VIGOR: Zero-Shot Visual Generalization via Latent-Space Consistency in Model-Based Reinforcement Learning

Mingyu Park*
Robotics Program
KAIST, ETRI
min9yu@kaist.ac.kr
mngupark@etri.re.kr

Samyeul Noh*
Electrical Engineering
KAIST, ETRI
samuel@kaist.ac.kr
samuel@etri.re.kr

Hyun Myung†
Electrical Engineering
KAIST
hmyung@kaist.ac.kr

Donghwan Lee†
Electrical Engineering
KAIST
donghwan@kaist.ac.kr

Abstract

Model-based reinforcement learning (MBRL) achieves strong sample efficiency by planning within learned latent dynamics, yet its performance degrades substantially under unseen visual distractions such as background variations, lighting changes, or camera shifts. Unlike model-free RL, where encoder perturbations affect only single-step predictions, MBRL suffers from a two-level vulnerability: visual distractions first push encoder outputs out of distribution, and these errors then compound through recursive latent rollouts over the planning horizon. We propose **VI**sual **G**eneralization via latent-space c**O**nsistency in model-based **R**L (**VIGOR**), a framework that enables zero-shot generalization to unseen visual distractions while retaining the sample efficiency of its MBRL backbone. VIGOR enforces latent-space consistency through three interdependent components: (i) *asymmetric weak-to-strong augmentation*, which pairs weak-only and weak-to-strong latent views within a single batch; (ii) *dynamics-level consistency*, which enforces augmentation-invariant transition predictions through direct latent regression; and (iii) *encoder-level stabilization*, which prevents encoder drift under the cross-augmentation supervision imposed by dynamics-level consistency. Evaluations on the DeepMind Control Suite (DMC) and Robosuite show that VIGOR, instantiated on TD-MPC2, outperforms state-of-the-art model-free and model-based baselines in zero-shot generalization, surpassing the second-best baseline by 3.4% on DMC and 43.6% on Robosuite. Moreover, VIGOR achieves these improvements while maintaining sample efficiency comparable to that of its backbone and requiring fewer environment interactions than the other evaluated baselines. Ablations further show that VIGOR's robustness is augmentation-agnostic: replacing the default augmentation with alternatives from distinct perturbation families preserves strong generalization, confirming that latent-space consistency, not the augmentation choice, drives robustness. Finally, a second instantiation of VIGOR on DreamerV3, a structurally distinct recurrent state-space world model, also improves zero-shot generalization over the DreamerV3 baseline on both benchmarks, suggesting that the benefits of VIGOR are not limited to the TD-MPC2 backbone. Code and videos are available on our project page: https://vigor-gen.github.io/.

*Equal contribution.
†Corresponding authors. Correspondence to: hmyung@kaist.ac.kr, donghwan@kaist.ac.kr

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

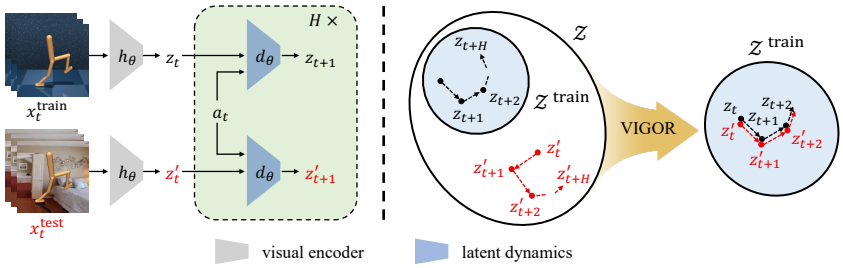


Figure 1: **Challenges of unseen visual distractions.** Left: During training (top), clean observations x_t^{train} are encoded into in-distribution latents $z_t \in \mathcal{Z}^{\text{train}}$, where the dynamics model d_θ produces accurate transitions. At test time (bottom), visually distracted observations x_t^{test} yield out-of-distribution latents $z'_t \notin \mathcal{Z}^{\text{train}}$, causing erroneous transitions. Right: These deviations compound over horizon H, leading to divergent rollouts and planning failures. The proposed VIGOR framework targets this failure mode by aligning perturbed rollouts $(z'_t, z'_{t+1}, \ldots, z'_{t+H})$ with their in-distribution counterparts $(z_t, z_{t+1}, \ldots, z_{t+H})$, anchoring predictions to the task-relevant manifold $\mathcal{Z}^{\text{train}}$ (mechanism described in Section 3).

1 Introduction

Visual reinforcement learning (visual RL) trains control policies directly from pixels, eliminating task-specific state estimation [25, 26, 44, 45]. Despite success in locomotion and manipulation [8, 17, 23], visual RL remains sensitive to *unseen* visual distractions: even minor shifts in backgrounds, lighting, or camera position can cause substantial policy degradation.

In model-free RL (MFRL), visual generalization has been addressed through data augmentation [16, 20], robust representation learning [4, 14, 47], and stabilized value-function training [19, 29]. Model-based RL (MBRL) offers superior sample efficiency by planning in learned latent dynamics [12, 17], but achieving visual generalization in MBRL is *structurally* more challenging. In MFRL, encoder perturbations affect only a single-step prediction. In MBRL, the problem is two-level: visual distractions first push encoder outputs out of the training distribution, and these out-of-distribution (OOD) latents are then recursively propagated by the dynamics model over a planning horizon H, causing errors to compound across steps (Figure 1). This two-level vulnerability, which does not arise in MFRL, means that techniques effective for single-step robustness are insufficient when the encoder simultaneously serves a recursive dynamics model, Q-value networks, and a planner.

We propose **VI**sual **G**eneralization via latent-space c**O**nsistency in model-based **RL** (**VIGOR**), a framework that enables zero-shot generalization while retaining the sample efficiency of its TD-MPC2 backbone. VIGOR addresses the two-level vulnerability through three interdependent components: (i) *asymmetric weak-to-strong augmentation* (AW), a batch-splitting strategy that produces paired weak-only and weak-to-strong latents; (ii) *dynamics-level consistency* (DC), which enforces augmentation-invariant transition predictions by anchoring dynamics outputs to stable targets; and (iii) *encoder-level stabilization* (ES), which prevents the shared encoder from drifting under the cross-augmentation pressure imposed by DC.

