STRUCTURING PRIVILEGED STATES WITH LLMS FOR ROBUST WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual world models enable control directly from high-dimensional visual observations through latent imagination, yet their learned representations can remain brittle under distribution shifts. While prior work studies generalization under complex visual distractions, we discover that simple changes to environment configurations and physical dynamics are sufficient to cause substantial performance degradation under zero-shot evaluation. Privileged states expose rich structural information about the underlying environment and have been broadly adopted to support robust generalization, but directly predicting raw states may require modeling task-irrelevant or visually inaccessible quantities, while manually designed auxiliary targets scale poorly across diverse tasks. In this work, we propose a framework that harnesses the reasoning capabilities and commonsense priors of Large Language Models (LLMs) to automatically structure privileged states into compact, task-relevant semantic properties. These LLM-derived properties serve as auxiliary targets that explicitly regularize the world model toward representations that remain stable across distribution shifts. To seamlessly integrate this supervision, we introduce an adaptive multi-loss calibration mechanism to dynamically balance the semantic prediction objective against the original world model losses, removing the need for task-specific manual weight tuning. Across Meta-World, DeepMind Control Suite, and CARLA, our approach consistently achieves substantial zero-shot generalization improvements over strong model-free and model-based baselines under both held-out configuration and physical dynamics shifts.

1 INTRODUCTION

Visual world models have emerged as a powerful paradigm for reinforcement learning (RL), enabling agents to learn compact latent dynamics from high-dimensional observations (Sutton, 1991; Sutton & Barto, 1998; Ha & Schmidhuber, 2018; Hafner et al., 2019a;b; 2020; 2025). By capturing the underlying temporal structure of the environment, these models allow agents to plan and optimize behaviors entirely through latent imagination. This imagination-driven approach has significantly advanced sample efficiency and enabled the successful control of complex simulated and embodied systems directly from raw pixels (Wu et al., 2023; Hansen et al., 2022). Ideally, learning a predictive model of the world should yield representations that support robust generalization to out-of-distribution (OOD) scenarios (Zhu et al., 2023; Röder et al., 2026).

In practice, world models often suffer substantial performance degradation under distribution shifts. Recent studies have extensively documented the struggles of world-model-based agents in complex OOD scenarios, where they fail to generalize across spurious visual variations, distracting backgrounds, or unseen environments (Deng et al., 2022; Anand et al., 2021; Zhu et al., 2023). However, we highlight that this representational fragility can be exposed by a much simpler setting: directly perturbing the simulator's internal parameters to modify spatial configurations and physical dynamics, such as motor strength and friction. In Figure 1, this fragility is illustrated both empirically, as episode returns degrade significantly during zero-shot evaluation, and intuitively, as the model exhibits critical failure modes that are rarely observed during training.

Leveraging privileged states during training is an established way to facilitate visual policy learning and robust generalization (Pinto et al., 2017; Hu et al., 2024; Salter et al., 2021). However, blindly predicting raw privileged states can require inferring information unavailable from visual observations,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

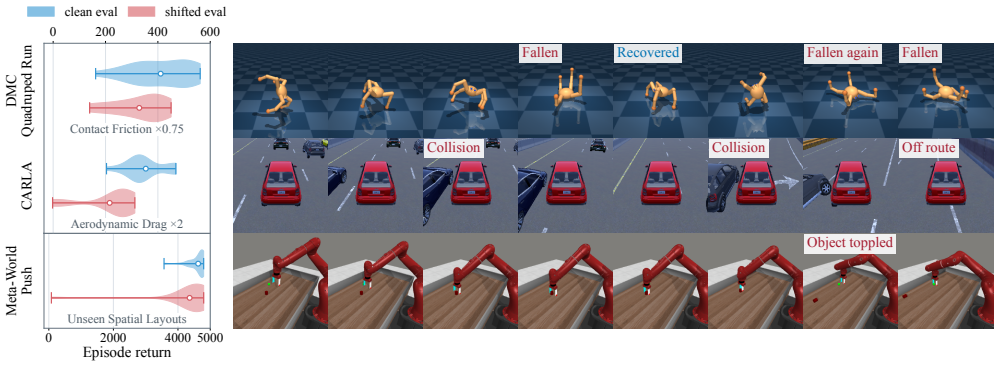

Figure 1: Simple shifts expose severe generalization failures across visual control domains. Left: episode-return distributions under clean and shifted evaluation conditions for DMC Quadruped Run with reduced contact friction, CARLA with increased aerodynamic drag, and Meta-World Push under unseen spatial layouts. These shifts modify only simulator configurations, without introducing artificial visual distractors. Right: Sample rollouts from perturbed environments reveal characteristic failures, including loss of balance, collisions and off-route driving, and object toppling.

creating an information gap that limits the utility of the privileged supervision (Lambrechts et al., 2023; 2026). Manually designed auxiliary targets offer a targeted alternative (Jonschkowski & Brock, 2015; Jaderberg et al., 2016), but rely heavily on human heuristics that fundamentally fail to scale across diverse tasks. This raises a critical question: how can we systematically specify which semantically meaningful properties a world model should capture to facilitate robust zero-shot generalization?

In this work, we propose a framework that guides world models toward robust representations by harnessing the reasoning capabilities and commonsense priors of Large Language Models (LLMs). LLMs have proven highly capable of translating high-level task semantics into executable control, reward, and state-representation functions (Liang et al., 2023; Ma et al., 2024b; Wang et al., 2024), and have increasingly automated the design of task-specific components that would otherwise require substantial human engineering (Xie et al., 2024; Wang et al., 2023). Building on this insight, our framework utilizes LLMs to structure privileged states into semantically meaningful properties. These LLM-derived properties serve as privileged semantic targets to explicitly regularize the latent space during world model training. To seamlessly integrate this supervision without destabilizing the world model, we introduce an adaptive multi-loss calibration mechanism that dynamically balances the semantic prediction loss with the original world-model objectives. By mitigating loss imbalance and preventing the auxiliary supervision from either dominating or being overshadowed during optimization, this mechanism removes the need for task-specific manual tuning of the semantic loss weight. Finally, the semantic guidance is used exclusively during training, preserving a purely pixel-based policy at deployment.

In summary, the main contributions of this work are threefold:

- We uncover a simple yet challenging source of generalization failure in visual RL. Without introducing artificial distractions, merely perturbing simulator configurations or physical parameters is sufficient to substantially degrade the performance of strong existing baselines.
- We propose an LLM-guided framework that automatically structures privileged states into task-relevant semantic properties for world-model supervision. These semantic targets are integrated into latent dynamics learning through an adaptive multi-loss calibration mechanism, avoiding task-specific manual loss weighting while preserving purely pixel-based deployment.
- We conduct extensive evaluations across Meta-World, DeepMind Control Suite, and CARLA, covering robotic manipulation, locomotion, and autonomous driving under both configuration and physical dynamics shifts. Our approach consistently achieves substantial generalization improvements over both strong model-free and model-based baselines.

2 Related Work

2.1 Visual World Models and Representation Learning

Visual world models learn compact latent dynamics for planning and policy optimization through imagination (Hafner et al., 2019a;b; 2020; 2025; Hansen et al., 2022; 2024). A central challenge is to retain predictive information while suppressing nuisance variation. One line of work promotes selective representations: Denoised MDPs disentangle controllable and reward-relevant factors from distractors, while RePo regularizes posterior predictability (Wang et al., 2022; Zhu et al., 2023). Another direction redesigns representation-learning objectives: Masked World Models decouple visual representation learning from latent dynamics through masked modeling and reward prediction (Seo et al., 2023), whereas reconstruction-free methods employ prototypes, value and action prediction, redundancy reduction, or temporal prediction (Deng et al., 2022; Burchi & Timofte, 2024; Morihira et al., 2026; Sun et al., 2024; Bredis et al., 2026). Furthermore, DALI models latent dynamics context for generalization under physical variations (Röder et al., 2026). Despite their diverse formulations, these approaches primarily derive supervision from observations, rewards, and interaction histories. Our approach complements these endogenous signals with LLM-guided auxiliary targets derived from privileged states, explicitly specifying the task-relevant information that the world model should preserve.

2.2 Learning with Privileged Information in RL

Learning with privileged information (LUPI) exploits signals available during training but unavailable at deployment (Vapnik & Vashist, 2009). In RL, privileged information supports asymmetric actor-critic learning, teacher-student distillation, and rapid adaptation (Pinto et al., 2017; Chen et al., 2020; Kumar et al., 2021). World models also exploit these signals: Informed Dreamer and Scaffolder use additional information for representation learning and training-time modeling (Lambrechts et al., 2023; Hu et al., 2024), while Reinformed Dreamer addresses the information gap between privileged targets and observable inputs (Lambrechts et al., 2026). Their utility depends on how supervision is specified: wholesale prediction may involve irrelevant or visually inaccessible quantities, whereas hand-designed auxiliary objectives demand unscalable, domain-specific engineering (Lambrechts et al., 2026; Jaderberg et al., 2016). We automate this specification by using LLMs to select and compose semantic properties from privileged states as structured supervision for the world model.

2.3 Large Language Models in RL

Large Language Models can reduce manual engineering by translating task semantics into executable learning components. Applications range from control-program synthesis in Code as Policies to reward generation in Text2Reward and Eureka (Liang et al., 2023; Xie et al., 2024; Ma et al., 2024b). DrEureka extends such automation to sim-to-real transfer through reward and domain-randomization design (Ma et al., 2024a). Beyond control and rewards, LESR generates state-representation functions for model-free RL (Wang et al., 2024), while DreamTIP studies LLM-assisted representation learning for sim-to-real quadruped locomotion (Liang et al., 2026). Our framework builds on this representation-specification perspective, using privileged simulator states to construct semantic targets for visual world-model generalization under configuration and physical dynamics shifts.

