important for contact dynamics. Visual examples and further experiments are in Appendices A and B.

For SD$^{\text{approx.}}$, we test with two foundation models: PerSAM adapted with one RGB example and its GT mask, and SegFormer adapted with five such examples. Despite slower convergence due to noisier targets, both SD$_1^{\text{PerSAM}}$ and SD$_5^{\text{SegFormer}}$ achieve similar final performance to SD$^{\text{GT}}$ in most tasks. A failure case for SD$_1^{\text{PerSAM}}$ is Reacher Easy, where a single data point is insufficient to obtain a quality segmentation for the small task-relevant objects.

5.1.2 Comparison with Baselines

We compare **SD$^{\text{approx.}}$** with state-of-the-art methods, including DreamerPro (Deng et al., 2022), RePo (Zhu et al., 2023), TIA (Fu et al., 2021), and TD-MPC2 (Hansen et al., 2023). DreamerPro incorporates prototypical representation learning in the DREAMER framework; RePo minimizes mutual information between observations and latent states while maximizing it between states and future rewards; TIA learns separate task-relevant and task-irrelevant representations that are combined to decode observations; and TD-MPC2 decodes a terminal value function. Only TIA relies on observation reconstruction. Further details are in Appendix M.

Our results in Fig. 3a show that our method consistently outperforms the baselines in performance and sample efficiency. TIA underperforms in many tasks, requiring many samples to infer task-relevant observations from rewards and needing exhaustive hyperparameter tuning. Even with optimal settings, it may lead to degenerate solutions where a single branch captures all information. In contrast, our method focuses on task-relevant parts without additional tuning by effectively injecting prior knowledge. RePo performs comparably to ours in Cartpole Swingup but underperforms in other tasks and converges more slowly.

TD-MPC2 struggles significantly in distracting environments. We speculate that spurious correlations from distractions introduce noise to value-function credit assignment that hinders representation learning. Our method mitigates this by directly supervising task-relevant features, yielding more consistent and lower-variance targets.

DreamerPro is the most competitive, demonstrating the effectiveness of prototypical representation learning for control. However, it often requires more environment interactions and converges to lower performance.

Notably, no prior work successfully solved Cartpole Swingup with sparse rewards, underscoring the challenge of inferring task relevance from weak signals. Our method achieves near-oracle performance and is the only one to train an agent with sparse rewards amidst distractions. This suggests the potential to train agents in real-world, distraction-rich environments without extensive reward engineering.

5.1.3 Ablation Study

We investigate the effects of the components in $\text{SD}^{\text{approx.}}$ by addressing: (1) the benefits of using segmentation models for targets vs. input preprocessing; (2) the effectiveness of the selective L_2 loss compared to the naive L_2 loss; and (3) the impact of the segmentation quality on RL performance. In these experiments, we fine-tune PerSAM with a single data point for segmentation mask prediction.

Using segmentation masks for an auxiliary task vs. input preprocessing. We create a variant of $\text{SD}_1^{\text{PerSAM}}$ that uses masked observations for both inputs and targets, denoted in Tab. 1 by *As Input*. This variant is analogous to prior methods (James et al., 2019; So et al., 2022) that use segmentation models for input preprocessing in control tasks. The result in Tab. 1 suggests that $\text{SD}_1^{\text{PerSAM}}$, in addition to not requiring mask prediction at test-time, also achieves better test performance and lower variance. Using predicted masks as input is more prone to segmentation errors, restricting the agent's perception when masks are incorrect and making training more challenging. In contrast, $\text{SD}^{\text{approx.}}$ receives intact observations, with task-relevant filtering at the encoder level, leading to better state abstraction. Further analysis on test-time segmentation quality's impact is in Appendix C.

Table 1: Final performance of SD variants. Mean over 4 runs with the standard error of the mean is reported. The highest means are highlighted.

Task	$\text{SD}_1^{\text{PerSAM}}$	As Input	Naive L_2
Cartpole Swingup	**730** ± **75**	565 ± 108	719 ± 62
Cartpole Swingup Sparse	**521** ± **92**	457 ± 151	408 ± 114
Cheetah Run	**619** ± **35**	524 ± 37	486 ± 58
Hopper Stand	**846** ± **27**	689 ± 39	790 ± 51
Reacher Easy	597 ± 97	**642** ± **116**	415 ± 50
Walker Run	**730** ± **13**	589 ± 28	557 ± 51

Selective L_2 loss vs. naive L_2 loss. As shown in Tab. 1, $\text{SD}_1^{\text{PerSAM}}$ consistently outperforms the *Naive* L_2 variant, especially in complex tasks like Cheetah Run and Walker Run. Segmentation models often miss embodiment components (Fig. 4, third row). With the naive L_2 loss, the model replicates these errors, leading to incomplete latent representations and harming dynamics learning (Fig. 4a, fourth row). In contrast, $\text{SD}^{\text{approx.}}$ self-corrects by skipping the L_2 computation where PerSAM targets are likely wrong (Fig. 4b, fourth row). Fig. 4(c)&(d) show that the naive L_2 loss follows PerSAM's trends, while the selective L_2 loss recovers from poor recall with only a moderate precision decrease.

Impact of segmentation quality on RL performance. Fig. 3b plots the training-time segmentation quality against the RL agent's test-time performance. Comparing three SD variants with different mask qualities (two estimated, one ground truth), we observe that better segmentation tends to lead to higher RL performance, as accurate targets better highlight task-relevant components. This suggests that improved segmentation models can enhance agent performance without ground-truth masks. In Cartpole Swingup, one of two exceptions, the IoU difference between $\text{SD}_1^{\text{PerSAM}}$ and $\text{SD}_5^{\text{SegFormer}}$ is small, and the test returns may fall within the margin of error. In Walker Run, the other exception, all variants show high segmentation quality and reach near-optimal performance.

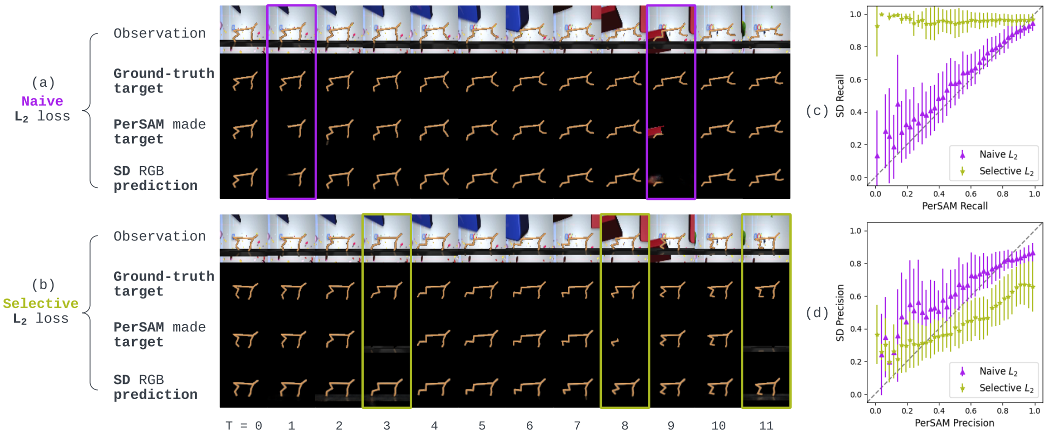


Figure 4: (a)+(b) **Qualitative comparison of SD trained with naive and selective L_2 loss.** Trajectories are taken from each method's train-time replay buffer, selected to have the same background. Frames with PerSAM error are highlighted. The model trained with the selective L_2 loss overcomes errors in the target, whereas the one trained with the naive L_2 loss memorizes target errors. (c)+(d) shows the precision and recall of PerSAM and the SD RGB decoder prediction. SD RGB predictions are binarized using a threshold to compute recall and precision w.r.t. the ground-truth mask. The data points used for plotting are from the same Cheetah Run training experiment as in (a)+(b). The selective L_2 loss significantly improves the recall with only a moderate impact on precision.

Here, we hypothesis that a small amount of noise in the target may act as a regularizer, contributing to marginally better downstream performance.

5.2 Meta-World Experiments

Object manipulation is a natural application for our method where prior knowledge can be applied straightforwardly by identifying and masking task-relevant objects and robot embodiments. We evaluate SD on six tasks from Meta-World (Yu et al., 2019), a popular benchmark for robotic manipulation. Depending on the difficulty of each task, we conduct experiments for 30K, 100K, and 1M environment steps, with an action repeat of 2 (details in Appendix H). Preliminary tests showed that SegFormer performs well with few-shot learning on small objects. We fine-tune SegFormer with 10 data points to estimate masks in these experiments.