We evaluate VIGOR on the DeepMind Control Suite (DMC) [40] and Robosuite [55] under the RL-ViGen zero-shot protocol [48], covering seven visual distraction types—a substantially broader evaluation than the two types (*background-color*, *background-video*) typically considered in prior work [4, 7, 14, 20] (Appendix F.3). VIGOR outperforms state-of-the-art MFRL and MBRL baselines, improving zero-shot generalization over the second-best baseline by 3.4% on DMC and 43.6% on Robosuite while retaining the sample efficiency of its TD-MPC2 backbone. Ablation studies further reveal that naive transfer of MFRL strategies fails in MBRL: applying strong augmentation uniformly across the batch, without the asymmetric splitting of AW, achieves only 48% of VIGOR's performance on DMC (Section 4.3). Furthermore, VIGOR's robustness is augmentation-agnostic: latent-space consistency, rather than any specific augmentation choice, drives generalization (Section 4.4; Appendix G.5).

Our contributions are:

- **Problem identification and analysis.** We identify a two-level failure mode in MBRL: visual distractions first produce OOD encoder outputs, which then compound through recursive rollouts—a failure mode unique to recursive latent dynamics (Figure 1; Section 4.5).

- **Integrated framework for MBRL robustness.** We propose VIGOR, which addresses this failure mode through three interdependent components (AW, DC, and ES) designed for the recursive world-model setting. We show that naive transfer of MFRL strategies fails in MBRL, that VIGOR's integration is necessary, and that the resulting robustness is augmentation-agnostic (Sections 4.3–4.5; Appendix G.5). We further show that VIGOR also improves zero-shot generalization when instantiated on DreamerV3, a structurally distinct recurrent state-space model (RSSM)-based backbone (Appendix I).
- **State-of-the-art zero-shot generalization.** VIGOR outperforms MFRL and MBRL baselines, surpassing the second-best baseline by 3.4% on DMC and 43.6% on Robosuite while preserving the strong sample efficiency of its TD-MPC2 backbone.

2 Related Work

2.1 Visual Model-Based Reinforcement Learning

Latent-space MBRL learns compact world models for planning or imagination in latent space [6, 9, 12, 17, 18]. The Dreamer series adopts a *decoder-based* architecture that reconstructs pixel observations [10–12], whereas TD-MPC uses a *decoder-free* architecture that plans without reconstruction [17, 18]. Both families assume identical training and test distributions; in practice, unseen visual distractions push encoder outputs out of distribution, causing compounding rollout errors (Figure 1). VIGOR equips latent-space MBRL with three interdependent components (AW, DC, and ES) designed to mitigate this failure mode. An extended discussion of MBRL architectures and our backbone selection rationale is provided in Appendix H.

2.2 Visual Generalization in Model-Free Reinforcement Learning

MFRL approaches address visual robustness through data augmentation [16, 20, 37], self-supervised representation learning [4, 14], and pretrained visual encoders [32, 47]. These methods operate at the *single-step* level: the encoder maps observations directly to Q-values or policy outputs without recursive rollouts. In MBRL, by contrast, encoder perturbations propagate through multi-step latent rollouts and compound across the planning horizon, motivating VIGOR's two-level design. A detailed survey is provided in Appendix B.1.

2.3 Visual Generalization in Model-Based Reinforcement Learning

Invariant MBRL methods [39, 41, 53, 54] learn representations that discard task-irrelevant visual factors but typically rely on exposure to distractions during training or on *test-time adaptation*. VIGOR targets the stricter *zero-shot* setting where agents are trained on clean observations and no adaptation data are available. Dr. G [7] is the most directly comparable prior work, as it explicitly targets zero-shot generalization in latent-space MBRL. It extends Dreamer [10] with dual contrastive objectives and an inverse-dynamics auxiliary loss. VIGOR differs in mechanism: rather than contrastive negatives and auxiliary supervision, it enforces dynamics-level consistency (DC) through direct latent regression, supported by asymmetric weak-to-strong augmentation (AW) and encoder-level stabilization (ES). Built on TD-MPC2 [17], VIGOR targets the same zero-shot setting through a fundamentally different inductive bias. A detailed survey is provided in Appendix B.2.

3 VIGOR

VIGOR addresses the two-level vulnerability identified in Section 1: visual distractions first perturb encoder outputs, and these perturbations then compound through recursive latent rollouts. The framework comprises three interdependent components, each targeting a different level of the world-model pipeline: input augmentation, latent dynamics, and encoder representation. Figure 2 provides an overview. The three components depend on each other: AW produces the paired latents on which DC operates; DC stabilizes dynamics but can cause encoder drift; ES counteracts this drift. Because AW modifies only the input pipeline while DC and ES act through auxiliary losses, VIGOR can be integrated into latent-space MBRL frameworks without modifying the underlying planner or optimization procedure. Throughout the main paper, we instantiate VIGOR on TD-MPC2 [17]; a second instantiation on DreamerV3 [12] is described in Appendix I.

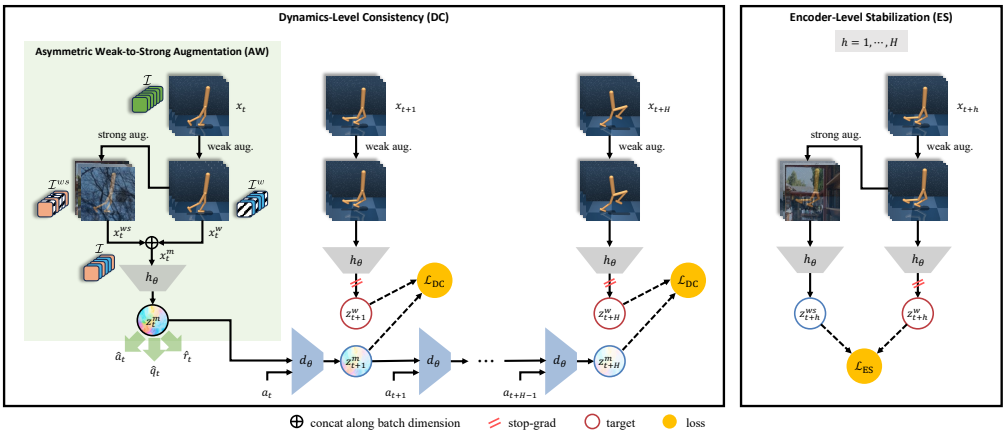


Figure 2: **Overview of VIGOR.** Left: AW partitions a mini-batch into weak-only ($\mathcal{I}^w$) and weak-to-strong ($\mathcal{I}^{ws}$) subsets, producing paired views x_t^w and x_t^{ws} that are concatenated along the batch dimension as $x_t^m = x_t^w \oplus x_t^{ws}$ and encoded by the shared encoder h_θ into the mixed representation z_t^m. DC rolls z_t^m forward through the dynamics model over horizon H and aligns each prediction with the weakly augmented target z_{t+h}^w for $h = 1, \ldots, H$, enforcing augmentation-invariant transition predictions. Right: ES aligns weak-only and weak-to-strong encodings of the same frame, preventing encoder drift under DC's cross-augmentation supervision.