3 Method

Our framework enhances the representation learning of visual world models by grounding the latent space in structured semantic priors. Rather than designing entirely new sequence models, we instantiate our approach atop the well-established DreamerV3 architecture (Hafner et al., 2025) to demonstrate the efficacy of our proposed supervision. In this section, we first introduce the latent dynamics model augmented with a lightweight semantic predictor (Section 3.1). We then detail how LLMs programmatically extract these structural targets from raw privileged states (Section 3.2), and finally present the adaptive multi-loss calibration mechanism that seamlessly integrates this auxiliary supervision without destabilizing standard world model objectives (Section 3.3).

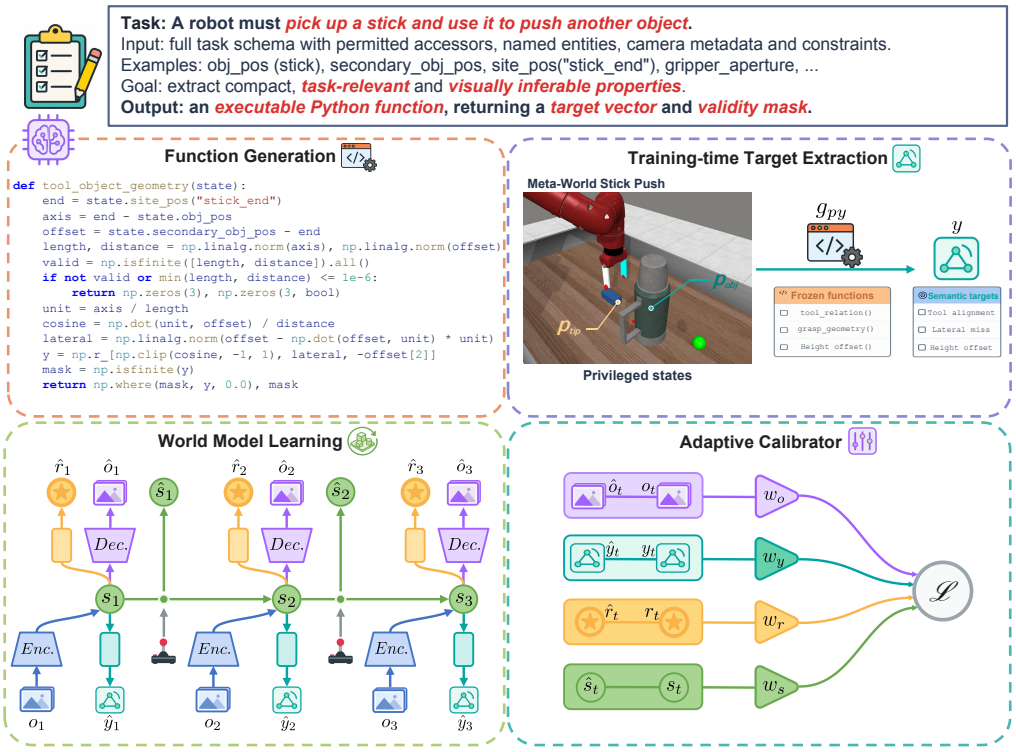

Figure 2: An overview of our proposed framework. LLMs generate a Python program to dynamically extract semantic targets from privileged states (Top). These targets provide auxiliary supervision for world-model learning, while an adaptive multi-loss calibrator balances the semantic and standard world-model objectives (Bottom).

3.1 Latent Dynamics Model and Semantic Prediction

Following DreamerV3, we utilize the Recurrent State-Space Model (RSSM) with a composite latent state $s_t = (h_t, z_t)$, consisting of a deterministic state h_t and a stochastic state z_t. By recursively compressing the historical sequence of high-dimensional observations o_t and actions a_t, this composite state serves as a compact Markovian summary that captures the underlying temporal evolution of the environment.

While standard visual world models rely heavily on a dense image decoder to shape the latent space, risking the encoding of task-irrelevant visual details, our architecture introduces a lightweight semantic predictor as depicted in the World Model Learning module of Figure 2. Specifically, this predictor is parameterized as a Multi-Layer Perceptron (MLP) with two hidden layers. This predictor maps the latent state s_t to a structured semantic space y_t, which encodes generalizable features programmatically extracted by LLMs (detailed in Section 3.2). By explicitly predicting these semantic properties, the latent space is regularized to internalize invariant task structures, expected to enhance zero-shot generalization under distribution shifts. The components of our augmented world model are defined as follows:

$$
\begin{aligned}
&\text{Image Encoder:} && e_t = f_\phi(o_t) \\
&\text{Sequence Model:} && h_t = f_\phi(s_{t-1}, a_{t-1}) \\
&\text{Dynamics Predictor:} && \hat{z}_t \sim p_\phi(\hat{z}_t \mid h_t) \\
&\text{Representation Model:} && z_t \sim q_\phi(z_t \mid h_t, e_t) \\
&\text{Reward Predictor:} && \hat{r}_t \sim p_\phi(\hat{r}_t \mid s_t) \\
&\text{Continue Predictor:} && \hat{c}_t \sim p_\phi(\hat{c}_t \mid s_t) \\
&\text{Observation Decoder:} && \hat{o}_t \sim p_\phi(\hat{o}_t \mid s_t) \\
&\text{Semantic Predictor:} && \hat{y}_t = g_\phi(h_t, z_t)
\end{aligned}
\tag{1}
$$

3.2 LLM-POWERED SEMANTIC PROPERTY EXTRACTION AND GROUNDING

To operationalize this LLM-guided semantic extraction, we leverage LLMs offline to synthesize semantic target extraction functions prior to world model training (as illustrated in Figure 2). Specifically, for a given task, we provide the LLM with a task prompt containing the high-level task schema, permitted simulator state accessors, named entities, and generation constraints. The LLM is tasked with generating an executable Python function, g_{py}. This offline generation process is performed only once per task, incurring no LLM query overhead during the RL training loop.

During the training phase, this frozen program acts as a structural filter, mapping raw privileged states x_t into a compact semantic vector y_t^{raw} alongside a binary validity mask m_t:

$$y_t^{\text{raw}}, m_t = g_{\text{py}}(x_t) \tag{2}$$

This mask explicitly excludes invalid components within y_t^{raw} corresponding to uncomputable geometric relations or temporarily absent entities at any given time step. By design, this programmatic extraction injects structured relational properties into the auxiliary targets. Instead of relying on absolute state variables (e.g., global coordinates and orientations) that may easily overfit to specific settings, we find that the generated functions intuitively compute robust relative abstractions—such as pairwise geometric distances, spatial alignments, and topological indicators.

Because the dimensions of y_t^{raw} may represent heterogeneous quantities and vary substantially in numerical scale, direct prediction is often challenging and prone to instability. To address this scale disparity, we apply a two-stage normalization independently to each target dimension. First, following DreamerV3, we apply a symlog transformation to compress extreme outliers while maintaining resolution for small values:

$$v_t = \text{sign}\left(y_t^{\text{raw}}\right) \odot \ln\left(1 + |y_t^{\text{raw}}|\right) \tag{3}$$

While the symlog transformation effectively compresses extreme magnitudes, the transformed dimensions may still exhibit varying statistical scales. We then standardize v_t using running mean and variance vectors, μ and ν, maintained via an exponential moving average (EMA) of their batch statistics. The final normalized target is computed as:

$$y_t = \frac{v_t - \mu}{\sqrt{\nu + \epsilon}} \tag{4}$$

With the targets robustly scaled, we train our Semantic Predictor $g_\phi(h_t, z_t)$ using a masked Mean Squared Error (MSE) objective:

$$\mathcal{L}_{desc} = \frac{1}{\max\left(1, \sum_{t,j} m_{t,j}\right)} \sum_{t,j} m_{t,j} \left(y_{t,j} - g_\phi(h_t, z_t)_j\right)^2 \tag{5}$$

Crucially, because g_{py} is strictly executed on privileged states during training, this entire semantic extraction and grounding process operates purely as an auxiliary regularization mechanism. It introduces zero computational overhead during deployment, ensuring the agent operates entirely on the learned pixel-based policy.

3.3 ADAPTIVE MULTI-LOSS CALIBRATION

While programmatic extraction automates semantic target specification, the magnitude and optimization dynamics of the resulting prediction loss $\mathcal{L}_{\text{desc}}$ can still vary substantially across domains. Assigning a static coefficient to this auxiliary objective would inevitably reintroduce task-specific manual tuning. To overcome this, we adaptively calibrate the semantic objective jointly with the standard world-model losses during training.

Following the rectified learnable loss-scaling mechanism (Ma et al., 2023), we associate each objective $\mathcal{L}$ with a learnable log-scale parameter s. The calibrated loss component is formulated as:

$$H_\rho(\mathcal{L}, s) = \exp(-s)\mathcal{L} + \rho \ln\left(1 + \exp(s)\right), \tag{6}$$

Table 1: Distribution-shift protocols. Training conditions and zero-shot evaluation settings across domains.