Fig. 5 suggests that our approach outperforms the baselines overall, with a more pronounced advantage in tasks involving small objects like Coffee-Button. Our method excels because it focuses on small, task-relevant objects, avoiding the reconstruction of unnecessary regions that occupy much of the input. In contrast, the baselines struggle as they often underestimate the significance of these small yet highly task-relevant objects. Among the baselines, RePo (Zhu et al., 2023) is the most competitive. However, RePo performs poorly in a sparse reward setup (see Appendix J).

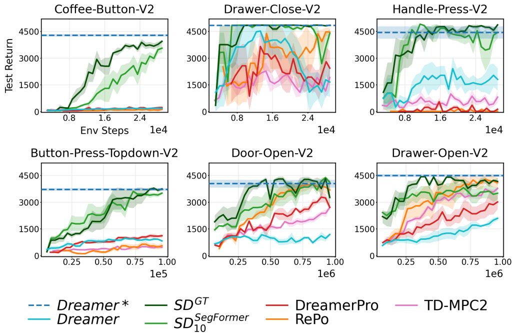
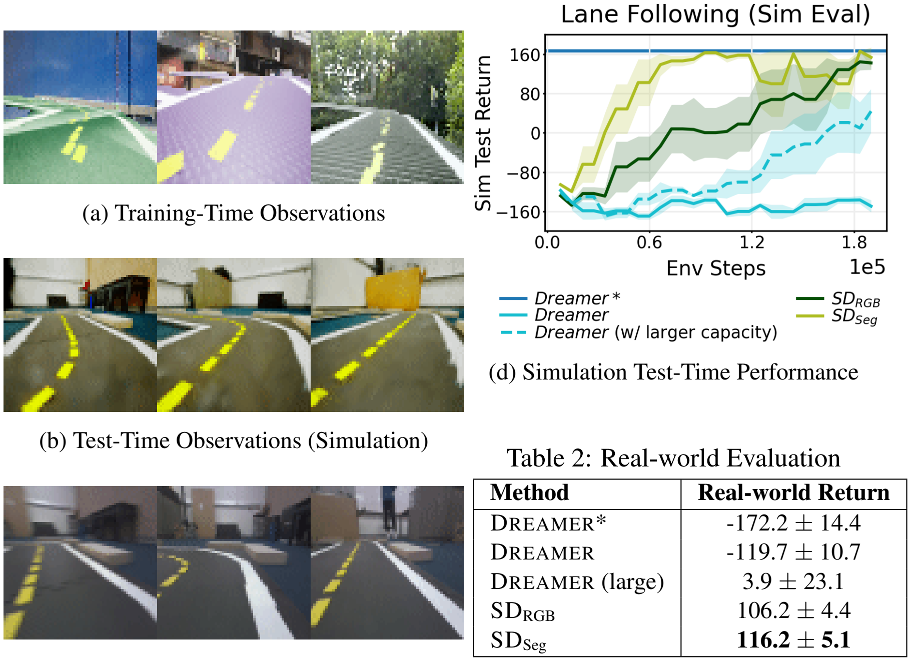


Figure 5: **Learning curves on six visual robotic manipulation tasks from Meta-World.** All curves show the mean over 4 seeds with the standard error of the mean shaded.

5.3 Duckiebot Lane Following Experiments

Domain randomization enables training generalizable agents by exposing them to diverse scenarios, allowing zero-shot transfer to unseen environments within the randomized range (Tobin et al., 2017; Tang et al., 2024). For that purpose, randomization intentionally introduces noise, which can make



(a) Training-Time Observations

(b) Test-Time Observations (Simulation)

(c) Test-Time Observations (Real-world)

(d) Simulation Test-Time Performance

Table 2: Real-world Evaluation

Method	Real-world Return
DREAMER*	-172.2 ± 14.4
DREAMER	-119.7 ± 10.7
DREAMER (large)	3.9 ± 23.1
SD_{RGB}	106.2 ± 4.4
SD_{Seg}	**116.2 ± 5.1**

Figure 6: (a) Domain-randomized simulation observations with variations in background, foreground color, texture, and layout. (b) Digital-twin observations generated using Gaussian splatting. (c) Real-world environment observations. (d) Test-time performance in the simulated environment. Table 2: Performance deploying each method in the real-world environment after 200k steps. We present mean and SEM values over 4 seeds.

training harder, require more data, and sometimes fail with limited-capacity models (Fu et al., 2021). On the other hand, too narrow a randomization range can limit generalization abilities and cause failures when agents are deployed outside that range. Ideally, an agent should be trained on a broad range of perturbations without unnecessarily increasing model complexity or training difficulty.

Introducing domain randomization allows us, as designers, to control what constitutes a distraction, and this design philosophy can also serve as prior knowledge for SD. We evaluate our method on a robotic lane-following task in the Duckietown platform (Paull et al., 2017), where the objective is to follow a marked lane on a looping track with a wheeled robot while minimizing deviations. We conduct training for this task in an Unreal Engine (Epic Games, 2024) simulation that employs domain randomization across multiple factors, including background and foreground colors and textures, lighting conditions, physics, and more (Fig. 6a). The training simulation offers segmentation map rendering, enabling us to use ground-truth masks for training SD. Thus, in this section, SD refers to SD^{GT}. Additional training details, such as the reward function and randomization axes, can be found in Appendix I.

We evaluate models in two test environments. The first test environment is an instance of the simulation designed to be a digital-twin of the real-world environment, constructed using Gaussian splatting (Kerbl et al., 2023), featuring a variation of colors, lighting, textures, layouts, and backgrounds that is unseen in training (Fig. 6b). It enables reproducible and repeatable experiments before real-world deployment. The second environment is the real-world track that the digital-twin approximates, where we assess how well the trained model zero-shot transfers to real-world robot conditions (Fig. 6c).

Fig. 6d shows that SD_{RGB}, which decodes task-relevant RGB pixels (i.e., pixels belonging to the lane), learns a generalizable agent that trains effectively under rigorous domain randomization. Its performance approaches that of DREAMER*, which is trained directly in the simulation test environment. The reconstruction targets, which remove control-irrelevant pixels, effectively guide the model to learn features that are invariant to background distractions. This result also suggests that SD can, to some extent, tolerate foreground perturbations (e.g., lane colors, camera view) that are not filtered out by the target images. A further exploration of SD under perturbations to task-relevant objects in DMC can be found in Appendix D.

Another variant of our method, SD_{Seg}, decodes segmentation maps as an auxiliary task target. Since the decoder target contains fewer distractions, this variant converges faster, as the model is explicitly guided to ignore irrelevant foreground noise. Both variants of SD generalize well, with SD_{RGB} retaining visual details in foreground that can be beneficial for downstream tasks, while SD_{Seg} enforces a more abstract representation for faster convergence; SD_{RGB} is preferable when appearance cues aid decision-making, whereas SD_{Seg} suits scenarios where structural consistency matters more. Appendix I presents sample decoding targets for each model.

On the other hand, DREAMER fails to train a drivable agent after 200K steps, likely because it allocates capacity to modeling task-irrelevant background information introduced by randomization. While increasing model capacity allows DREAMER to learn a better policy (dashed curve), it requires significantly more samples to achieve agent that successfully drives. The performance gap between SD_{RGB} and DREAMER suggests that background distractions pose a particular challenge during training, as they often occupy large portions of the pixel space and are dynamic. Encoding background information in the latent space introduces task-irrelevant dynamics, increasing learning complexity and leading to inefficient use of model capacity.

We deploy these models for 5 episodes each in the real-world environment after 200k steps of training and show average episode returns in Tab. 2. Although DREAMER* performs well in the simulation test environment, its real-world deployment suffers due to visual disparity between the training and testing environment. While the Gaussian splat simulation closely resembles the real world, photometric properties such as brightness and hue are not perfectly aligned, preventing effective zero-shot transfer. Similar to its performance in simulation, DREAMER fails to drive in the real world and shows slight improvement with a larger model capacity. In contrast, both variants of SD successfully achieve zero-shot transfer to the real-world, despite encountering unseen appearances and a small dynamics distribution shift during deployment. SD enables generalizable perception and zero-shot transfer without introducing additional overhead at test time, making it particularly practical for real-world applications.