3.1 Asymmetric Weak-to-Strong Augmentation (AW)

We distinguish between *weak* augmentations (minor perturbations that preserve most visual information) and *strong* augmentations that impose substantial visual changes. We refer to the sequential composition $\tau^s \circ \tau^w$ as *weak-to-strong* augmentation, emphasizing application order rather than a progressive curriculum. We adopt *random-shift* [45] as τ^w and *random-overlay* [14] as τ^s.

AW produces paired *weak-only* and *weak-to-strong* samples within a single mini-batch, providing the structural foundation for DC while preserving the original batch size. As illustrated in Figure 2, observations are processed through two augmentation branches: the *weak-only* branch applies τ^w alone, while the *weak-to-strong* branch applies $\tau^s \circ \tau^w$. The two branches are then concatenated along the batch dimension and jointly encoded by the shared encoder h_θ into the mixed representation z_t^m, which is rolled forward by the dynamics model.

Batch split. Given a mini-batch of N samples with index set $\mathcal{I} = \{1, \ldots, N\}$, we partition $\mathcal{I}$ uniformly at random into two disjoint subsets:

$$\mathcal{I} = \mathcal{I}^w \cup \mathcal{I}^{ws}, \quad \mathcal{I}^w \cap \mathcal{I}^{ws} = \varnothing, \quad \zeta = |\mathcal{I}^w|/|\mathcal{I}^{ws}|. \tag{1}$$

Samples in $\mathcal{I}^w$ are routed through the weak-only branch, while samples in $\mathcal{I}^{ws}$ are routed through the weak-to-strong branch. We set $\zeta = 1$ throughout our experiments, splitting the batch evenly.

Augmented representations. At the first time step of each trajectory segment of horizon H, we construct weak-only and weak-to-strong observation stacks for the two subsets:

$$x_t^w = \tau^w\big(\{x_{t,n}\}_{n\in\mathcal{I}^w}, \upsilon^w\big), \quad x_t^{ws} = \tau^s\big(\tau^w\big(\{x_{t,n}\}_{n\in\mathcal{I}^{ws}}, \upsilon^w\big), \upsilon^s\big), \tag{2}$$

where $x_{t,n}$ denotes the observation of the n-th sample at time t, and $\upsilon^w \sim \Upsilon^w$ and $\upsilon^s \sim \Upsilon^s$ parameterize the respective augmentations. The mixed observation $x_t^m = x_t^w \oplus x_t^{ws}$, where $\oplus$ denotes batch-dimension concatenation, is encoded into $z_t^m = h_\theta(x_t^m)$ in a single forward pass. The mixed representation z_t^m initializes the latent rollout, whereas weakly augmented views of the subsequent observations $x_{t+1:t+H}$ provide the DC targets and, paired with their weak-to-strong counterparts, the ES pairs (Sections 3.2–3.3). Further analyses of AW are provided in Appendix G.5.

3.2 Dynamics-Level Consistency (DC)

The key insight is that task-relevant dynamics should remain invariant to visual distractions: the world model should produce consistent next-state predictions regardless of the visual perturbation applied to its input. The DC loss enforces this invariance at the dynamics level, a constraint specific to latent-space MBRL, where the encoder output feeds a recursive transition model.

For reference, the standard dynamics loss in latent-space MBRL [17] is:

$$\mathcal{L}_{\text{dyn}}(\theta) = \left\| d_\theta(z_t, a_t) - \text{sg}\big(h_\theta(x_{t+1})\big) \right\|_2^2, \tag{3}$$

where $z_t = h_\theta(x_t)$ is the encoded observation, d_θ is the dynamics model, and $\text{sg}(\cdot)$ denotes the stop-gradient operator.

DC modifies this objective in two ways: at the first time step of each trajectory segment, the dynamics model receives the mixed representation z_t^m from AW; the target is the encoding of a *weakly augmented* successor. The single-step DC loss is:

$$\mathcal{L}_{\text{DC}}(\theta) = \left\| d_\theta(\hat{z}_t^m, a_t) - \text{sg}(z_{t+1}^w) \right\|_2^2, \quad \text{where } z_{t+1}^w = h_\theta\big(\tau^w(x_{t+1}, \upsilon^w)\big). \tag{4}$$

The mixed representation z_t^m is injected only at the initial step of each trajectory segment of horizon H ($\hat{z}_t^m = z_t^m$), after which the dynamics model rolls forward recursively: $\hat{z}_{t+h}^m = d_\theta(\hat{z}_{t+h-1}^m, a_{t+h-1})$ for $h = 1, \ldots, H$. At every step, the rollout prediction $\hat{z}_{t+h}^m$ is aligned with the weakly augmented target z_{t+h}^w, propagating the initial perturbation through the entire horizon while requiring augmentation-invariant transitions.

The weak target z_{t+h}^w provides a stable supervisory signal. Applying τ^w universally, rather than maintaining a clean path, is a deliberate choice: consistent with prior work on visual RL [44, 45], weak augmentation improves learning and provides a stable yet slightly diversified anchor for the subsequent objectives. The model must produce consistent predictions from the perturbed mixed input, thereby enforcing *augmentation-invariant* latent dynamics.

3.3 Encoder-Level Stabilization (ES)

While DC enforces invariance at the dynamics level, the encoder may still produce inconsistent representations across augmentations. In VIGOR, a single shared encoder feeds the dynamics model, Q-value networks, and model predictive control (MPC) planner. The cross-augmentation supervision from DC can distort this shared representation, causing encoder drift that propagates through multi-step rollouts. ES adapts the alignment objective of SODA [14], originally proposed for MFRL, to a different role in VIGOR: counteracting the encoder drift induced by DC's cross-augmentation supervision, a failure mode specific to MBRL, where the encoder feeds recursive latent rollouts.

At each time step, the ES loss minimizes the ℓ_2 distance between ℓ_2-normalized weak-only and weak-to-strong encodings of the same frame:

$$\mathcal{L}_{\text{ES}}(\theta) = \left\| \frac{z_t^{ws}}{\|z_t^{ws}\|_2} - \frac{\text{sg}(z_t^w)}{\|\text{sg}(z_t^w)\|_2} \right\|_2^2. \tag{5}$$

ES thus anchors weak-to-strong encodings to their weak-only counterparts, keeping the shared representation stable while DC constrains the dynamics. A detailed discussion of the structural role of ES is provided in Appendix D.