Benchmark	Training	Zero-shot evaluation
Meta-World	50 object/goal configurations	50 held-out object/goal configurations
DMC	Default dynamics	Motor strength and contact friction $\times\{0.50, 0.75, 1.25, 1.50\}$; matched compound shifts
CARLA	Default dynamics	Drag $\times\{0.50, 0.75, 2.00, 4.00\}$; friction and torque $\times\{0.60, 0.80, 1.20, 1.40\}$; matched compound shifts

where the exponential term adaptively modulates the gradient contribution of the objective, and the softplus regularizer prevents the network from trivially minimizing the loss by pushing $s \to \infty$. Importantly, ρ is not an additional free hyperparameter, but a normalizing coefficient derived from the underlying loss grouping. All scale parameters are initialized to zero and optimized concurrently with the world model.

Let $\mathcal{L}_{\text{obs}}$, $\mathcal{L}_{\text{rew}}$, $\mathcal{L}_{\text{dyn}}$, $\mathcal{L}_{\text{rep}}$, and $\mathcal{L}_{\text{cont}}$ denote the observation reconstruction, reward, dynamics KL, representation KL, and continue prediction losses from DreamerV3, respectively. To maintain the intrinsic balance between the two KL objectives, we group them into a unified dynamics loss. Denoting the fixed representation-loss weight inherited from DreamerV3 as λ_{rep}, we define:

$$\mathcal{L}_d = \frac{\mathcal{L}_{\text{dyn}} + \lambda_{\text{rep}}\mathcal{L}_{\text{rep}}}{1 + \lambda_{\text{rep}}}. \tag{7}$$

This grouped objective shares a single learnable scale s_d. Because the softplus regularizer is applied only once to this combined term, the corresponding coefficient in H_ρ is $1/(1+\lambda_{\text{rep}})$. The observation, reward, and semantic prediction losses are assigned independent scales s_o, s_r, and s_y, respectively. Consistent with standard DreamerV3 settings, the continue loss $\mathcal{L}_{\text{cont}}$ retains its fixed original weight and is excluded from adaptive calibration.

The resulting world-model objective is therefore constructed as:

$$\mathcal{L}_{\text{model}} = H_1(\mathcal{L}_{\text{obs}}, s_o) + H_1(\mathcal{L}_{\text{rew}}, s_r) + H_{\frac{1}{1+\lambda_{\text{rep}}}}(\mathcal{L}_d, s_d) + H_1(\mathcal{L}_{\text{desc}}, s_y) + \mathcal{L}_{\text{cont}}. \tag{8}$$

By jointly learning the relative scale of the semantic objective alongside generative modeling terms, our framework organically incorporates task-dependent physical priors without requiring any per-task hyperparameter tuning. Note that the actor-critic objectives follow their original optimization scheme and remain strictly independent of this calibration process.

4 Experiments

4.1 Experimental Settings

Distribution Shift Benchmarks. We evaluate zero-shot generalization robustness by systematically perturbing internal simulator parameters while keeping the overarching task semantics consistent. We deploy three distinct benchmarks encompassing robotic manipulation, continuous locomotion, and autonomous driving, with exact training configurations and zero-shot evaluation perturbations summarized in Table 1. For manipulation, we follow Masked World Models (Seo et al., 2023) by selecting 11 Hard/Very Hard Meta-World tasks (Yu et al., 2020), uniformly training agents for 1M steps before evaluating on held-out spatial layouts. For continuous locomotion, we target five DeepMind Control Suite (DMC) tasks (Tassa et al., 2018) where standard DreamerV3 notably struggles under altered dynamics. Finally, we use the highway driving scenario in CARLA, a high-fidelity simulator that rigorously tests the agent's ability to deal with complex visual inputs (Dosovitskiy et al., 2017), exposing models to severe compound physical degradations.

Implementation Details. We instantiate our framework atop the DreamerV3, retaining its default network structures and training hyperparameters unless otherwise specified. The newly introduced semantic predictor is parameterized as a lightweight MLP consisting of two hidden layers. For normalization,

Table 2: Zero-shot generalization on Meta-World at 1M environment steps. Success rates (%) on held-out spatial configurations. The best results are highlighted in bold.

Task	SAC	DrQ-v2	TACO	MaDi	Dreamer V3	Harmony Dream	DALI	R2-Dreamer	Ours
Assembly	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	**6.7** ± 9.4	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Hand Insert	0.0 ± 0.0	7.3 ± 5.2	20.7 ± 7.4	12.7 ± 10.6	35.3 ± 17.0	58.7 ± 11.5	50.0 ± 14.7	60.7 ± 16.4	**92.0** ± 4.3
Pick Out of Hole	0.0 ± 0.0	23.3 ± 6.2	34.7 ± 23.3	33.3 ± 17.2	51.3 ± 37.4	70.7 ± 5.2	65.3 ± 5.0	74.7 ± 10.0	**93.3** ± 3.4
Pick Place	6.7 ± 6.8	30.0 ± 9.1	32.7 ± 8.4	28.7 ± 15.2	68.0 ± 14.2	76.0 ± 4.9	72.7 ± 8.1	76.0 ± 4.3	**91.3** ± 9.4
Push	0.0 ± 0.0	24.7 ± 2.5	26.7 ± 0.9	33.3 ± 7.7	68.7 ± 4.1	77.3 ± 4.1	73.3 ± 7.7	73.3 ± 9.6	**90.0** ± 14.1
Push Back	0.0 ± 0.0	0.0 ± 0.0	14.7 ± 14.3	25.3 ± 29.0	0.0 ± 0.0	45.3 ± 27.8	48.7 ± 15.4	72.7 ± 6.8	**92.7** ± 9.0
Shelf Place	0.0 ± 0.0	11.3 ± 9.0	60.0 ± 9.1	57.3 ± 19.8	62.0 ± 43.9	62.7 ± 32.9	63.3 ± 38.6	50.7 ± 19.1	**66.0** ± 46.7
Disassemble	0.0 ± 0.0	20.0 ± 20.4	21.3 ± 15.2	32.7 ± 39.3	21.3 ± 9.8	51.3 ± 37.4	44.0 ± 38.6	44.7 ± 10.6	**52.0** ± 29.0
Stick Pull	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	18.0 ± 25.5	0.0 ± 0.0	30.7 ± 35.2	27.3 ± 37.2	33.3 ± 5.7	**53.3** ± 41.1
Stick Push	0.0 ± 0.0	0.0 ± 0.0	14.7 ± 20.7	28.0 ± 31.5	46.0 ± 14.0	71.3 ± 6.2	66.7 ± 31.3	64.0 ± 12.3	**100.0** ± 0.0
Pick Place Wall	1.3 ± 0.9	2.7 ± 3.8	9.3 ± 3.8	22.0 ± 17.2	0.7 ± 0.9	76.0 ± 12.8	68.0 ± 17.0	80.0 ± 8.6	**92.0** ± 3.3
Average	0.7	10.8	21.3	27.1	32.1	56.4	52.7	57.3	**74.8**

the running mean and variance are updated via an exponential moving average (EMA) with a decay rate of 0.99, incorporating a small constant $\epsilon = 10^{-4}$ to ensure numerical stability. All experimental results are aggregated across 5 independent random seeds, with the mean performance reported.

4.2 Zero-Shot Generalization Results

For comparison, we evaluate against a suite of representative prior methods, including both model-free and model-based agents. The model-free baselines include SAC (Haarnoja et al., 2018), DrQ-v2 (Yarats et al., 2021), TACO (Zheng et al., 2023), and MaDi (Grooten et al., 2023). The model-based baselines include the original DreamerV3 (Hafner et al., 2025), HarmonyDream (Ma et al., 2023), DALI (Röder et al., 2026), and R2-Dreamer (Morihira et al., 2026).

Meta-World. We first evaluate our method on contact-rich manipulation tasks in the Meta-World benchmark, which require robust spatial understanding and precise control under novel scene layouts. Table 2 demonstrates that our method significantly outperforms all baselines on the primary metric of zero-shot success rate. This superior performance is also reflected in the agent's broad mastery, achieving best results across 10 out of the 11 evaluated tasks. We hypothesize that standard visual world models are highly susceptible to overfitting to absolute pixel arrangements. Our semantic regularization mechanism is designed to counteract this failure mode by explicitly grounding the latent space in invariant relative geometries, leading to a more robust and effective zero-shot manipulation policy.

DMC. We further evaluate our method on continuous locomotion tasks in the DMC to assess robustness against zero-shot physical parameter shifts. As summarized in Table 3, our approach achieves state-of-the-art performance with an average return of 488.5, yielding a substantial improvement over the baselines. This commanding advantage is consistent across the benchmark, securing the highest returns in 4 out of the 5 evaluated tasks.

CARLA Autonomous Driving. Finally, we evaluate our method in the high-fidelity CARLA simulator to test zero-shot generalization under strict sample constraints (100k training steps) and physical dynamics variations. As detailed in Table 4, our approach achieves state-of-the-art performance, outperforming all evaluated baselines by a substantial margin across all shifted conditions. However,

Table 3: Zero-shot generalization on DMC at 1M environment steps. Episode returns averaged over 12 held-out physical dynamics shifts. The best results are highlighted in **bold**.