6 Conclusion

We propose SD, a simple yet effective method for learning task-relevant features in MBRL frameworks by using segmentation masks informed by domain knowledge. Using ground-truth masks, SD^{GT} achieves performance comparable to undistracted DREAMER with high sample efficiency in distracting environments given accurate prior knowledge. Our main method, $\text{SD}^{\text{approx.}}$, uses estimated masks from off-the-shelf one-shot or few-shot segmentation models and employs a selective L_2 loss. Experimental results across diverse domains, including a sim-to-real lane-following task, suggest that our method is a practical and powerful tool for training generalizable, deployable agents in dynamic environments, with no additional overhead at test time.

The proposed method achieves strong performance across diverse tasks with distractions and effectively incorporates human input to indicate task relevance. This enables practitioners to readily train an agent for their own purposes without extensive reward engineering. This work also advances the integration of computer vision and RL by demonstrating how recent segmentation advances can help address challenges in visual control tasks. We discuss limitations and future directions in Appendix O.

References

Rajaram Anantharaman, Matthew Velazquez, and Yugyung Lee. Utilizing mask r-cnn for detection and segmentation of oral diseases. In *International Conference on Bioinformatics and*

Biomedicine, pp. 2197–2204. IEEE, 2018.

Homanga Bharadhwaj, Mohammad Babaeizadeh, Dumitru Erhan, and Sergey Levine. Information prioritization through empowerment in visual model-based rl. In *International Conference on Learning Representations*, 2022.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pp. 1597–1607. PMLR, 2020.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In Dekai Wu, Marine Carpuat, Xavier Carreras, and Eva Maria Vecchi (eds.), *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111, Doha, Qatar, October 2014. Association for Computational Linguistics. DOI: 10.3115/v1/W14-4012. URL https://aclanthology.org/W14-4012.

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

Epic Games. Unreal engine, 2024. URL https://www.unrealengine.com.

Norm Ferns, Prakash Panangaden, and Doina Precup. Bisimulation metrics for continuous markov decision processes. *SIAM Journal on Computing*, 40(6):1662–1714, 2011.

Stefano Ferraro, Pietro Mazzaglia, Tim Verbelen, and Bart Dhoedt. Focus: Object-centric world models for robotics manipulation. *arXiv preprint arXiv:2307.02427*, 2023.

Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *International Conference on Robotics and Automation*, pp. 2786–2793. IEEE, 2017.

Xiang Fu, Ge Yang, Pulkit Agrawal, and Tommi Jaakkola. Learning task informed abstractions. In *International Conference on Machine Learning*, pp. 3480–3491. PMLR, 2021.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning*, pp. 2170–2179. PMLR, 2019.

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. Maniskill2: A unified benchmark for generalizable manipulation skills. In *International Conference on Learning Representations*, 2023.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

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

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in Neural Information Processing Systems*, 31, 2018.

Nicklas Hansen and Xiaolong Wang. Generalization in reinforcement learning by soft data augmentation. In *International Conference on Robotics and Automation*, pp. 13611–13617. IEEE, 2021.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 3680–3693, 2021.

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

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *International Conference on Computer Vision*, pp. 2961–2969, 2017.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations*, 2017.

Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *Computer Vision and Pattern Recognition*, pp. 12627–12637, 2019.

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021.

Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. In *Conference on Learning for Dynamics and Control*, 2020.

Gaspard Lambrechts, Adrien Bolland, and Damien Ernst. Informed POMDP: Leveraging additional information in model-based RL. *Reinforcement Learning Journal*, 1, 2024.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pp. 5639–5650. PMLR, 2020.

Jessy Lin, Yuqing Du, Olivia Watkins, Danijar Hafner, Pieter Abbeel, Dan Klein, and Anca Dragan. Learning to model the world with language. In *International Conference on Machine Learning*, 2024.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. *Advances in Neural Information Processing Systems*, 28, 2015.

Suraj Nair, Silvio Savarese, and Chelsea Finn. Goal-aware prediction: Learning to model what matters. In *International Conference on Machine Learning*, pp. 7207–7219. PMLR, 2020.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, 2022.

Tung D Nguyen, Rui Shu, Tuan Pham, Hung Bui, and Stefano Ermon. Temporal predictive coding for model-based planning in latent space. In *International Conference on Machine Learning*, pp. 8130–8139. PMLR, 2021.

Liam Paull, Jacopo Tani, Heejin Ahn, Javier Alonso-Mora, Luca Carlone, Michal Cap, Yu Fan Chen, Changhyun Choi, Jeff Dusek, Yajun Fang, et al. Duckietown: an open, inexpensive and flexible platform for autonomy education and research. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1497–1504. IEEE, 2017.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Computer Vision and Pattern Recognition*, pp. 779–788, 2016.

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.

Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. In *Conference on Robot Learning*, pp. 1332–1344. PMLR, 2022.

John So, Amber Xie, Sunggoo Jung, Jeffrey Edlund, Rohan Thakker, Ali Agha-mohammadi, Pieter Abbeel, and Stephen James. Sim-to-real via sim-to-seg: End-to-end off-road autonomous driving without real data. In *Conference on Robot Learning*, 2022.

Austin Stone, Oscar Ramirez, Kurt Konolige, and Rico Jonschkowski. The distracting control suite–a challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. Deep reinforcement learning for robotics: A survey of real-world successes. *Annual Review of Control, Robotics, and Autonomous Systems*, 8, 2024.

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

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Tongzhou Wang, Simon S Du, Antonio Torralba, Phillip Isola, Amy Zhang, and Yuandong Tian. Denoised mdps: Learning world models better than the world itself. In *International Conference on Machine Learning*, 2022.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *Advances in Neural Information Processing Systems*, 2021.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, 2019.

Yuhui Yuan, Lang Huang, Jianyuan Guo, Chao Zhang, Xilin Chen, and Jingdong Wang. Ocnet: Object context network for scene parsing. *arXiv preprint arXiv:1809.00916*, 2018.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2021.

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew Johnson, and Sergey Levine. Solar: Deep structured representations for model-based reinforcement learning. In *International Conference on Machine Learning*, pp. 7444–7453. PMLR, 2019.

Renrui Zhang, Zhengkai Jiang, Ziyu Guo, Shilin Yan, Junting Pan, Hao Dong, Peng Gao, and Hongsheng Li. Personalize segment anything model with one shot. *arXiv preprint arXiv:2305.03048*, 2023.

Weipu Zhang, Adam Jelley, Trevor McInroe, and Amos Storkey. Objects matter: object-centric world models improve reinforcement learning in visually complex environments. *arXiv preprint arXiv:2501.16443*, 2025.

Chuning Zhu, Max Simchowitz, Siri Gadipudi, and Abhishek Gupta. Repo: Resilient model-based reinforcement learning by regularizing posterior predictability. In *Advances in Neural Information Processing Systems*, 2023.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Visualization of Tasks

A.1 DeepMind Control suite (DMC)

Fig. 7 visualizes the six tasks in DMC (Tassa et al., 2018) used in our experiments. Each row presents the observation from the standard environment, the corresponding observation with added distractions, the ground-truth segmentation mask, and the RGB target with the ground-truth mask applied. Cartpole Swingup Sparse and Cartpole Swingup share the same embodiment and dynamics. Cartpole Swingup Sparse only provides a reward when the pole is upright, whereas Cartpole Swingup continuously provides dense rewards weighted by the proximity of the pole to the upright position. Reacher Easy entails two objects marked with different colors in the segmentation mask, as shown in Fig. 7e 3rd column. Before passing the mask to SD, the mask is converted to a binary format where both objects are marked as *true* as task-relevant.