3.4 Overall Training Objective

VIGOR modifies the TD-MPC2 [17] learning objective by replacing its latent consistency loss (Eq. (3)) with $\mathcal{L}_{\text{DC}}$ and adding $\mathcal{L}_{\text{ES}}$. For a trajectory segment $\Gamma = (x_i, a_i, r_i, x_{i+1})_{i=t}^{t+H}$ sampled from replay buffer $\mathcal{B}$, the overall objective is:

$$\mathcal{L}_{\text{total}}(\theta) = \mathbb{E}_{\Gamma \sim \mathcal{B}} \Bigg[\underbrace{\mathbb{E}_{\upsilon^w \sim \Upsilon^w, \upsilon^s \sim \Upsilon^s} \Big[\textstyle\sum_{i=t}^{t+H} \lambda^{i-t} \underbrace{\mathcal{L}_{\text{TD-MPC2}}(\theta; \mathcal{L}_{\text{rew}}, \mathcal{L}_Q, \mathcal{L}_{\text{DC}}, \Gamma_i)}_{\text{world-model losses}} + \mathcal{L}_{\text{ES}}(\theta; \upsilon^w, \upsilon^s) \Big]}_{\mathcal{L}_{\text{VIGOR}}(\theta; \upsilon^w, \upsilon^s)} \Bigg], \tag{6}$$

where $\Gamma_i = (x_i, a_i, r_i, x_{i+1})$ denotes the per-step transition tuple within the segment, and $\mathcal{L}_{\text{TD-MPC2}}$ denotes the base world-model objective comprising the reward prediction loss ($\mathcal{L}_{\text{rew}}$) and the Q-value loss ($\mathcal{L}_Q$), with the DC loss ($\mathcal{L}_{\text{DC}}$) in place of the standard latent consistency loss. The temporal coefficient $\lambda \in (0, 1]$ downweights losses at later horizon steps. Here, $\mathcal{L}_{\text{VIGOR}}(\theta; \upsilon^w, \upsilon^s)$ denotes the per-step VIGOR loss. The terms highlighted in red are the only additions to the original TD-MPC2 objective. The training procedure is summarized in Algorithm 1 (Appendix C); implementation details and hyperparameters are given in Appendix E; and a detailed discussion of each component appears in Appendix D.

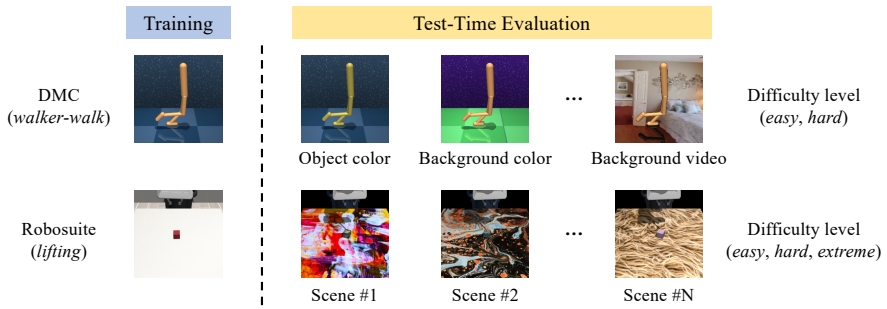


Figure 3: **Illustrative training–evaluation setup.** Agents are trained on clean observations (leftmost column) and evaluated *zero-shot* under unseen visual distractions. Shown are representative examples from DMC (top, *walker-walk*) and Robosuite (bottom, *lifting*) across diverse distraction types, scenes, and difficulty levels.

Table 1: **Zero-shot generalization scores.** Mean and interquartile mean (IQM) of episode returns aggregated across all tasks, distraction types, and difficulty levels. Values in brackets denote 95% stratified bootstrap confidence intervals. Cyan-shaded cells with boldface indicate the best score per row.

		MFRL				MBRL				
Env	Metric	DrQ-v2	SVEA	SRM	SimGRL	TD-MPC2	DreamerV3	HRSSM	Dr. G	**VIGOR (Ours)**
DMC	Mean	626.1 [618.7, 633.6]	726.2 [721.2, 731.2]	752.0 [747.2, 756.7]	710.2 [706.4, 713.8]	599.1 [592.2, 606.1]	635.3 [629.3, 641.3]	663.1 [657.8, 668.3]	623.7 [616.9, 630.4]	**777.6 [773.6, 781.5]**
	IQM	739.5 [727.1, 751.6]	815.5 [811.6, 819.4]	834.9 [831.3, 838.5]	807.9 [805.5, 810.2]	672.2 [661.1, 683.3]	738.3 [728.6, 747.6]	749.1 [742.3, 755.9]	704.8 [694.3, 714.9]	**856.9 [853.2, 860.6]**
Robosuite	Mean	41.0 [39.3, 42.8]	82.8 [78.7, 87.0]	91.8 [87.2, 96.6]	40.3 [38.3, 42.6]	40.2 [39.2, 41.2]	64.3 [62.5, 66.3]	45.1 [44.3, 45.9]	54.0 [52.0, 56.3]	**131.8 [124.9, 138.8]**
	IQM	15.7 [14.2, 17.2]	38.7 [33.9, 43.9]	50.5 [45.5, 55.6]	7.2 [5.5, 9.1]	1.7 [1.4, 2.0]	30.1 [27.2, 33.0]	6.2 [5.7, 6.8]	3.6 [2.4, 5.1]	**58.7 [50.8, 67.2]**

4 Experiments

We evaluate VIGOR's zero-shot generalization and sample efficiency against state-of-the-art MFRL and MBRL methods on six continuous control tasks from DMC [40] and four manipulation tasks from Robosuite [55]. All agents are trained in clean environments and evaluated *zero-shot* under unseen visual distractions (Figure 3; representative evaluation scenes are catalogued in Appendix F.5). Our experiments address the following questions:

- **Q1.** Does VIGOR achieve superior zero-shot generalization while preserving sample efficiency?
- **Q2.** How does AW contribute to the efficiency–robustness trade-off, and why is its batch-splitting design necessary for MBRL?
- **Q3.** How do DC and ES individually contribute, and how do they interact?
- **Q4.** How do specific design choices (augmentation scheduling, augmentation family, and auxiliary objective) affect generalization?
- **Q5.** Does VIGOR preserve latent-space consistency at both the encoder and dynamics levels under unseen visual shifts?