Method	Quadruped Walk	Quadruped Run	Walker Run	Quadruped Escape	Hopper Hop	Avg.
SAC	26.7 ± 12.5	16.0 ± 5.4	16.3 ± 4.2	22.3 ± 9.5	0.0 ± 0.0	16.3
DrQ-v2	509.0 ± 58.2	375.3 ± 23.9	421.0 ± 40.0	54.3 ± 24.6	104.3 ± 13.3	292.8
TACO	568.0 ± 57.4	441.3 ± 16.8	526.0 ± 27.8	57.7 ± 14.3	151.0 ± 24.6	348.8
MaDi	450.7 ± 43.3	403.3 ± 21.6	530.7 ± 52.9	66.3 ± 19.7	130.0 ± 22.6	316.2
DreamerV3	725.0 ± 78.9	406.0 ± 26.5	501.7 ± 41.3	96.5 ± 32.4	171.5 ± 85.2	380.1
HarmonyDream	773.7 ± 15.1	555.0 ± 44.2	512.7 ± 23.1	99.7 ± 9.0	167.3 ± 10.3	421.7
DALI	783.0 ± 15.3	519.0 ± 34.0	$\mathbf{597.6} \pm 38.5$	103.0 ± 13.7	158.7 ± 21.5	432.3
R2-Dreamer	733.3 ± 54.8	530.7 ± 30.9	523.0 ± 47.4	101.3 ± 9.8	153.7 ± 24.4	408.4
Ours	$\mathbf{847.6} \pm 18.6$	$\mathbf{652.3} \pm 19.3$	588.4 ± 35.6	$\mathbf{153.3} \pm 18.7$	$\mathbf{201.0} \pm 29.9$	**488.5**

the results also highlight the exceptional difficulty of the *Compound* shift setting, where simultaneous perturbations to multiple physical parameters cause even advanced world models to degrade significantly. This reveals that robustly disentangling underlying kinematics from high-dimensional visual renderings under multifaceted distribution shifts remains an open problem, marking this challenging setting as a critical direction for future research.

Table 4: Zero-shot generalization on CARLA at 100k training steps. Episode returns evaluated under dynamics shifts. The best results are highlighted in **bold**.

Method	Drag	Friction	Torque	Compound	Shift Avg.
SAC	80.0 ± 28.9	65.0 ± 33.2	77.0 ± 30.5	42.0 ± 29.9	66.0
DrQ-v2	126.0 ± 39.0	110.0 ± 45.0	133.0 ± 29.7	78.0 ± 31.0	111.8
TACO	144.0 ± 35.3	129.0 ± 40.2	155.0 ± 35.8	109.0 ± 50.1	134.3
MaDi	140.0 ± 36.4	122.0 ± 46.7	108.0 ± 58.3	92.0 ± 42.9	115.5
DreamerV3	206.1 ± 66.2	221.4 ± 44.8	200.4 ± 77.0	166.5 ± 72.3	198.6
HarmonyDream	289.6 ± 44.7	282.0 ± 52.2	287.0 ± 34.6	204.0 ± 44.3	265.7
DALI	316.0 ± 56.9	299.0 ± 54.4	254.0 ± 38.1	225.0 ± 45.8	273.5
R2-Dreamer	300.2 ± 55.0	332.4 ± 45.1	316.9 ± 40.3	255.3 ± 43.5	301.2
Ours	$\mathbf{361.7} \pm 36.8$	$\mathbf{387.5} \pm 50.1$	$\mathbf{366.4} \pm 30.3$	$\mathbf{346.1} \pm 37.0$	**365.4**

4.3 Ablation Study

Component Contribution. Figure 3(a) shows that while incorporating LLM-generated semantic targets (+*target*) or adaptive multi-loss calibration (+*calibration*) individually improves upon the baseline, their combination yields the highest returns. This highlights a strong synergy: stable semantic representations provide a reliable foundation for the adaptive multi-loss calibration to effectively guide optimization.

LLM Capability. We vary the underlying offline LLM across the GPT family (Figure 3(b)). Zero-shot performance improves substantially from GPT-3.5 to GPT-4o, but saturates thereafter. Frontier models (e.g., GPT-5, GPT-5.6 Sol) yield marginal improvements. This demonstrates that the effectiveness of our framework does not heavily rely on the advanced reasoning capabilities of frontier LLMs. For consistency, all main experiments in this paper default to GPT-5.6 Sol.

Calibration Effectiveness. Figure 3(c) compares our adaptive multi-loss calibration against fixed-coefficient variants ($0.1, 1$, and 5). Our dynamically calibrated agent consistently achieves higher final returns. Adaptively modulating the regularization strength effectively avoids the sub-optimal trade-offs of manual hyperparameter tuning.

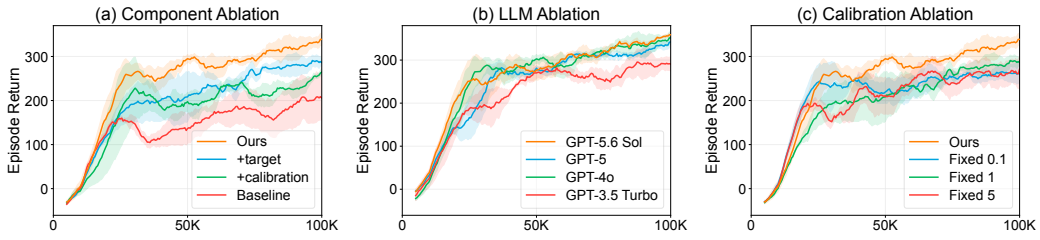

Figure 3: Ablation studies on CARLA. (a) Ablation on the effectiveness of each component. (b) Ablation on the capability of the underlying LLMs across the GPT family. (c) Ablation on the effectiveness of adaptive loss calibration versus fixed-parameter settings.

4.4 Discussion and Analysis

Representation Linear Probing. To assess latent space quality, we perform linear probing by training regressors on frozen latent states to predict critical physical metrics: TCP-object and object-goal distances (Figure 4(a)). Our method achieves lower NMSE than DreamerV3, indicating that LLM-generated semantic targets effectively guide the world model to encode task-critical semantics.

Visual Attention Maps. We further visualize the spatial attention maps under distribution shifts (Figure 4(b)). When encountering novel physical scenes, the standard DreamerV3 exhibits scattered and uninformative attention patterns, often completely losing track of critical objects (e.g., the target object in Meta-World or the agent's body structure in DMC). In contrast, our method maintains sharp, highly concentrated attention on essential kinematic entities despite severe distribution shifts. This confirms that structurally anchoring the latent space with semantic descriptors explicitly guides the visual encoder to filter out irrelevant background noise and robustly track invariant features.

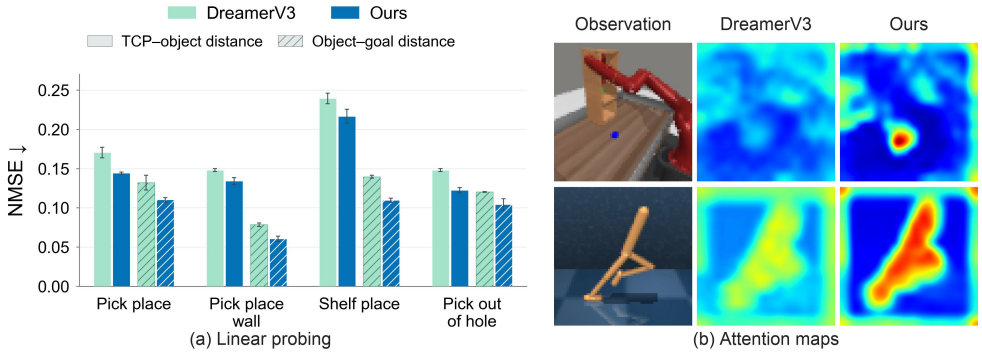

Figure 4: (a) Linear-probe NMSE for predicting TCP–object distance and Object–goal distance. (b) Visual attention maps under distribution shifts.

5 Conclusion

We proposed an LLM-guided framework that structures privileged states into compact semantic properties to improve the robustness of visual world models under distribution shifts. Our method uses offline LLM-generated functions to extract task-relevant auxiliary targets for grounding the latent representation, together with adaptive multi-loss calibration to integrate this supervision without task-specific loss tuning while preserving purely pixel-based deployment. Extensive experiments across Meta-World, DeepMind Control Suite, and CARLA demonstrated substantial improvements over strong model-free and model-based baselines under held-out configuration and physical dynamics shifts, highlighting that the value of privileged information lies not in predicting more state variables, but in identifying the structure that robust world models should preserve.

AI USE STATEMENT

Generative AI tools were used as part of the proposed framework to generate task-specific semantic extractors, as described in Section 3 and Appendix D. Generative AI tools were also used during the preparation of this work to assist with language editing and improving the clarity and organization of the manuscript, as well as for literature search and summarization. They were also used to provide feedback on the presentation of mathematical formulations. All AI-generated suggestions were carefully assessed by the authors before being incorporated into the manuscript. Relevant literature suggested by AI tools was manually verified against the original sources, and all AI-assisted text and technical content were reviewed and revised by the authors. The authors take full responsibility for the final content of this work, including all text, claims, results, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed information to facilitate the reproduction of our results. Section 3 specifies the proposed framework and adaptive multi-loss calibration, including the corresponding training objectives. Section 4.1 describes the benchmark settings and distribution-shift protocols. Appendix B provides additional environment configurations, implementation details, and computational resources, while Appendix C reports the hyperparameter settings used across all experiments. Appendix D further presents the LLM prompt design and representative semantic extractor implementations across Meta-World, DeepMind Control Suite, and CARLA. Anonymous source code is also provided in the supplementary materials. Together, these materials document the methodological and experimental details required to reproduce the reported results.

REFERENCES

Ankesh Anand, Jacob Walker, Yazhe Li, Eszter Vértes, Julian Schrittwieser, Sherjil Ozair, Theophane Weber, and Jessica B Hamrick. Procedural generalization by planning with self-supervised world models. *arXiv preprint arXiv:2111.01587*, 2021.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM international conference on architectural support for programming languages and operating systems, volume 2*, pp. 929–947, 2024.