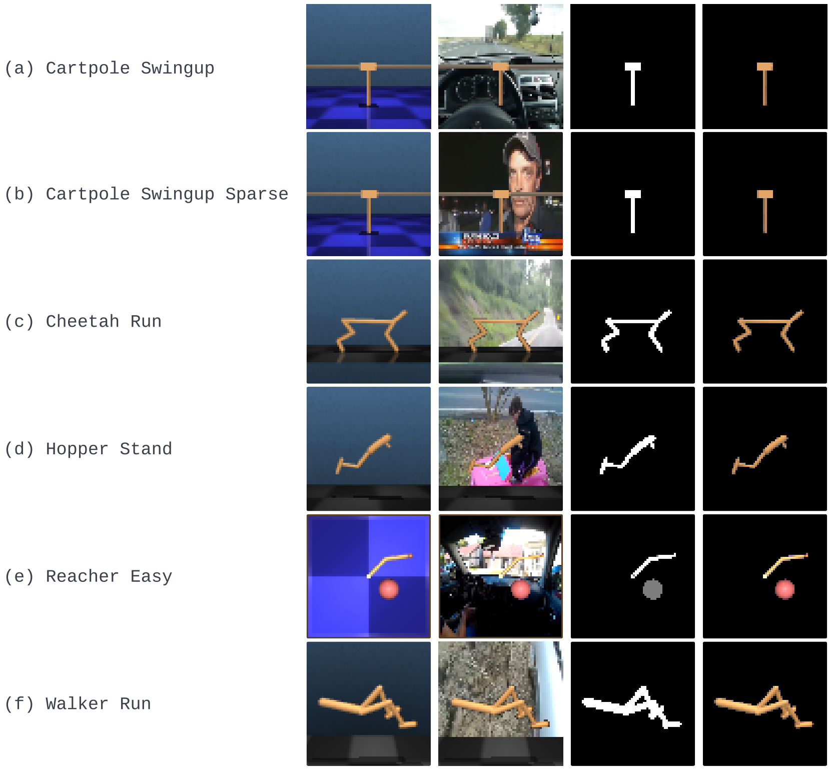
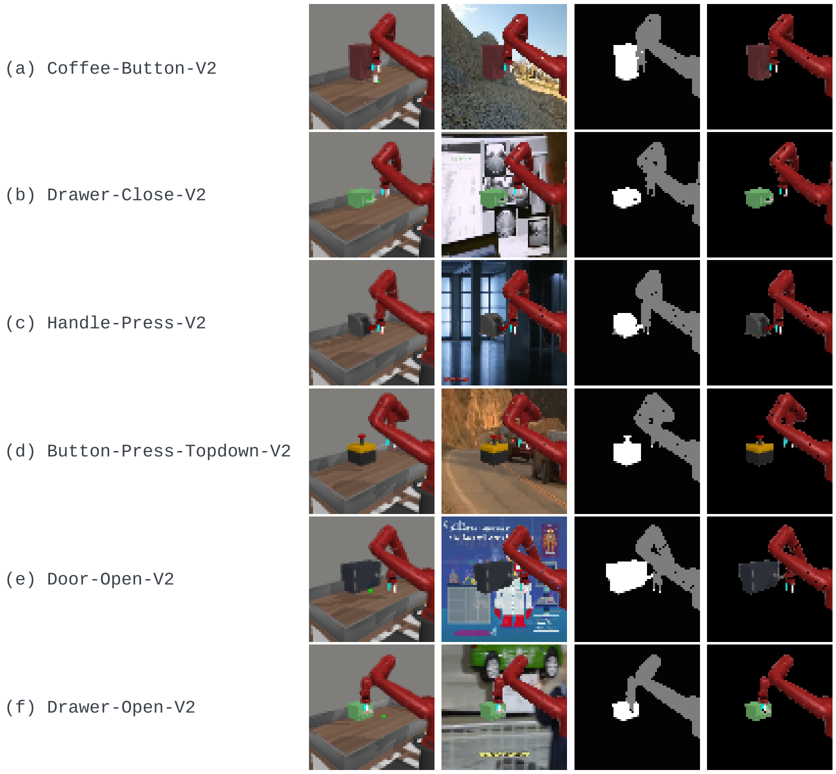


Figure 7: DMC tasks. Left to right: (1) standard environment observations, (2) distracting environment observations, (3) ground-truth segmentation masks, and (4) RGB observations with ground-truth masks applied. We use (4) as auxiliary reconstruction targets in SD^{GT}.

A.2 Meta-World

Fig. 8 shows the six tasks from Meta-World-V2 used in our experiments. Meta-World is a realistic robotic manipulation benchmark with challenges such as multi-object interactions, small objects, and occlusions.



(a) Coffee-Button-V2

(b) Drawer-Close-V2

(c) Handle-Press-V2

(d) Button-Press-Topdown-V2

(e) Door-Open-V2

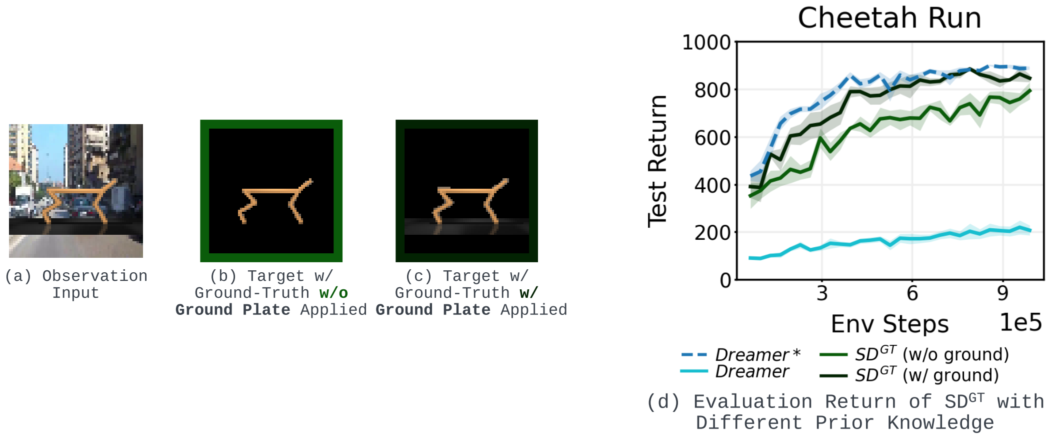
(f) Drawer-Open-V2

Figure 8: Meta-World tasks. Left to right: (1) standard environment observations, (2) distracting environment observations, (3) ground-truth segmentation masks, and (4) RGB observations with ground-truth masks applied. We use (4) as auxiliary reconstruction targets in SD^{GT}. Masks with multiple classes for different objects are converted to binary masks (all non-background regions are *true* and task-relevant) before use with SD.

B The Impact of Prior Knowledge

We investigate the impact of accurate prior knowledge of task-relevant objects. Specifically, we conduct additional experiments on Cheetah Run—the task showing the largest disparity between DREAMER* and SD$^{\mathrm{GT}}$ in Fig. 3a. In our primary experiment, we designated only the cheetah's body as the task-relevant object. However, since the cheetah's dynamics are influenced by ground contact, the ground plate should have also been considered task-relevant.

Fig. 9 (a–c) illustrates the observation with distractions, the auxiliary target without the ground plate, and with the ground plate included, respectively. Fig. 9d compares SD$^{\mathrm{GT}}$ trained with different selections of task-relevant objects included in the masked RGB reconstruction targets. We show that including the ground plate leads to faster learning and performance closer to that of the oracle. This highlights the significant influence of prior knowledge on downstream tasks, suggesting that comprehensively including task-relevant objects yields greater benefits.



(a) Observation Input

(b) Target w/ Ground-Truth **w/o Ground Plate** Applied

(c) Target w/ Ground-Truth **w/ Ground Plate** Applied

(d) Evaluation Return of SD$^{\mathrm{GT}}$ with Different Prior Knowledge

Figure 9: **The impact of prior knowledge on Cheetah Run.** (d) The mean over 4 seeds with the standard error of the mean (SEM) is shaded.

C The Impact of Test-Time Segmentation Quality on Performance

We investigate how test-time segmentation quality affects $\mathrm{SD}^{\text{approx.}}$ as well as the *As Input* variation that applies mask predictions to RGB inputs in addition to reconstruction targets. For this analysis, we use PerSAM fine-tuned with a single data point for segmentation prediction. To measure segmentation quality, we compute episodic segmentation quality by averaging over frame-level IoU. In Fig. 10 we plot episode segmentation quality versus test-time reward on the evaluation episodes during the last 10% of training time.

Fig. 10 illustrates that $\mathrm{SD}^{\text{approx.}}$ exhibits greater robustness to test-time segmentation quality compared to the *As Input* variation, with the discrepancy increasing as the IoU decreases. This disparity primarily arises because *As Input* relies on observations restricted by segmentation predictions, and thus its performance deteriorates quickly as the segmentation quality decreases. In contrast, $\mathrm{SD}^{\text{approx.}}$ takes the original observation as input and all feature extraction is handled by the observation encoder, informed by our masked RGB reconstruction objective. Consequently, $\mathrm{SD}^{\text{approx.}}$ maintains resilience to test-time segmentation quality.