4.1 Experimental Setup

Environments. We benchmark VIGOR on six DMC tasks (*walker-walk*, *finger-spin*, *cheetah-run*, *walker-stand*, *reacher-easy*, and *cartpole-swingup*) and four Robosuite manipulation tasks (*door-opening*, *nut-assembly*, *peg-in-hole*, and *lifting*). Details are provided in Appendix F.

Baselines. We compare VIGOR against eight baselines spanning both paradigms: four MFRL methods (DrQ-v2 [45], SVEA [16], SRM [20], SimGRL [37]) and four MBRL methods (TD-MPC2 [17], DreamerV3 [12], HRSSM [39], Dr. G [7]). Baseline descriptions are provided in Appendix F.2.

Distractions. We follow the RL-ViGen zero-shot protocol [48] and evaluate all methods under seven visual distraction types, substantially broader than the two types (*background-color*, *background-video*) used in most prior work [4, 7, 14, 20]. Details are provided in Appendix F.3.

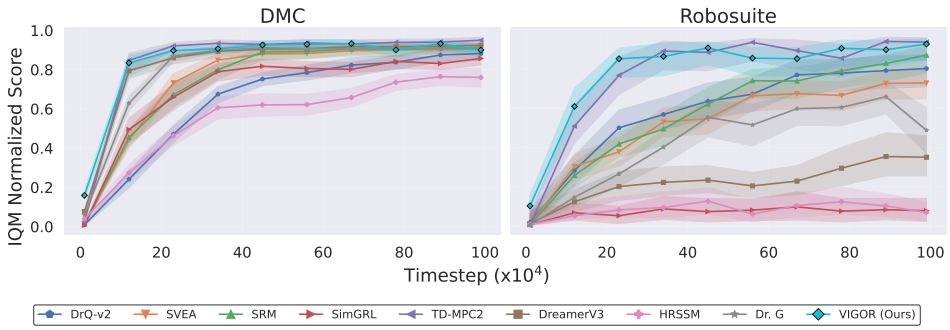


Figure 4: **Sample efficiency.** Learning curves on DMC and Robosuite, reported as IQM normalized scores across methods. Shaded regions denote 95% stratified bootstrap confidence intervals.

4.2 Experimental Results

To address **(Q1)**, we evaluate VIGOR along two axes: *zero-shot generalization* and *sample efficiency*. All results are averaged over ten random seeds.

Zero-shot generalization. All agents are trained in a clean setting and evaluated *zero-shot* under unseen visual distractions across multiple difficulty levels (*easy* and *hard* in DMC; *easy*, *hard*, and *extreme* in Robosuite). Table 1 reports mean and interquartile mean (IQM) [1] aggregated across all tasks, distraction types, and difficulty levels; detailed breakdowns are provided in Appendix G.1.

VIGOR achieves the strongest zero-shot generalization on both benchmarks. On DMC, it leads on both metrics (mean: 777.6; IQM: 856.9), surpassing the second-best baseline (SRM) by 3.4% in mean and 2.6% in IQM. The gap widens substantially on Robosuite, where most baselines degrade under severe distribution shift: VIGOR reaches mean 131.8 and IQM 58.7, exceeding the second-best baseline (SRM) by 43.6% in mean and 16.2% in IQM. Notably, our backbone TD-MPC2 ranks lowest among all baselines on both benchmarks and both metrics, indicating that latent-space MBRL without explicit consistency mechanisms is particularly fragile under visual shifts. VIGOR's strong performance suggests that jointly enforcing encoder-level stabilization and dynamics-level consistency enables structured rollouts under unseen distractions.

Sample efficiency. Figure 4 reports aggregated IQM normalized scores; per-task curves and statistical analyses are provided in Appendix G.2. On DMC, VIGOR closely matches TD-MPC2's rapid convergence while requiring fewer samples than all other baselines. On Robosuite, the efficiency gap widens further: TD-MPC2 and VIGOR converge at comparable rates, while other baselines fall behind in both speed and asymptotic returns. These results confirm that VIGOR preserves the strong sample efficiency of its TD-MPC2 backbone while substantially improving robustness—a combination that prior methods achieve only in part. We corroborate this under a tighter budget on DMC 100K (Appendix G.3) and report wall-clock cost in Appendix G.4.

Beyond the TD-MPC2 backbone. To examine whether these benefits are specific to TD-MPC2, we additionally instantiate VIGOR on DreamerV3, an RSSM-based world model with stochastic latents and actor–critic learning in imagination (Appendix I). Without backbone-specific tuning, DreamerV3+VIGOR raises the mean zero-shot return of DreamerV3 from 635.3 to 767.8 on DMC and from 64.3 to 98.1 on Robosuite, exceeding the mean return of every baseline in Table 1 on both benchmarks, although it remains below the default TD-MPC2 instantiation. VIGOR's gains are thus not limited to the TD-MPC2 backbone.

4.3 Ablation Studies

We conduct ablation studies addressing **(Q2)** and **(Q3)**, isolating the contribution of each component to VIGOR's efficiency–robustness balance. Unless otherwise noted, the ablations below use two tasks per benchmark (*cartpole-swingup* and *reacher-easy*; *door-opening* and *peg-in-hole*).

AW strategy (Q2). We compare AW against three variants: weak-only (`WO`), strong-only (`SO`), and weak-to-strong-only (`WTSO`), which applies both transforms to every sample without batch splitting. We also sweep the mixture ratio $\zeta = |\mathcal{I}^w|/|\mathcal{I}^{ws}|$ to probe the sensitivity of the weak-only

Table 2: **AW ablation: zero-shot generalization.** Mean episode returns ($\pm$ std) averaged across distraction types at each difficulty level. Percentages denote performance relative to VIGOR (AW, ζ=1).