George Bredis, Nikita Balagansky, Daniil Gavrilov, and Ruslan Rakhimov. Next embedding prediction makes world models stronger. *arXiv preprint arXiv:2603.02765*, 2026.

Maxime Burchi and Radu Timofte. Mudreamer: Learning predictive world models without reconstruction. *arXiv preprint arXiv:2405.15083*, 2024.

Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on robot learning*, pp. 66–75. PMLR, 2020.

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.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pp. 1–16. PMLR, 2017.

Bram Grooten, Tristan Tomilin, Gautham Vasan, Matthew E Taylor, A Rupam Mahmood, Meng Fang, Mykola Pechenizkiy, and Decebal Constantin Mocanu. Madi: Learning to mask distractions for generalization in visual deep reinforcement learning. *arXiv preprint arXiv:2312.15339*, 2023.

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

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

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, 2019b.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

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

Nick Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, volume 2024, pp. 47376–47405, 2024.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

Edward Hu, James Springer, Oleh Rybkin, and Dinesh Jayaraman. Privileged sensing scaffolds reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 2608–2631, 2024.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016.

Rico Jonschkowski and Oliver Brock. Learning state representations with robotic priors. *Autonomous Robots*, 39(3):407–428, 2015.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

Gaspard Lambrechts, Adrien Bolland, and Damien Ernst. Informed pomdp: Leveraging additional information in model-based rl. *arXiv preprint arXiv:2306.11488*, 2023.

Gaspard Lambrechts, Adrien Bolland, Daniel Ebi, and Damien Ernst. Reinformed dreamer: An asymmetric world model efficiently trained through latent guidance. *arXiv preprint arXiv:2607.26040*, 2026.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International conference on robotics and automation (ICRA)*, pp. 9493–9500. IEEE, 2023.

Junyang Liang, Yuxuan Liu, Yabin Chang, Junfan Lin, Junkai Ji, Hui Li, Changxin Huang, and Jianqiang Li. Learning task-invariant properties via dreamer: Enabling efficient policy transfer for quadruped robots. *arXiv preprint arXiv:2604.02911*, 2026.

Haoyu Ma, Jialong Wu, Ningya Feng, Chenjun Xiao, Dong Li, Jianye Hao, Jianmin Wang, and Mingsheng Long. Harmonydream: Task harmonization inside world models. *arXiv preprint arXiv:2310.00344*, 2023.

Jason Ma, William Liang, Hung-Ju Wang, Yuke Zhu, Linxi Fan, Osbert Bastani, and Dinesh Jayaraman. Dreureka: Language model guided sim-to-real transfer. In *Robotics: Science and Systems*, 2024a.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Jim Fan, et al. Eureka: Human-level reward design via coding large language models. In *International conference on learning Representations*, volume 2024, pp. 26516–26560, 2024b.

Naoki Morihira, Amal Nahar, Kartik Bharadwaj, Yasuhiro Kato, Akinobu Hayashi, and Tatsuya Harada. R2-dreamer: Redundancy-reduced world models without decoders or augmentation. *arXiv preprint arXiv:2603.18202*, 2026.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542*, 2017.

Frank Röder, Jan Benad, Manfred Eppe, and Pradeep Banerjee. Dynamics-aligned latent imagination in contextual world models for zero-shot generalization. *Advances in Neural Information Processing Systems*, 38:72736–72771, 2026.

Sasha Salter, Dushyant Rao, Markus Wulfmeier, Raia Hadsell, and Ingmar Posner. Attention-privileged reinforcement learning. In *Conference on Robot Learning*, pp. 394–408. PMLR, 2021.

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, 2023.

Ruixiang Sun, Hongyu Zang, Xin Li, and Riashat Islam. Learning latent dynamic robust representations for world models. *arXiv preprint arXiv:2405.06263*, 2024.

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

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

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.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural networks*, 22(5-6):544–557, 2009.

Boyuan Wang, Yun Qu, Yuhang Jiang, Jianzhun Shao, Chang Liu, Wenming Yang, and Xiangyang Ji. Llm-empowered state representation for reinforcement learning. *arXiv preprint arXiv:2407.13237*, 2024.

Tongzhou Wang, Simon S Du, Antonio Torralba, Phillip Isola, Amy Zhang, and Yuandong Tian. Denoised mdps: Learning world models better than the world itself. *arXiv preprint arXiv:2206.15477*, 2022.

Yufei Wang, Zhou Xian, Feng Chen, Tsun-Hsuan Wang, Yian Wang, Katerina Fragkiadaki, Zackory Erickson, David Held, and Chuang Gan. Robogen: Towards unleashing infinite data for automated robot learning via generative simulation. *arXiv preprint arXiv:2311.01455*, 2023.

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

Tianbao Xie, Siheng Zhao, Chen Wu, Yitao Liu, Qian Luo, Victor Zhong, Yanchao Yang, and Tao Yu. Text2reward: Reward shaping with language models for reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 35663–35699, 2024.

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*, pp. 1094–1100. PMLR, 2020.

Ruijie Zheng, Xiyao Wang, Yanchao Sun, Shuang Ma, Jieyu Zhao, Huazhe Xu, Hal Daumé III, and Furong Huang. Taco: Temporal latent action-driven contrastive loss for visual reinforcement learning. *Advances in Neural Information Processing Systems*, 36:48203–48225, 2023.

Chuning Zhu, Max Simchowitz, Siri Gadipudi, and Abhishek Gupta. Repo: Resilient model-based reinforcement learning by regularizing posterior predictability. *Advances in Neural Information Processing Systems*, 36:32445–32467, 2023.

A VISUALIZATION

We qualitatively analyze the DreamerV3 baseline under zero-shot shifts. Standard visual world models tend to overfit to default training dynamics and absolute layouts. Consequently, environment perturbations render their internalized dynamics inaccurate, leading to catastrophic errors (Figure A.1).

DeepMind Control Suite (DMC). Under weakened motor actuation (motor strength $\times 0.5$, top rows), the baseline fails to adapt its locomotion gait due to a lack of invariant postural representations. In tasks like `Walker Run`, `Hopper Hop` and `Quadruped Escape`, this mismatch rapidly destabilizes the agent, causing it to fall heavily without recovery.

CARLA. Facing friction shifts (friction $\times 1.4$, middle rows), the baseline severely misjudges braking distances and steering limits. This kinematic distortion results in loss of control, manifesting as dangerous off-route driving, severe tilting, and collisions with walls or dynamic vehicles.

Meta-World. Evaluated on unseen spatial layouts (bottom rows), the baseline struggles because it memorizes absolute pixel coordinates rather than true geometric relationships. This leads to severely degraded manipulation primitives, causing the agent to miss initial grasps, topple objects, or push targets out of the reachable workspace.

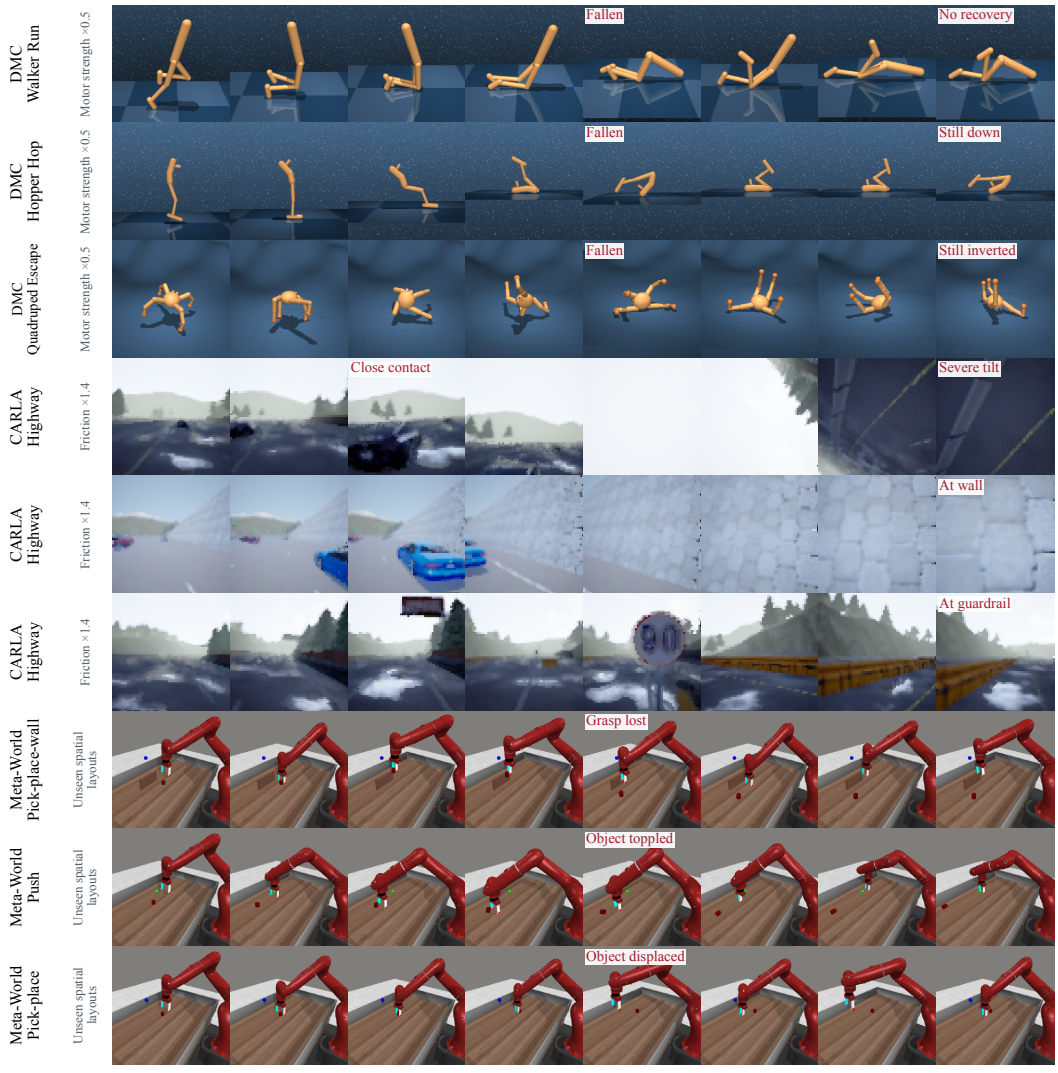

Figure A.1: Visualization of the DreamerV3's failure modes under zero-shot shifts.