An intriguing observation is that a poorly trained agent can lead to poor test-time segmentation quality. For instance, Cartpole Swingup (Sparse) exhibits different segmentation quality distributions between $\mathrm{SD}^{\text{approx.}}$ and *As Input*. This discrepancy occurs because the sub-optimal agent often positions the pole at the cart track edge, causing occlusion and hindering accurate segmentation prediction by PerSAM.

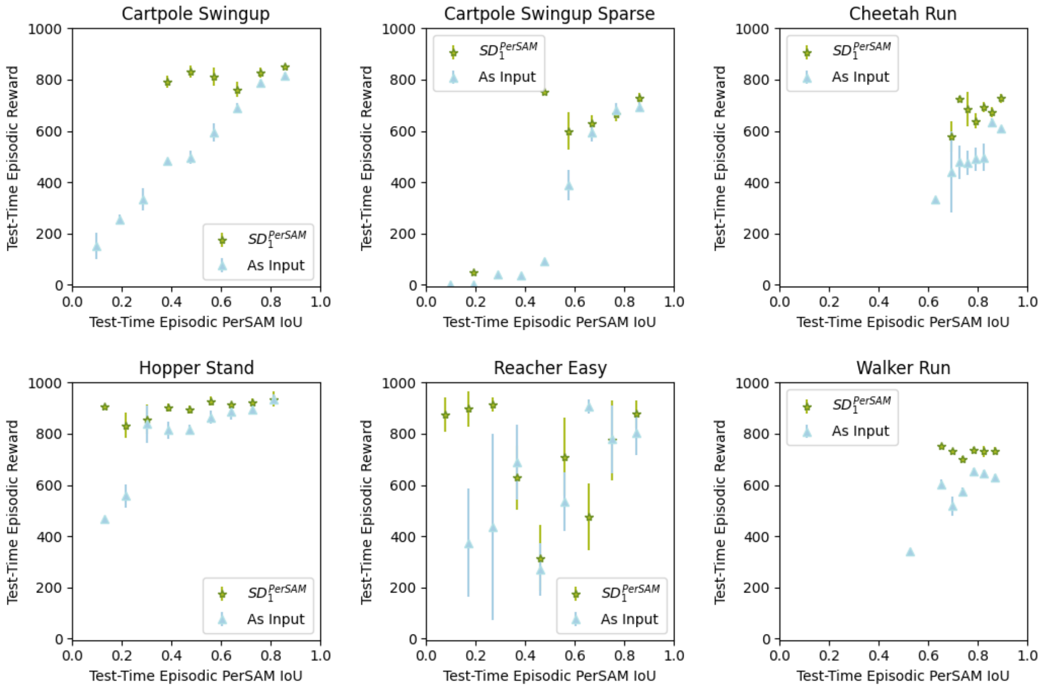


Figure 10: Test-time episodic reward vs PerSAM episodic IoU for $\mathrm{SD}_1^{\text{PerSAM}}$ and *As Input* ($\mathrm{SD}_1^{\text{PerSAM}}$ with masked RGB observations as input). $\mathrm{SD}_1^{\text{PerSAM}}$ is more robust to test-time segmentation prediction errors.

D Robustness to Foreground Distractions

SD is primarily designed to improve world model learning by omitting task-irrelevant background features from its latent state. In this section, we additionally investigate SD's robustness to distractions that affect the task-relevant foreground. Specifically, on DMC Walker Run and Cartpole Swingup, we test SD's performance when training and testing under three types of visual perturbations: (1) foreground occlusions, (2) color shifts, and (3) camera angle shifts. We find that the performance of both SD and the segmentation models used to train SD is not significantly diminished by the inclusion of small foreground distractions. We first outline each of the distraction types we experiment with and then discuss experimental results:

D.1 Foreground Distraction Details

Foreground Occlusion To simulate occlusions of task-relevant features, we introduce a moving foreground distractor—a blue rectangle (Fig. 11) rendered near the center of the scene for 4 to 6 consecutive frames, appearing after every 18 to 22 frames. These intervals are uniformly sampled each time the distractor appears, so approximately 25% of the image frames in an episode contain the distractor. Its movement follows pixel-space trajectories defined by randomized Δx and Δy values drawn from the interval (–3, 3), with new values sampled each time the distractor is rendered. Although we only test with a blue rectangle, given the capabilities of visual foundation models (VFMs), we expect our method to generalize well to a variety of foreground occluders with different properties.

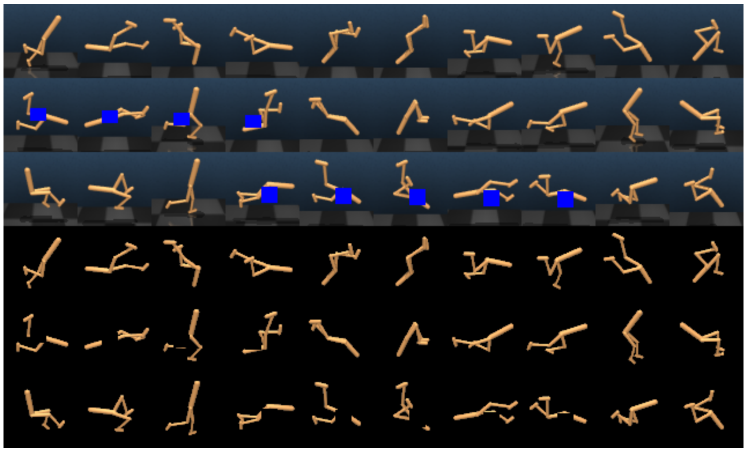


Figure 11: Examples of frames with **foreground occlusion** in the environment and corresponding predictions from the segmentation model that remain robust to occlusions in the test set.

Color Shifts To simulate variations in the agent's appearance or task-relevant objects, we apply color perturbations following Stone et al. (2021), setting their proposed environment color shift hyperparameters to have a max delta of 0.1 and a step standard deviation of 0.0, resulting in a randomly sampled, temporally-constant color shift throughout each episode (Fig. 12). These perturbations mimic real-world factors like lighting variations during deployment and test the model's ability to generalize to such mismatches at deployment time.

Camera Angle Shifts To introduce variations in camera perspective, we similarly follow Stone et al. (2021), applying a scaling factor of 0.1 which defines a viewing range of the camera, shifting the camera view by a random amount in each episode (Fig. 13). These perturbations simulate real-world scenarios where the agent's viewpoint changes due to physical discrepancies and test the model's robustness to altered perspectives.

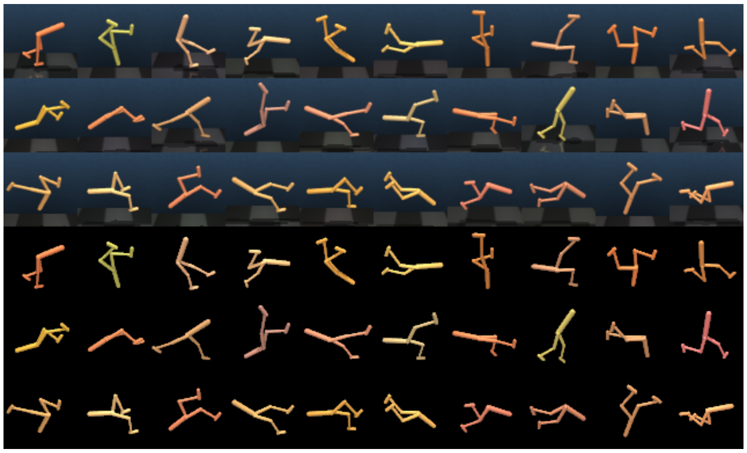


Figure 12: Examples of **color** perturbations applied to the agent and corresponding predictions from the segmentation model that remain robust to color changes in the test set.