Env	Difficulty	weak-only	strong-only	weak-to-strong-only	VIGOR (ζ=4)	VIGOR (ζ=2)	VIGOR (ζ=0.5)	**VIGOR (Ours; AW, ζ=1)**
DMC	easy	773.9 $\pm$ 246.0	423.8 $\pm$ 421.9	550.6 $\pm$ 377.6	854.4 $\pm$ 176.5	820.8 $\pm$ 192.0	780.9 $\pm$ 235.2	**863.9 $\pm$ 155.4**
	hard	606.6 $\pm$ 326.8	374.9 $\pm$ 401.0	481.3 $\pm$ 358.3	736.3 $\pm$ 274.3	735.8 $\pm$ 270.6	683.0 $\pm$ 302.1	**781.1 $\pm$ 261.2**
	Average	690.2 $\pm$ 301.0 (83%)	399.4 $\pm$ 412.2 (48%)	516.0 $\pm$ 369.6 (62%)	795.3 $\pm$ 238.0 (96%)	778.3 $\pm$ 238.4 (94%)	731.9 $\pm$ 275.1 (88%)	**822.5 $\pm$ 218.9**
Robosuite	easy	134.2 $\pm$ 143.8	187.1 $\pm$ 182.2	297.6 $\pm$ 155.2	276.2 $\pm$ 195.0	313.2 $\pm$ 165.7	329.6 $\pm$ 187.3	**332.0 $\pm$ 186.2**
	hard	112.6 $\pm$ 118.0	137.3 $\pm$ 137.6	138.1 $\pm$ 131.3	151.8 $\pm$ 156.4	144.7 $\pm$ 144.5	172.2 $\pm$ 157.4	**236.7 $\pm$ 186.5**
	extreme	93.4 $\pm$ 96.9	121.3 $\pm$ 122.6	130.1 $\pm$ 120.7	109.0 $\pm$ 110.6	117.1 $\pm$ 111.7	125.1 $\pm$ 123.2	**146.0 $\pm$ 143.0**
	Average	113.4 $\pm$ 122.0 (47%)	148.6 $\pm$ 152.0 (62%)	188.6 $\pm$ 156.6 (79%)	179.0 $\pm$ 172.8 (75%)	191.7 $\pm$ 166.5 (80%)	209.0 $\pm$ 180.5 (87%)	**238.3 $\pm$ 188.9**

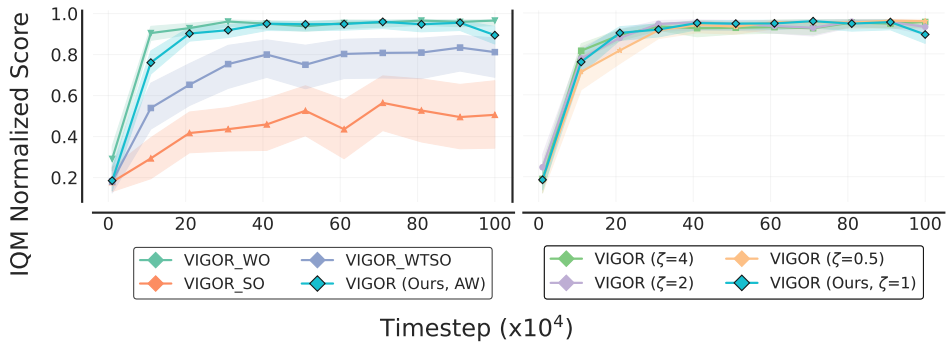


Figure 5: **AW ablation: sample efficiency.** IQM normalized scores as a function of environment steps. `WO`: weak-only; `SO`: strong-only; `WTSO`: weak-to-strong-only; ζ: $|\mathcal{I}^w|/|\mathcal{I}^{ws}|$.

to weak-to-strong balance. Table 2 and Figure 5 report zero-shot generalization and sample efficiency, respectively; further analyses are provided in Appendix G.5.

The results reveal complementary failure modes that illuminate why AW's specific design is necessary for MBRL. On DMC, `SO` degrades to 48% of VIGOR, confirming that applying strong augmentations to an entire batch triggers the two-level vulnerability: perturbed encoder outputs compound through recursive rollouts, destabilizing world-model learning. `WTSO` partially recovers (62%) but remains far below AW, indicating that persistent strong exposure without a stable weak-only anchor disrupts optimization. `WO` converges fast (Figure 5, left), but under-generalizes (83%), as weak shifts alone are insufficient for harder distractions. On Robosuite, where distribution shifts are more severe, the ranking inverts: `WO` becomes the weakest (47%), `SO` improves relatively (62%), and `WTSO` reaches 79% yet still falls short of AW. AW is the only strategy that succeeds on both benchmarks, because its asymmetric splitting simultaneously provides optimization stability and perturbation diversity.

The ζ-sweep reveals a complementary cross-benchmark pattern: on DMC, increasing the weak-only fraction improves performance (ζ=4: 96% > ζ=2: 94% > ζ=0.5: 88%), while on Robosuite the trend reverses (ζ=0.5: 87% > ζ=2: 80% > ζ=4: 75%). Only the balanced mixture (ζ=1) achieves the best performance on both, confirming that this ratio balances stability and generalization coverage.

DC and ES contributions (Q3). Table 3 isolates the effect of each loss. Removing DC reduces generalization to 89% of full VIGOR on DMC and to 46% on Robosuite; removing ES reduces it to 93% and 88%, respectively. Removing both losses jointly reduces performance further (76% on DMC; 33% on Robosuite).

The asymmetry between DC and ES is more pronounced under severe distribution shifts. On DMC, both components contribute modestly and similarly (89% vs. 93%). On Robosuite, the gap widens sharply: DC alone (w/o ES) retains 88%, whereas ES alone (w/o DC) retains only 46%. DC constrains the recursive prediction pathway: without it, the dynamics model produces inconsistent rollouts under visual shifts. ES alone cannot compensate, because encoder-level alignment does not prevent the dynamics model from amplifying residual discrepancies across multi-step rollouts. Conversely, the 12% gap between DC alone and full VIGOR on Robosuite shows that ES's encoder-level alignment is also needed for DC's dynamics-level constraint to function effectively. The two losses are thus complementary: DC anchors transition predictions, while ES anchors the representations on which DC operates. We further visualize these failure modes in Appendix G.6.

Table 3: **Ablation on DC and ES.** Mean episode returns ($\pm$ std) averaged across distraction types at each difficulty level. Percentages denote performance relative to full VIGOR.

Env	Difficulty	w/o DC & ES	w/o DC	w/o ES	**VIGOR (Ours)**
DMC	easy	730.7 ± 296.0	816.4 ± 232.2	826.7 ± 205.0	$\mathbf{863.9 \pm 155.4}$
	hard	522.3 ± 357.2	652.1 ± 328.1	713.8 ± 300.5	$\mathbf{781.1 \pm 261.3}$
	Average	626.5 ± 344.2 (76%)	734.3 ± 295.8 (89%)	770.2 ± 263.3 (93%)	$\mathbf{822.5 \pm 218.9}$
Robosuite	easy	74.2 ± 78.6	128.9 ± 143.3	311.8 ± 184.9	$\mathbf{332.0 \pm 186.4}$
	hard	83.2 ± 85.8	108.5 ± 113.5	193.5 ± 178.6	$\mathbf{236.7 \pm 186.7}$
	extreme	80.2 ± 82.5	95.9 ± 99.9	127.6 ± 120.0	$\mathbf{146.0 \pm 143.2}$
	Average	79.2 ± 82.3 (33%)	111.1 ± 120.8 (46%)	211.0 ± 180.4 (88%)	$\mathbf{238.3 \pm 189.0}$