B Experiment Details

B.1 Benchmark Configurations and Shift Settings

CARLA. Our autonomous driving evaluation is conducted in a highway scenario constructed within the CARLA simulator, an open-source platform renowned for its high-fidelity, realistic urban environments (Dosovitskiy et al., 2017). The complexity of CARLA makes it a particularly challenging benchmark for vision-based RL. The environment features a high density of dynamic agents, complex visual elements including varied lighting and shadows, and requires the agent to make fine-grained control decisions.

The agent is incentivized by a reward function designed to encourage efficient and safe driving:

$$r_t = v_{ego}^{\top} \hat{u}_{highway} \cdot \Delta t - \lambda_1 \cdot \text{collision} - \lambda_2 \cdot |\text{steer}| \tag{9}$$

where v_{ego} is the velocity vector of the ego-vehicle, projected onto the highway's unit vector $\hat{u}_{highway}$, and multiplied by the time discretization step $\Delta t = 0.05s$ to measure forward progress in meters. The penalty coefficients λ_1 and λ_2 are set to 1.0 and 0.0001 respectively. The first term encourages maximizing forward progress, while the latter two terms penalize collisions and sharp, unnecessary steering operations.

While standard evaluations in CARLA focus on complex visual distractions, we rigorously test the agent's zero-shot generalization capabilities by systematically perturbing the underlying physical dynamics of the ego-vehicle. Specifically, the agent is trained under default vehicle dynamics (e.g., standard friction and aerodynamic drag) but evaluated under severe, unseen physical shifts. As summarized in Table 1, we introduce four challenging shift conditions during zero-shot evaluation: (1) **Drag Shift**, where the aerodynamic drag coefficient is scaled by factors of $\{0.50, 0.75, 2.00, 4.00\}$; (2) **Friction Shift**, where tire friction is scaled by $\{0.60, 0.80, 1.20, 1.40\}$; (3) **Torque Shift**, where engine torque is scaled by $\{0.60, 0.80, 1.20, 1.40\}$; and (4) a **Compound Shift**, which applies matched combinations of these perturbations simultaneously.

DeepMind Control Suite (DMC). We further evaluate our method on five continuous locomotion tasks from the DeepMind Control Suite (DMC): `Quadruped Walk`, `Quadruped Run`, `Walker Run`, `Quadruped Escape`, and `Hopper Hop`. DMC provides a standardized set of continuous control tasks simulated in the MuJoCo physics engine. In the vision-based setting, the agent must infer complex physical dynamics—such as joint articulation, balance, and momentum—strictly from high-dimensional pixel streams. The agent receives standard task-specific dense rewards, which typically incentivize forward velocity and posture maintenance.

To evaluate zero-shot generalization against physical dynamics shifts, we train the agent under default simulator parameters and evaluate it across 12 unseen shifted conditions. As detailed in Table 1, we scale the motor strength (actuation) and contact friction by factors of $\{0.50, 0.75, 1.25, 1.50\}$, and further evaluate on matched compound shifts where multiple parameters are perturbed simultaneously. The top row of Figure B.1 provides visual examples of the agent's observations across these diverse locomotion tasks.

Meta-World. For contact-rich robotic manipulation, we select 11 Hard/Very Hard tasks from the Meta-World benchmark following Masked World Models (Seo et al., 2023). These tasks require precise end-effector control and robust spatial understanding to manipulate objects and navigate obstacles.

Unlike standard setups that evaluate on fixed spatial configurations, we rigorously test generalization against layout shifts. The agent is trained on a limited set of 50 initial object and goal configurations, and evaluated zero-shot on 50 unseen held-out configurations. This protocol forces the agent to truly understand the task geometry rather than merely memorizing absolute pixel arrangements. The middle rows of Figure B.1 provide visual examples of the agent's observations, illustrating the diverse spatial layouts and manipulated objects across different tasks.

B.2 DreamerV3 Baseline

Our framework leverages DreamerV3 (Hafner et al., 2025) as its foundational world model. Designed as a highly scalable model-based RL agent, DreamerV3 aims to achieve the goal of mastering diverse

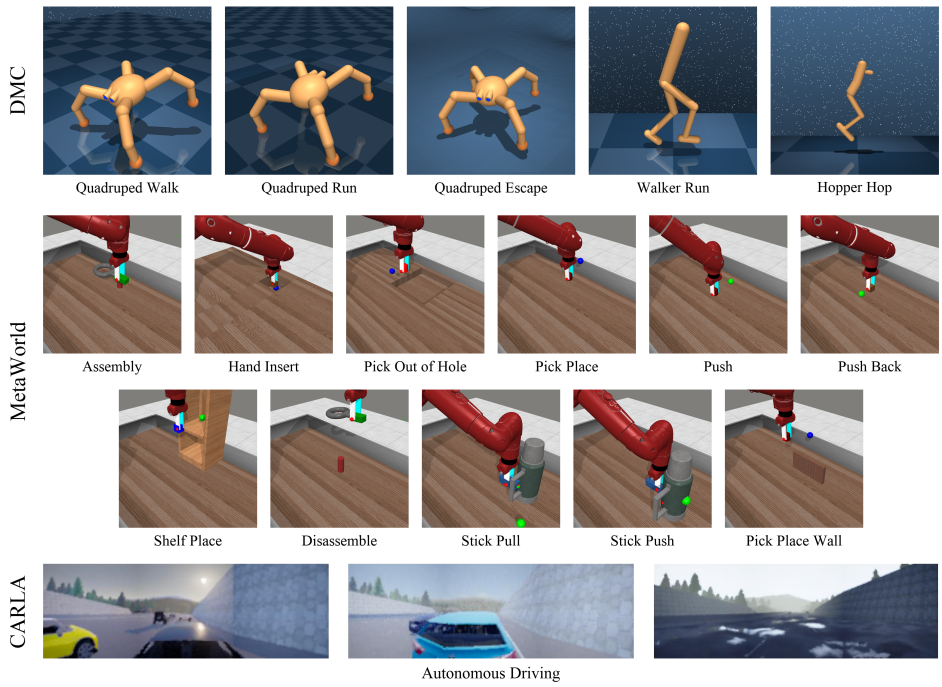

Figure B.1: An overview of the diverse suite of visual benchmarks used in our evaluation. Top: DeepMind Control Suite (DMC), featuring five continuous locomotion tasks. Middle: Meta-World, featuring eleven contact-rich robotic manipulation tasks with diverse spatial layouts. Bottom: CARLA, featuring high-fidelity autonomous driving under varied visual conditions (e.g., dusk, sunny, and rainy weather).

domains without requiring task-specific hyperparameter tuning. To this end, it introduces several critical algorithmic and architectural enhancements over its predecessor, DreamerV2 (Hafner et al., 2020), to ensure broad robustness and stability. The core innovations are summarized below.

Symlog Transformations. To robustly handle the diverse scales of rewards and observations, DreamerV3 applies a `symlog` transformation to the targets of its predictive models instead of using domain-specific clipping or normalization. The transformation is defined as:

$$\text{symlog}(x) = \text{sign}(x)\ln(|x|+1) \tag{10}$$

Automated KL Regularization. While DreamerV2 required manual tuning of the KL divergence penalty, DreamerV3 automates this process by integrating KL balancing with a free-bits objective. It bounds the dynamics and representation losses to ignore KL values below a 1-nat threshold:

$$\mathcal{L}_{\text{dyn}}(\phi) \doteq \max(1, \text{KL}[\text{sg}(q_\phi(z_t|h_t,e_t))|| \quad p_\phi(z_t|h_t)\ ]) \tag{11}$$
$$\mathcal{L}_{\text{rep}}(\phi) \doteq \max(1, \text{KL}[\quad q_\phi(z_t|h_t,e_t)\ ||\text{sg}(p_\phi(z_t|h_t))]) \tag{12}$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operation, which enforces KL balancing by independently constraining the gradients to the prior and the posterior.

Stable Actor-Critic Optimization. DreamerV3 introduces two primary modifications to facilitate stable policy learning across diverse reward structures. First, the actor is optimized using a REINFORCE-style objective with robust return normalization, enabling the use of a fixed entropy regularizer η across tasks. Specifically, the actor loss is defined as:

$$\mathcal{L}_{\text{actor}}(\theta) \doteq -\sum_{t=1}^{T}\left[\text{sg}\left(\frac{R_t^\lambda - v_\psi(s_t)}{\max(1,S)}\right)\log\pi_\theta(a_t \mid s_t) + \eta\mathcal{H}[\pi_\theta(\cdot \mid s_t)]\right], \tag{13}$$

where R_t^λ denotes the λ-return computed along the imagined trajectory, $v_\psi(s_t)$ is the critic prediction used as a baseline, and $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator. The return scale S is estimated from the range between the 5th and 95th percentiles of the λ-returns and smoothed using an exponential moving average:

$$S \doteq \mathrm{EMA}\big(\mathrm{Per}(R_t^\lambda, 95) - \mathrm{Per}(R_t^\lambda, 5), 0.99\big)\,. \tag{14}$$

The denominator $\max(1, S)$ prevents small return scales from amplifying noise while making the relative strength of the entropy regularizer insensitive to the absolute reward magnitude.