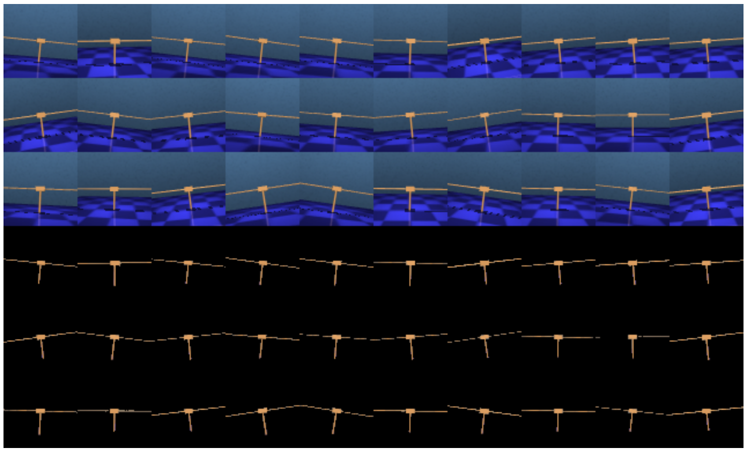


Figure 13: Examples of **camera angle** perturbations and corresponding predictions from the segmentation model that remain robust to camera view variations in the test set.

D.2 Results with Foreground Distractions

For each perturbation type, we train and test both the segmentation model and SD with domain randomization over the distraction method's parameters. We use the SegFormer foundation model finetuned on 100 pairs of domain-randomized images in these experiments. We evaluate SD on the same parameter distributions as used in training time.

Segmentation Model Robustness to Foreground Distractions Figures 11–13 show that the SegFormer model still effectively isolates task-revelant objects of interest despite challenges presented by foreground occlusions, color shifts, and camera view changes. These findings align with our primary experiments on background distractions, further reinforcing that segmentation models provide a robust strategy for guiding representation learning in RL under many types of domain randomization.

SD Robustness to Foreground Distractions Unlike background distractions, foreground perturbations cannot be fully filtered out and remain in the SD decoding target. While this might raise concerns about wasted capacity by encoding spurious information, our results (Table 3) demonstrate that SD still learns effective agent behavior with these perturbations applied. Notably, in environments with color or camera view changes, our method, by focusing on agent-centric features, outperforms Dreamer* trained in the unmodified environment.

	Standard Environment (Dreamer*)	Foreground Occlusions (SD)	Foreground Occlusions (SD Naive L_2 Loss)	Color Shifts (SD)	Camera Angle Shifts (SD)
Walker Run	752 ± 9	740 ± 4	688 ± 37	761 ± 3	752 ± 10
Cartpole Swingup	818 ± 52	860 ± 2	852 ± 7	870 ± 4	863 ± 4

Table 3: Test returns comparing Dreamer* in the base environment with SD in modified environments where foreground distractions are applied. Despite additional variation added to task-relevant features, SD's performance is not significantly diminished compared to Dreamer's performance in the original environment. With color and camera angle shifts applied to SD only, SD still outperforms Dreamer*.

Additionally, the selective L_2 loss proves highly effective in handling occlusions, enabling the recovery of occluded foreground agent features. We compare our default SD method against a version with Naive L_2 loss and see a large drop in test-time return when performing this ablation. This highlights the versatility of selective L_2 loss across different scenarios. While SD was originally designed to mitigate distractions outside task-relevant objects, these results demonstrate its robustness across a broader range of real-world perturbations.

E Segmentation Quality in Meta-World

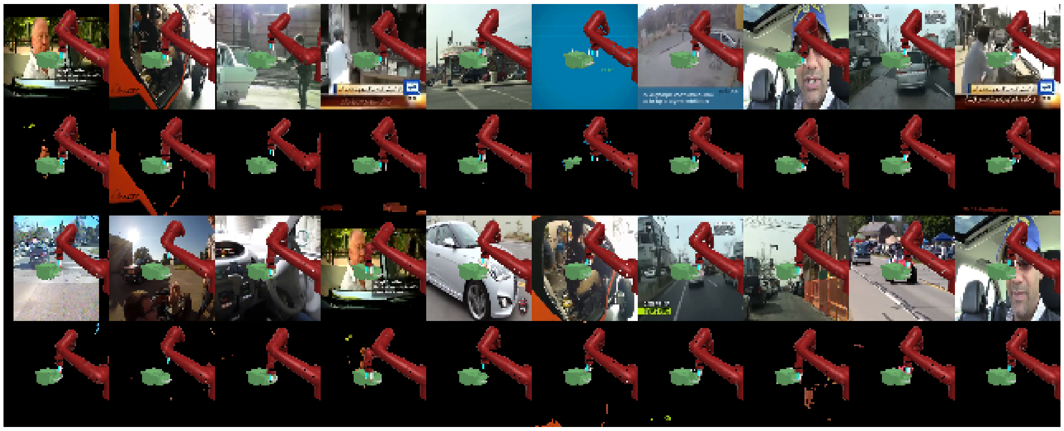


Figure 14: Examples of background perturbations and corresponding predictions from the segmentation model on Drawer-Open-V2 in the test set.

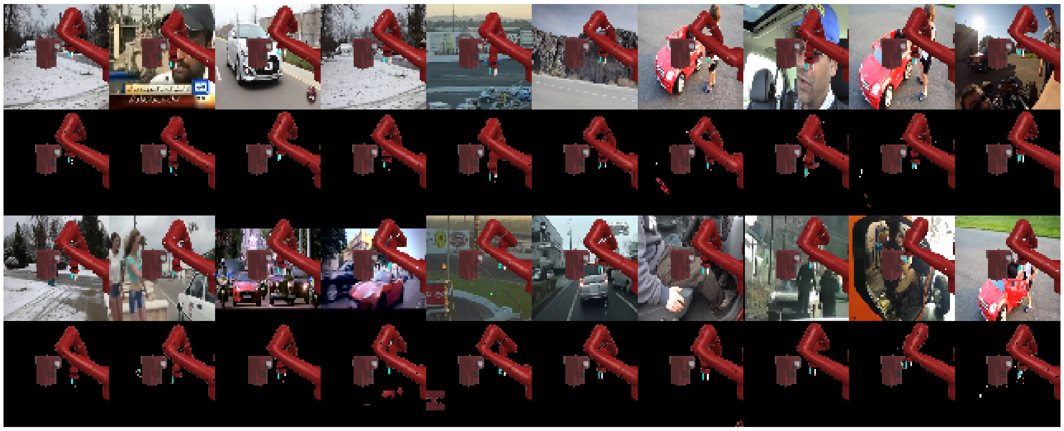


Figure 15: Examples of background perturbations and corresponding predictions from the segmentation model on Coffee-Button-V2 in the test set.

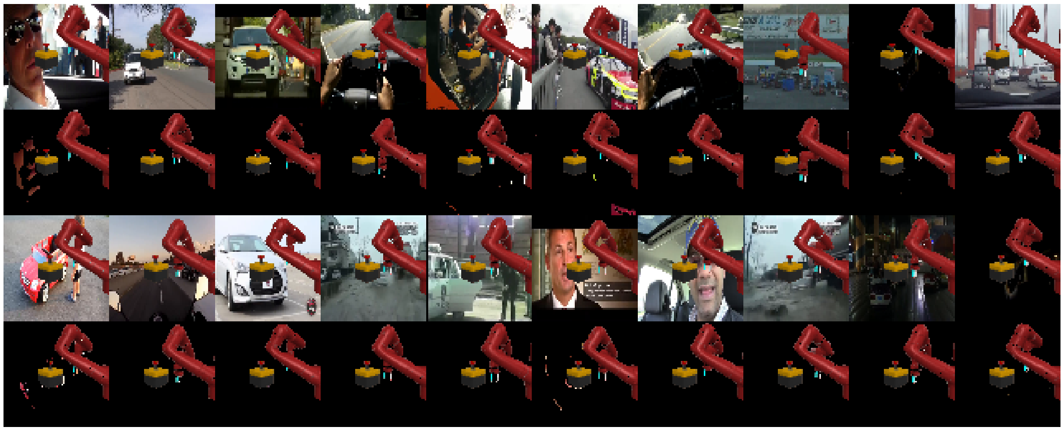


Figure 16: Examples of background perturbations and corresponding predictions from the segmentation model on Button-Press-Topdown-V2 in the test set.