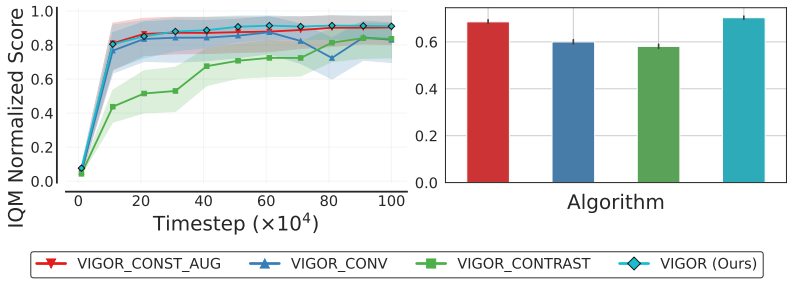


Figure 6: **Ablation on design choices.** Left: Sample efficiency. Right: Zero-shot generalization.

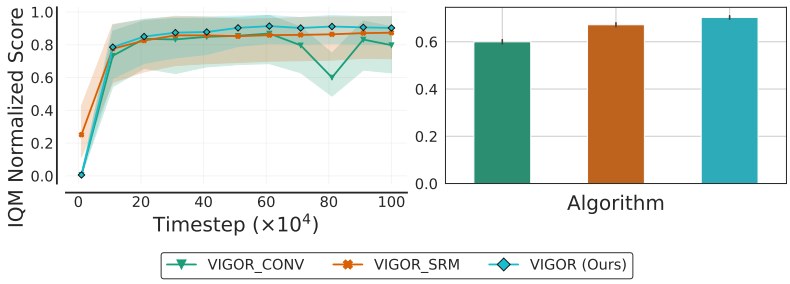


Figure 7: **Augmentation-agnostic robustness.** Three VIGOR variants trained with three different strong augmentations (*random-conv*, *SRM*, and *random-overlay*) all maintain strong zero-shot generalization.

Summary. VIGOR's gains stem from the interplay of all three components. AW provides the efficiency–robustness balance through asymmetric batch splitting; DC and ES are jointly necessary for reliable generalization, with DC playing the dominant role under severe distribution shifts by directly constraining the recursive dynamics pathway.

4.4 Ablation on Key Design Choices

We analyze **(Q4)** by ablating three design choices, each isolated through a single-variable variant: `VIGOR_CONST_AUG` (fixed augmentation parameters across the horizon), `VIGOR_CONV` (*random-conv* replacing *random-overlay*), and `VIGOR_CONTRAST` (contrastive ES objective replacing the alignment-based one). Figure 6 summarizes results on *finger-spin*, *cheetah-run*, and *walker-walk*. VIGOR outperforms all variants on both axes, indicating that (i) dynamic augmentation resampling is important for stable rollouts; (ii) *random-overlay* provides stronger robustness than convolution-based perturbations; and (iii) alignment-based ES complements AW and DC more effectively than contrastive ES. Formal definitions and detailed results are provided in Appendix G.7.

Augmentation-agnostic robustness. A central question is whether VIGOR's robustness depends on the specific augmentation choice or on latent-space consistency. We replace *random-overlay* with *random-conv* (textural perturbations) and *SRM* (spectral masking), two families that produce qualitatively distinct perturbations (Appendix G.5, Figure 19). As shown in Figure 7, all three variants maintain strong zero-shot on the same three tasks, indicating that VIGOR's robustness is driven by latent-space consistency rather than by the choice of augmentation.

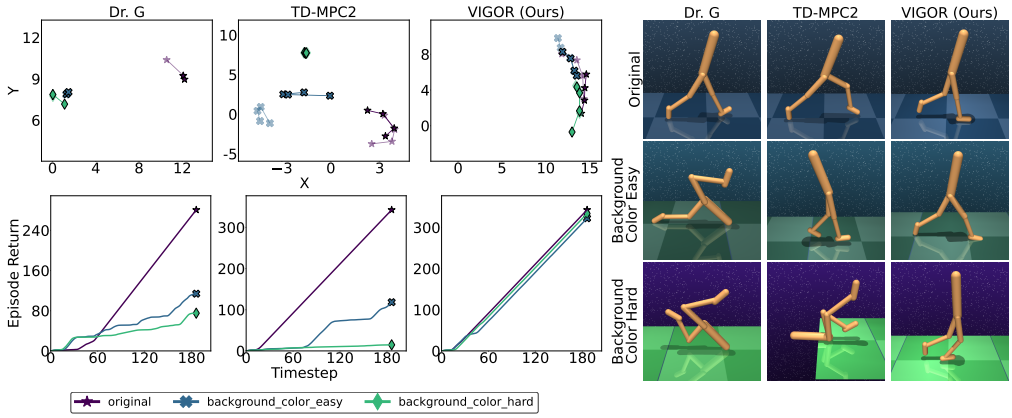


Figure 8: **Latent-space consistency under visual perturbations.** Left: UMAP projections of latent rollouts (top) and episode returns (bottom) on *walker-walk* with *background-color* distractions. Right: Environment snapshots across evaluation settings. In the UMAP plots, markers denote difficulty levels (★: *original*, ×: *easy*, ◊: *hard*); darker trajectories show rollouts starting from z_t ($z_{t:t+H}$), and lighter trajectories show rollouts starting from the preceding step z_{t-1} ($z_{t-1:t+H-1}$), providing temporal context. VIGOR's latent rollouts across difficulty levels remain compact and overlap closely, while TD-MPC2 [17] and Dr. G [7] produce scattered initial encodings or divergent rollouts; the corresponding return curves are shown below each panel.

4.5 Latent-Space Consistency Analysis

To address **(Q5)**, we compare VIGOR against TD-MPC2 (its backbone) and Dr. G (the most directly comparable prior method) by rolling out each agent's world model from identical initial inputs and projecting the predicted latent states into 2D via UMAP [31] (Figure 8).

VIGOR maintains a compact latent manifold across difficulty levels, consistently aligning perturbed trajectories with their clean counterparts. This consistency holds at both levels of the vulnerability identified in Figure 1: the initial encoder outputs under visual distractions remain close to their clean counterparts (encoder level), and this alignment is preserved throughout the rollout horizon (dynamics level). The resulting structural stability translates into stable returns. TD-MPC2 and Dr. G exhibit scattered or divergent rollouts, leading to degraded performance, despite Dr. G's contrastive objectives being designed for generalization. This comparison suggests that direct dynamics-level regression (DC) based on asymmetric batch-splitting (AW), combined with encoder stabilization (ES), provides a more effective inductive bias for MBRL robustness than contrastive objectives with inverse-dynamics supervision, because it jointly constrains both levels of the prediction pathway that drives planning. Additional visualizations are provided in Appendix G.8; we further verify that this consistency holds under deeper planning horizons up to $H{=}10$ (Appendix G.9).