Second, the critic predicts a categorical distribution over exponentially spaced value bins rather than directly regressing a scalar return. Let $\tau_t = \mathrm{twohot}(R_t^\lambda)$ denote the two-hot encoding of the target return. The critic is trained by minimizing the corresponding categorical cross-entropy loss:

$$\mathcal{L}_{\mathrm{critic}}(\psi) = -\sum_{t=1}^{T} \tau_t^\top \log p_\psi(\cdot \mid s_t). \tag{15}$$

Architectural Enhancements. To further stabilize the learning process, DreamerV3 adopts Unimix to regularize its categorical distributions and upgrades its core network architectures by employing RMSNorm and SiLU activation functions.

B.3 Computational Resources

We train all models on a server equipped with NVIDIA GeForce RTX 3090 GPUs and a 64-core AMD EPYC 7H12 2.6GHz CPU. We implement our method based on the DreamerV3 architecture using PyTorch (Paszke et al., 2019). Furthermore, we utilize PyTorch 2.8 and leverage the `torch.compile` feature to significantly accelerate the training process (Ansel et al., 2024).

C HYPERPARAMETERS

Table C.1 provides a comprehensive list of the hyperparameter settings used for all experiments in this paper. Our method is built upon DreamerV3, and for a fair comparison, we retain its default hyperparameters for the general training loop, world model, and actor-critic components. We note that these core algorithmic settings are kept fixed across all benchmarks, with only the environment-specific training budgets and action repeats explicitly outlined in the final section.

Table C.1: Hyperparameter settings.

Parameter	Symbol	Value
General & Optimization		
Replay capacity	–	5×10^5
Batch size	B	16
Batch length	T	64
Activation & Normalization	–	SiLU + RMSNorm
Mixed precision	–	float16 autocast
Optimizer	–	LaProp
Learning rate	–	4×10^{-5}
Optimizer betas	(β_1, β_2)	$(0.9, 0.999)$
Optimizer epsilon	ϵ	10^{-20}
Gradient clipping (AGC)	–	0.3
World Model (RSSM)		
Deterministic state size	–	2048
Stochastic variables	–	32
Classes per variable	–	16
Image-loss base scale	β_{obs}	1.0
Reward-loss base scale	β_{rew}	1.0
Dynamics KL base scale	β_{dyn}	1.0
Representation KL base scale	β_{rep}	0.1
Actor-Critic		
Imagination horizon	H	15
Discount horizon	$1/(1-\gamma)$	333
Return λ	λ	0.95
Value EMA decay	–	0.98
Return-norm EMA decay	–	0.99
Actor entropy scale	η	3×10^{-4}
Privileged Semantic Supervision (Ours)		
Descriptor head architecture	–	2 layers, 256 units
Target Symlog EMA decay	–	0.99
Target variance floor	ϵ_{desc}	10^{-4}
Environment Specific Overrides		
Action repeat (All)	–	2
Meta-World budget / envs	–	1.01×10^6 steps / 16 envs
DMC budget / envs	–	1.01×10^6 steps / 4 envs
CARLA budget / envs	–	10^5 steps / 1 env
CARLA prefill / pretrain	–	2500 steps / 100 updates

D LLM PROMPT DESIGN AND SEMANTIC EXTRACTOR EXAMPLES

In this section, we detail how we leverage Large Language Models (LLMs) to automatically structure privileged environmental states into compact semantic features. We provide the prompt template used for generation and showcase examples of the resulting extractor code across different control domains.

D.1 LLM PROMPT DESIGN

Our framework provides a structured task prompt to the LLM during the offline phase. To encourage zero-shot generalization, the prompt explicitly asks the LLM to avoid absolute state variables, such as absolute positions in a global coordinate system, and instead construct robust relative abstractions, including pairwise geometric distances, spatial alignments, and topological indicators.

The complete prompt consists of four core components:

1. **Task Schema:** A high-level description of the task the agent needs to accomplish (e.g., "A robot must pick up a stick and use it to push another object").
2. **Input & Constraints:** Information about the permitted simulator-state accessors, named entities, and camera metadata.
3. **Goal:** A directive to extract compact, task-relevant, and visually inferable properties.
4. **Output Format:** A requirement to output an executable Python function that returns a feature vector y_t^{raw} together with a binary validity mask m_t.

Meta-Prompt Template.

```
You are an expert roboticist and RL researcher designing semantic
world-model state extractors.

[Task]
{Task Schema Description}

[Input]
You have access to the following privileged state accessors:
{Permitted Accessors & Named Entities}

[Goal]
Your goal is to extract compact, task-relevant, and visually inferable
properties.

CRITICAL: Do NOT output absolute coordinates or raw joint angles.
You MUST compute robust relative abstractions such as:
- Pairwise geometric distances between end-effectors and target
↪  objects.
- Spatial alignments (e.g., cosine similarities of directional
↪  vectors).
- Topological indicators.

[Output]
Write an executable Python function `def extract_semantics(state):`
that returns:
1. A target vector `y` (numpy array of floats).
2. A boolean validity mask `mask` (numpy array of bools) to exclude
   uncomputable states (e.g., division by zero or absent entities).

Ensure all dimensions in `y` have clear physical meaning.
```

D.2 Semantic Extractor Examples

To illustrate how simulator states can be mapped to compact, physically interpretable descriptors across different control domains, we provide illustrative semantic extractors for Meta-World, DeepMind Control Suite (DMC), and CARLA.

D.2.1 Example 1: Meta-World Pick Place

In Pick Place, the robotic arm grasps an object and moves it to a target location. The representative extractor below combines geometry and contact information into three grasp-related descriptors: object centering between the finger pads, bilateral contact, and object height conditioned on bilateral contact.

```
def extract_semantics(state):
    """Representative extractor of grasp-related relations."""

    # 1. Read finger and object positions.
    left = np.asarray(state.geom_pos("leftpad_geom"), dtype=float)
    right = np.asarray(state.geom_pos("rightpad_geom"), dtype=float)
    obj = np.asarray(state.obj_pos, dtype=float)
    table_height = float(state.table_height)

    y = np.zeros(3, dtype=np.float32)
    mask = np.zeros(3, dtype=bool)

    # Feature 1: Object centering along the finger-closing axis.
    # 0 means centered; +/-1 marks the pad-center projections.
    if all(p.shape == (3,) and np.isfinite(p).all()
           for p in (left, right, obj)):
        axis = right - left
        width = np.linalg.norm(axis)

        if width > 1e-8:
            midpoint = (left + right) / 2.0
            centering_offset = (
                2.0 * np.dot(obj - midpoint, axis / width) / width
            )
            y[0] = centering_offset
            mask[0] = True

    # Feature 2: Simultaneous contact with both finger pads.
    left_contact = state.touching("leftpad_geom", "objGeom")
    right_contact = state.touching("rightpad_geom", "objGeom")
    bilateral_contact = bool(left_contact and right_contact)

    y[1] = float(bilateral_contact)
    mask[1] = True

    # Feature 3: Object height conditioned on bilateral contact.
    # Zero without contact; otherwise reference-point height
    # above the table, not bottom-surface clearance.
    contact_height = 0.0
    if not bilateral_contact:
        y[2] = contact_height
        mask[2] = True
    elif (obj.shape == (3,) and np.isfinite(obj[2])
          and np.isfinite(table_height)):
        contact_height = obj[2] - table_height
        y[2] = contact_height
        mask[2] = True

    # 2. Ensure all returned targets are finite.
    mask &= np.isfinite(y)
    return np.where(mask, y, 0.0), mask
```

Semantic interpretation. `centering_offset` describes object alignment along the finger-closing axis, while `bilateral_contact` identifies simultaneous contact with both finger pads. Together with `contact_height`, which encodes object height conditioned on bilateral contact, these targets provide structured supervision for grasp-related geometry and contact relations. Bilateral contact is a grasp-related cue rather than a guarantee of stable grasping or task success.

D.2.2 EXAMPLE 2: DMC QUADRUPED RUN

In Quadruped Run, the agent controls a quadruped to run at a target speed. The representative extractor below uses simulator geometry to describe torso posture and instantaneous foot configuration through three continuous descriptors: torso uprightness, relative height, and front–back foot separation.

```
def extract_geometry(state):
    """Representative extractor of torso and foot geometric relations."""

    # 1. Read orientation and positions from the simulator state.
    torso_rotation = np.asarray(
        state.named.data.xmat["torso"], dtype=float
    ).reshape(3, 3)

    torso_pos = np.asarray(
        state.named.data.xpos["torso"], dtype=float
    )

    foot_names = (
        "toe_front_left",
        "toe_front_right",
        "toe_back_left",
        "toe_back_right",
    )
    feet_pos = np.asarray(
        [state.named.data.xpos[name] for name in foot_names],
        dtype=float,
    )

    # Each feature has its own validity mask.
    y = np.zeros(3, dtype=np.float32)
    mask = np.zeros(3, dtype=bool)

    # 2. Compute relative geometric features.

    # Feature 1: Torso uprightness.
    # Cosine between the torso z-axis and the world upward direction.
    uprightness = torso_rotation[2, 2]
    if np.isfinite(uprightness):
        y[0] = np.clip(uprightness, -1.0, 1.0)
        mask[0] = True

    # Feature 2: Torso height relative to the mean foot height.
    # This measures foot-relative height, not ground clearance.
    if (np.isfinite(torso_pos[2])
            and np.isfinite(feet_pos[:, 2]).all()):
        relative_height = torso_pos[2] - np.mean(feet_pos[:, 2])
        y[1] = relative_height
        mask[1] = True

    # Feature 3: Horizontal separation of front and rear foot pairs.
    # This measures instantaneous configuration, not stride length.
    if np.isfinite(feet_pos[:, :2]).all():
        front_midpoint = np.mean(feet_pos[:2, :2], axis=0)
        rear_midpoint = np.mean(feet_pos[2:, :2], axis=0)
        front_back_span = np.linalg.norm(
            front_midpoint - rear_midpoint
        )
        y[2] = front_back_span
        mask[2] = True

    # 3. Ensure all returned targets are finite.
    mask &= np.isfinite(y)
    return np.where(mask, y, 0.0), mask
```

Semantic interpretation. `uprightness` explicitly describes torso orientation relative to the world upward direction, while `front_back_span` summarizes the instantaneous foot configuration through the horizontal separation between the front and rear foot pairs. Together with `relative_height`, which measures torso height relative to the mean foot height, these targets provide structured supervision for the latent representation of locomotion.