F Ablation without Stop Gradient

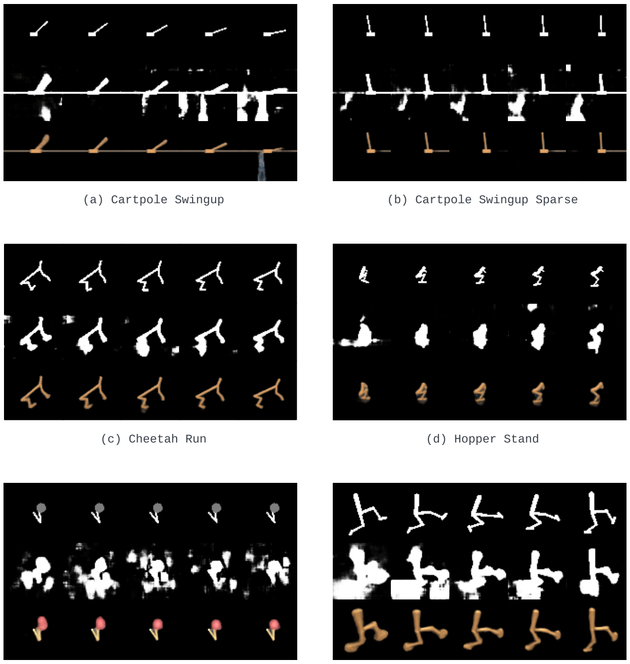
Should the SD$^{\text{approx.}}$ world model be shielded from gradients of the binary mask decoder head?

To estimate potential regions on RGB targets where task-relevant regions are incorrectly masked out, we train a binary mask prediction head on the world model to help detect false negatives in masks provided by the foundation model. We see better performance when gradients from this binary mask decoder objective are not propagated to the rest of the world model. Thus, the default SD$^{\text{approx.}}$ architecture is trained with the gradients of the binary mask branch stopped at its $[h_t; z_t]$ inputs, and the latent representations in the world model are trained only by the task-relevant RGB branch in addition to the standard DREAMER reward/continue prediction and KL-divergence between the dynamics prior and observation encoder posterior. Tab. 4 shows that the performance drops significantly when training without stopping these gradients.

We also examine masks predicted by the binary mask decoder head in Fig. 17. Predictions are coarser grained than their RGB counterparts, lacking details important for predicting intricate forward dynamics. Overall, reconstructing RGB observations with task-relevance masks applied demonstrates itself as a superior inductive bias to learn useful features for downstream tasks compared to binary masks or raw unfiltered RGB observations.

Table 4: Final performance of SD and SD without stop gradient.

Task	SD$_1^{\text{PerSAM}}$	No SG
Cartpole Swingup	**730** ± **75**	439 ± 81
Cartpole Swingup Sparse	**521** ± **92**	112 ± 40
Cheetah Run	**619** ± **35**	376 ± 50
Hopper Stand	**846** ± **27**	587 ± 127
Reacher Easy	**597** ± **97**	273 ± 74
Walker Run	**730** ± **13**	407 ± 62



(a) Cartpole Swingup

(b) Cartpole Swingup Sparse

(c) Cheetah Run

(d) Hopper Stand

(e) Reacher Easy

(f) Walker Run

Figure 17: From the top row to the bottom row: (1) ground-truth segmentation masks, (2) SD$^{\text{approx.}}$ binary mask predictions, and (3) SD$^{\text{approx.}}$ RGB predictions.

G Distracting DMC Setup

We follow the DBC (Zhang et al., 2021) implementation to replace the background with color videos. The ground plate is also presented in the distracting environment. We used hold-out videos as background for testing. We sampled 100 videos for training from the Kinetics 400 training set of the 'driving car' class, and test-time videos were sampled from the validation set of the same class.

H Distracting Meta-World Setup

We test on six tasks from Meta-World-V2. For all tasks, we use the `corner3` camera viewpoint. The maximum episode length for Meta-World tasks is 500 environment steps, with the action repeat of 2 (making 250 policy decision steps). We classify these tasks into `easy`, `medium`, and `difficult` categories based on the training curve of DREAMER* (DREAMER trained in the standard environments). Coffee Button, Drawer Close, and Handle Press are classified as `easy`, and we train baselines on these for 30K environment steps. Button Press Topdown (`medium`) is trained for 100K steps, and Door Open and Drawer Open (`difficult`) are trained for 1M environment steps.

I Duckiebot Setup

Environment Configuration In the Duckiebot Lane-Following domain, the agent is tasked with driving quickly along the right lane of a looping track while staying close to the lane center. For observations, we provide the current camera view as a size 64×64 RGB image. The action space is a 2D continuous vector in $[-1, 1]^2$ representing target forward and yaw velocities. The agent starts each episode in simulation at a random position on the right lane.

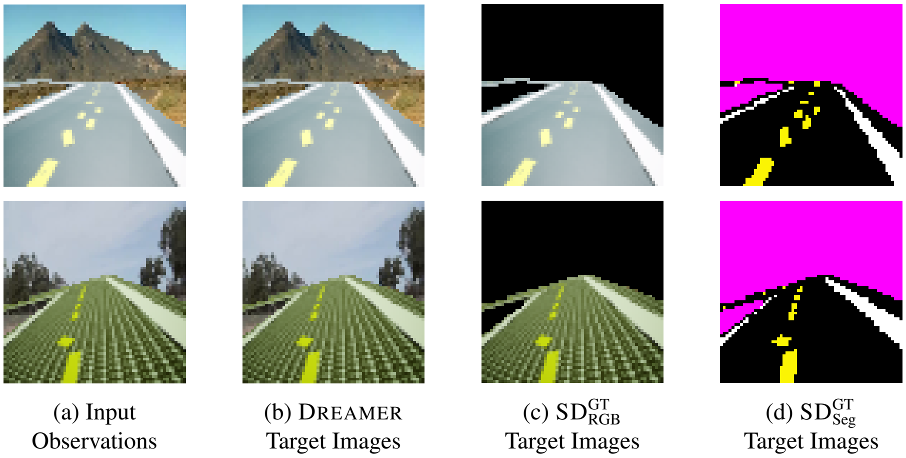
In simulation, the agent is rewarded in every step with a value in $[0, 1]$ proportional to its velocity along the center of right lane on the track. In each step that the agent deviates more than 5cm from the center of the lane, it instead receives a penalty of -1. To encourage smooth driving, the agent is additionally penalized each step proportional to the magnitude of its rotational yaw velocity when moving forward. If the agent drives off the track, the episode terminates, and the agent receives a penalty of -100. Except upon driving out-of-bounds, the episode horizon is 200.

We evaluate rewards in the real environment by tracking the robot's state with an HTC Vive motion tracker. We then replay the agent's states and actions in the Gaussian splat digital-twin simulation of matching size and proportion to calculate equivalent simulation rewards. In real evaluation, we use an episode horizon of 300. We start all real evaluation episodes from the same position on the track.

Domain Randomization We apply domain randomization across four categories to promote robustness and generalization from simulation to the real robot:

- Background: Videos from the Kinetics 400 dataset (Kay et al., 2017) 'Driving Car' class are played in the background to simulate task-irrelevant dynamics.
- Foreground appearance perturbations: We perturb the appearance of foreground objects, such as lane color, lighting, and texture, to ensure the model can handle variations in visual appearance.
- Foreground geometry perturbations: We introduce variations in layout (*i.e.* line marker positioning) and camera view (e.g., tilting, varying field-of-view). This helps the agent generalize to different scene layouts and camera configurations.
- Physics perturbations: We randomize physics parameters to facilitate zero-shot transfer from simulation to the real world. This is done by adding noise to actions and camera positions at each step.

Auxiliary Task Target Visualization Fig. 18 visualizes sample auxiliary target images for DREAMER (b), $\text{SD}^{\text{GT}}_{\text{RGB}}$ (c), and $\text{SD}^{\text{GT}}_{\text{Seg}}$ (d). Moving from (b) to (d), the target images become pro-


(a) Input Observations

(b) DREAMER Target Images

(c) $\text{SD}^{\text{GT}}_{\text{RGB}}$ Target Images

(d) $\text{SD}^{\text{GT}}_{\text{Seg}}$ Target Images

Figure 18: Input observations and corresponding sample decoding target images for each model.

gressively less detailed while retaining task-relevant information, guiding the model to only learn state features necessary for optimal control. For instance, (c) filters out background information, while (d) introduces additional abstraction by hiding pixel values and retaining only layout information. This promotes learning invariant features by ignoring task-irrelevant noise and guiding the encoder to be robust to variations in foreground appearance. It is important to note that not all types of perturbations can be hidden. For instance, all models must handle foreground geometry and physics perturbations that are not filtered by the target images. Our experiments indicate that these types of perturbations are relatively easier for the world models to learn and generalize from.

Training Details We train all models for 200K environment steps with an action repeat of 1. We use the same hyperparameters as DREAMER-V3, except for a reduced model size. Specifically, we set `RSSM.deter`= 32, `units`= 32, and `cnn_depth`= 8 for all models, except for DREAMER (large), which uses `RSSM.deter`= 64, `units`= 64, and `cnn_depth`= 16.

All models except DREAMER* are trained in a domain-randomized simulation environment. DREAMER* is trained in a Gaussian splat simulation environment. Both simulation environments include the same physics and camera shake perturbations.

Policies are evaluated in two settings: (1) Gaussian splat simulation without physics perturbations, and (2) the real-world environment with physics partially mismatched to simulation.

J Results on Meta-World with Sparse Rewards

We also evaluate on sparse reward variations of the distracting Meta-World environments where a reward of 1 is only provided on timesteps when a *success* signal is given by the environment (e.g. objects are at their goal configuration). Rewards are 0 in all other timesteps. The maximum attainable episode reward is 250.

The sparse reward setting is more challenging because the less informative reward signal makes credit assignment more difficult for the RL agent. Fig. 19 shows that our method consistently achieves higher sample efficiency and better performance, showing promise for training agents robust to visual distractions without extensive reward engineering. In Meta-World experiments, TIA (Fu et al., 2021) is not included as it requires exhaustive hyperparameter tuning for new domains and is the lowest-performing method in DMC in general.

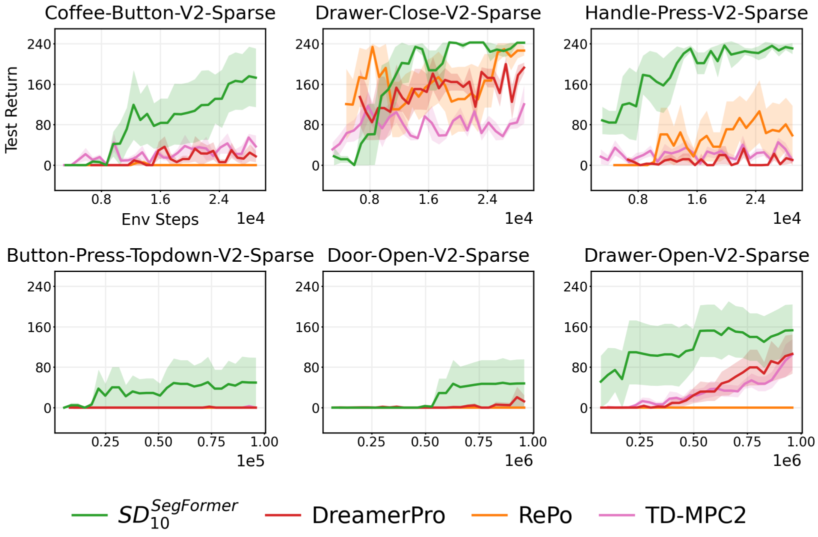


Figure 19: Learning curves on six visual robotic manipulation tasks from Meta-World with sparse rewards.

K Fine-tuning PerSAM and SegFormer

In this section, we describe how we fine-tune segmentation models and collect RGB and segmentation mask examples to adapt them.

PerSAM. Personalized SAM (PerSAM) (Zhang et al., 2023) is a segmentation model designed for personalized object segmentation building upon the Segment Anything Model (SAM) (Kirillov et al., 2023). This model is particularly a good fit for our SD use case since it can obtain a personalized segmentation model without additional training by one-shot adapting to a *single* in-domain image. In our experiments, we use the model with ViT-T as a backbone.

SegFormer. We use 5 or 10 pairs of examples to fine-tune SegFormer (Xie et al., 2021) MiT-b0.

To collect a one-shot in-domain RGB image and mask example for DMC and MetaWorld experiments, we sample a state from the initial distribution p_0 and render the RGB observation. In the few-shot scenario, we deploy a random agent in each environment to collect more diverse observations from reachable states.

To generate the associated masks for these states, we make additional queries to the simulation rendering API. We represent the pixel values for background and irrelevant objects as *false* and task-relevant objects as *true*. In multi-object cases, we may perform a separate adaptation operation for each task-relevant object, resulting in more than 2 mask classes. In such cases, before integrating masks with $\text{SD}^{\text{approx.}}$, we will combine the union of the mask classes for all pertinent objects as a single *true* task-relevant class, creating a binary segmentation mask compatible with our method.

In cases where example masks cannot be programmatically extracted, because such a small number of examples are required (1-10), it should also be very feasible for a human to use software to manually annotate the needed mask examples from collected RGB images.

L Details on Selective L_2 Loss

The binary mask prediction branch in $\text{SD}^{\text{approx.}}$ is equipped with the sigmoid layer at its output. In order to obtain binary mask_{SD}, we binarize the SD binary mask prediction with a threshold of 0.9.

M Details on Baselines

It is known that RePo (Zhu et al., 2023) outperforms many earlier works (Fu et al., 2021; Hansen et al., 2022; Zhang et al., 2021; Wang et al., 2022; Gelada et al., 2019) and that DreamerPro (Deng et al., 2022) surpasses TPC (Nguyen et al., 2021). However, theses two groups of works have been using slightly different environment setups and have not been compared with each other despite addressing the same high-level problem on the same DMC environments. In our experiments, we evaluate the representatives in each cluster on a common ground (See Appendix G) and compare them with our method.

In our experiments, we use hyperparameters used in the original papers for all the baselines, except RePo (Zhu et al., 2023) in Meta-World. RePo does not have experiments on Meta-World in which case we use hyperparameters used for Maniskill2 (Gu et al., 2023) which is another robot manipulation benchmark.

N Extended Related work

There are several model-based RL approaches which also explore the introduction of new auxiliary tasks. Dynalang (Lin et al., 2024) integrates language modeling as a self-supervised learning objective in world-model training. It shows impressive performance on benchmarks where the dynamics can be effectively described in natural language. However, it is not trivial to apply this method in low-level control scenarios such as locomotion control in DMC. Informed Dreamer (Lambrechts et al., 2024) introduces an information decoder which uses priviledged simulator information to decode a sufficient statistic for optimal control. This shares the idea of using additional information available at training time with our method SD$^{\text{GT}}$. Although it can be effective on training in simulation where well-shaped proprioceptive states exist, Informed Dreamer cannot be applied to cases where such information is hard to obtain. In goal-conditioned RL, GAP (Nair et al., 2020) proposes to decode the difference between the future state and goal state to help learn goal-relevant features in the latent state space.

O Limitations

Segmentation Dreamer achieves strong performance across diverse tasks in the presence of distractions and provides a human interface to indicate task relevance. This capability enables practitioners to readily train an agent for their specific purposes without suffering from poor learning performance due to visual distractions. However, there are several limitations to consider.

First, since SD$^{\text{approx.}}$ harnesses a segmentation model, it can become confused when a scene contains distractor objects that resemble task-relevant objects. This challenge can be mitigated by combining our method with approaches such as InfoPower (Bharadhwaj et al., 2022), which learns controllable representations through empowerment (Mohamed & Jimenez Rezende, 2015). This integration would help distinguish controllable task-relevant objects from those with similar appearances but move without agent interaction.

Second, although we provide a preliminary exploration in Appendix D, our method does not explicitly address randomization in the visual appearance of *task-relevant* objects, such as variations in brightness, illumination, or color. Two observations of the same internal state but with differently colored task-relevant objects may be guided toward different latent representations because our task-relevant "pixel-value" reconstruction loss forces them to be differentiated. Ideally, these observations should map to the same state abstraction since they exhibit similar behaviors in terms of the downstream task. Given that training with pixel-value perturbations on task-relevant objects is easier compared to dealing with dominating background distractors (Stone et al., 2021), our method is expected to manage such perturbations effectively without modifications. However, augmenting our approach with additional auxiliary tasks based on behavior similarity (Zhang et al., 2021) would further enhance representation learning and directly address this issue.

Finally, our approximation model faces scalability challenges when task-relevant objects constitute an open set. For instance, in autonomous driving scenarios, obstacles are task-relevant but cannot be explicitly specified. While our method serves as an effective solution when task-relevant objects are easily identifiable, complementary approaches should be considered when this assumption does not hold true.

P Code Release

We plan to make the code for Segmentation Dreamer (SD) publicly available upon acceptance.