D.2.3 EXAMPLE 3: CARLA AUTONOMOUS DRIVING

In autonomous driving, the vehicle must maintain its position and heading relative to the road while responding to changes in motion. The representative extractor below combines privileged vehicle pose, velocity, and local lane geometry into three continuous descriptors: signed heading error, lateral offset, and lateral velocity. These quantities express the vehicle's state relative to the local lane rather than through absolute world coordinates.

```
def extract_semantics(state):
    """Representative extractor using the project CARLA environment."""

    y = np.zeros(3, dtype=np.float32)
    mask = np.zeros(3, dtype=bool)

    # 1. Read vehicle pose and the closest driving-lane waypoint.
    pose = state.vehicle.get_transform()
    waypoint = state.map.get_waypoint(
        pose.location,
        project_to_road=True,
        lane_type=state.carla.LaneType.Driving,
    )
    if waypoint is None:
        return y, mask

    # 2. Construct a horizontal lane frame.
    lane_pose = waypoint.transform
    forward = lane_pose.get_forward_vector()
    tangent = np.array([forward.x, forward.y], dtype=float)
    length = np.linalg.norm(tangent)
    if not np.isfinite(length) or length <= 1e-8:
        return y, mask

    tangent /= length
    normal = np.array([-tangent[1], tangent[0]])

    # Feature 1: Signed heading error, wrapped to [-pi, pi].
    angle = np.deg2rad(
        pose.rotation.yaw - lane_pose.rotation.yaw
    )
    y[0] = np.arctan2(np.sin(angle), np.cos(angle))

    # Feature 2: Signed offset from the local lane tangent.
    offset = np.array([
        pose.location.x - lane_pose.location.x,
        pose.location.y - lane_pose.location.y,
    ])
    y[1] = np.dot(offset, normal)

    # Feature 3: Signed velocity along the same lane normal.
    velocity = state.vehicle.get_velocity()
    y[2] = np.dot([velocity.x, velocity.y], normal)

    # 3. Mask invalid targets and return finite values.
    mask = np.isfinite(y)
    return np.where(mask, y, 0.0), mask
```

Semantic interpretation. `heading_error` measures the signed angular difference between the vehicle heading and the local lane tangent, while `lateral_offset` measures its signed displacement from the tangent line at the reference waypoint. The `lateral_velocity` descriptor projects vehicle velocity onto the same lane normal, capturing motion across the lane. Together, these targets describe both instantaneous lane alignment and lateral motion, providing compact, physically interpretable supervision for the latent representation of driving.

E More Analysis

E.1 Comparison with Raw Privileged-State Prediction

To examine the effect of the auxiliary prediction target, we compare the DreamerV3 baseline with two variants on CARLA: predicting raw privileged states and predicting LLM-generated semantic targets. All methods are evaluated under the same dynamics-shift conditions. As shown in Table E.1, both auxiliary prediction variants improve mean returns over the baseline across all evaluated shifts. LLM-generated semantic targets achieve higher mean returns than raw privileged-state prediction under friction, torque, and compound shifts, while the two variants perform similarly under drag shifts.

These results suggest that the choice of auxiliary prediction target matters beyond the availability of privileged supervision. On the evaluated CARLA shifts, LLM-generated semantic targets provide greater average benefit than directly predicting raw privileged states.

Table E.1: Comparison of auxiliary prediction targets on CARLA at 100k training steps, using DreamerV3 as the baseline. We report episode returns under dynamics shifts and their arithmetic mean (Shift Avg.). The best results are highlighted in **bold**.

Method	Drag	Friction	Torque	Compound	Shift Avg.
Baseline	206.1 ± 66.2	221.4 ± 44.8	200.4 ± 77.0	166.5 ± 72.3	198.6
Baseline + Raw privileged prediction	$\mathbf{299.1} \pm 56.7$	244.6 ± 42.3	260.8 ± 50.1	225.0 ± 65.9	257.4
Baseline + LLM semantic target	298.4 ± 37.0	$\mathbf{340.6} \pm 43.9$	$\mathbf{308.3} \pm 33.7$	$\mathbf{279.3} \pm 44.8$	**306.7**

E.2 Clean-to-Shift Performance Degradation

To assess performance degradation under dynamics shifts, we compare DreamerV3, HarmonyDream, raw privileged-state prediction, and our full method on CARLA. We report the clean return R_{clean}, the average shifted return R_{shift}, and the return drop $\Delta R = R_{\text{clean}} - R_{\text{shift}}$. Here, R_{shift} is the arithmetic mean across drag, friction, torque, and compound shifts. A smaller return drop indicates less performance degradation and should be interpreted alongside the absolute returns.

As shown in Table E.2, our method achieves the highest clean and average shifted returns, together with the smallest return drop among the compared methods. Its smaller drop therefore does not result from a lower clean-performance reference. These results support improved performance and reduced sensitivity to the evaluated dynamics shifts on CARLA.

Table E.2: Clean and shifted episode returns on CARLA at 100k training steps. R_{shift} is averaged across the four dynamics-shift conditions. Higher returns and a lower return drop are preferred; the best results are highlighted in **bold**.

Method	$R_{\text{clean}} \uparrow$	$R_{\text{shift}} \uparrow$	$\Delta R \downarrow$
DreamerV3	356.5	198.6	157.9
HarmonyDream	403.8	265.7	138.1
Baseline + Raw privileged prediction	379.9	257.4	122.5
Ours	**415.5**	**365.4**	**50.1**

E.3 Gradient Analysis of Adaptive Multi-Loss Calibration

To analyze how adaptive calibration modulates the gradient magnitudes of the semantic prediction objective relative to standard world-model objectives, we conduct offline diagnostics using checkpoints from a Meta-World Push training run.

Gradient measurement. For the shared encoder and RSSM parameters, we compute the gradients for four loss components: image reconstruction (Image), reward prediction (Reward), grouped KL

divergence (KL), and the semantic objective (Target). We compare the original unweighted gradient norms $\|\mathbf{g}_j\|_2$ against the calibrated gradient norms $\exp(-s_j)\|\mathbf{g}_j\|_2$.

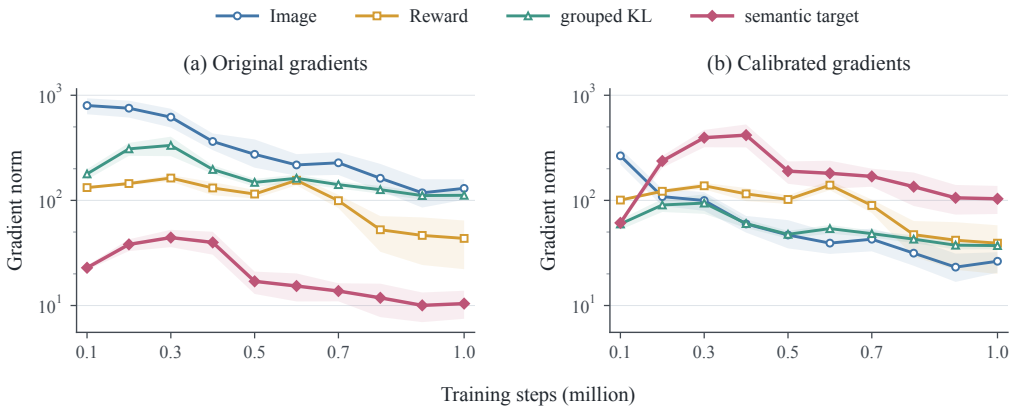

Figure E.1: Gradient magnitudes before and after adaptive calibration. We measure the encoder and RSSM gradients for image reconstruction, reward prediction, grouped KL, and semantic prediction (Target). Panel (a) shows the original unweighted gradient norms, and panel (b) shows the norms after applying the learned calibration coefficients.

Results and interpretation. As shown in Figure E.1, before applying the learned coefficients, the mean gradient norm of the semantic prediction objective (Target) is consistently smaller than those of the other objectives, particularly image reconstruction (Image). Applying the learned calibration coefficients brings the semantic gradients to a comparable scale with the other world-model objectives, allowing auxiliary semantic supervision to provide a comparable gradient signal to the shared encoder and RSSM.

These measurements confirm that the adaptive calibration coefficients effectively strengthen the auxiliary semantic signal, preventing it from being overwhelmed by the generative modeling objectives.