5 Conclusion

We introduced **VIGOR**, a framework that achieves robust zero-shot visual generalization in latent-space MBRL while preserving the strong sample efficiency of its TD-MPC2 backbone. VIGOR addresses the two-level vulnerability of latent-space MBRL: visual distractions perturb encoder outputs, and these perturbations compound through recursive rollouts. The framework integrates three interdependent components (AW, DC, and ES) to mitigate this failure mode. Across challenging OOD benchmarks, VIGOR surpasses the second-best baseline by 3.4% on DMC and 43.6% on Robosuite. Ablation studies confirm that all three components are jointly necessary and that VIGOR's robustness is augmentation-agnostic: latent-space consistency, not any particular augmentation choice, is what drives generalization. A second instantiation on DreamerV3 (Appendix I) also improves zero-shot generalization over its backbone on both benchmarks, suggesting that these benefits are not limited to the TD-MPC2 backbone.

Limitations and future work. VIGOR targets appearance-based perturbations and does not address geometric distortions; evaluations under unseen camera rotations (Appendix G.10) reveal that such shifts remain challenging, motivating geometry-aware representations as a primary future direction. Beyond visual shifts, extending VIGOR to dynamics- and task-level shifts [5, 36] is a natural next step, and validating it in real-robot deployments would assess its practical applicability.

Acknowledgments

This work was supported in part by the Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government [26ZR1100, Research on Intelligent Industrial Convergence]; the internal fund/grant of ETRI [25YR1500, A Study on Generalizable Action Intelligence and Its Application to Physical Robots]; and the National Research Foundation of Korea (NRF) grants funded by the Korean government (MSIT) (Nos. RS-2024-00348461 and RS-2026-25510069).

References

[1] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 34:29304–29320, 2021.

[2] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in Atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

[3] Sumeet Batra and Gaurav S Sukhatme. Zero shot generalization of vision-based rl without data augmentation. In *International Conference on Machine Learning*, pages 3251–3273. PMLR, 2025.

[4] David Bertoin, Adil Zouitine, Mehdi Zouitine, and Emmanuel Rachelson. Look where you look! saliency-guided q-networks for generalization in visual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:30693–30706, 2022.

[5] Michael Beukman, Devon Jarvis, Richard Klein, Steven James, and Benjamin Rosman. Dynamics generalisation in reinforcement learning via adaptive context-aware policies. *Advances in Neural Information Processing Systems*, 36:40167–40203, 2023.

[6] David Ha and Jürgen Schmidhuber. World models. *Advances in Neural Information Processing Systems*, 31, 2018.

[7] Jeongsoo Ha, Kyungsoo Kim, and Yusung Kim. Dream to generalize: Zero-shot model-based reinforcement learning for unseen visual distractions. *arXiv preprint arXiv:2506.05419*, 2025.

[8] 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*, pages 1861–1870. PMLR, 2018.

[9] 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*, pages 2555–2565. PMLR, 2019.

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

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

[12] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *Nature*, 640, 2025.

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

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

[15] Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A. Efros, Lerrel Pinto, and Xiaolong Wang. Self-supervised policy adaptation during deployment. In *International Conference on Learning Representations*, 2021.

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

[17] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

[18] Nicklas A Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*, pages 8387–8406. PMLR, 2022.

[19] Sili Huang, Yanchao Sun, Jifeng Hu, Siyuan Guo, Hechang Chen, Yi Chang, Lichao Sun, and Bo Yang. Learning generalizable agents via saliency-guided features decorrelation. *Advances in Neural Information Processing Systems*, 36, 2024.

[20] Yangru Huang, Peixi Peng, Yifan Zhao, Guangyao Chen, and Yonghong Tian. Spectrum random masking for generalization in image-based reinforcement learning. *Advances in Neural Information Processing Systems*, 35:20393–20406, 2022.

[21] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.

[22] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for Atari. *arXiv preprint arXiv:1903.00374*, 2019.

[23] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[24] Kyungmin Kim, JB Lanier, and Roy Fox. Make the pertinent salient: Task-relevant reconstruction for visual control with distractions. In *Reinforcement Learning Conference*, 2025.

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

[26] Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *Advances in Neural Information Processing Systems*, 33:19884–19895, 2020.

[27] Jeong Woon Lee and Hyoseok Hwang. Diffusion guided adaptive augmentation for generalization in visual reinforcement learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 880–889, 2025.

[28] Kimin Lee, Kibok Lee, Jinwoo Shin, and Honglak Lee. Network Randomization: A simple technique for generalization in deep reinforcement learning. In *Conference on Learning Representations*, 2020.

[29] Siao Liu, Zhaoyu Chen, Yang Liu, Yuzheng Wang, Dingkang Yang, Zhile Zhao, Ziqing Zhou, Xie Yi, Wei Li, Wenqiang Zhang, et al. Improving generalization in visual reinforcement learning via conflict-aware gradient agreement augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23436–23446, 2023.

[30] Vincent Mai, Kaustubh Mani, and Liam Paull. Sample efficient deep reinforcement learning via uncertainty estimation. In *International Conference on Learning Representations*, 2022.

[31] Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[32] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, pages 892–909. PMLR, 2023.

[33] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[34] Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

[35] Max Schwarzer, Johan Samir Obando Ceron, Aaron Courville, Marc G Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, better, faster: Human-level atari with human-level efficiency. In *International Conference on Machine Learning*, pages 30365–30380. PMLR, 2023.

[36] Younggyo Seo, Kimin Lee, Ignasi Clavera Gilaberte, Thanard Kurutach, Jinwoo Shin, and Pieter Abbeel. Trajectory-wise multiple choice learning for dynamics generalization in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:12968–12979, 2020.

[37] Wonil Song, Hyesong Choi, Kwanghoon Sohn, and Dongbo Min. A simple framework for generalization in visual RL under dynamic scene perturbations. *Advances in Neural Information Processing Systems*, 37:121790–121826, 2024.

[38] 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.

[39] Ruixiang Sun, Hongyu Zang, Xin Li, and Riashat Islam. Learning latent dynamic robust representations for world models. In *Forty-first International Conference on Machine Learning*, 2024.

[40] 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.

[41] Tongzhou Wang, Simon 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*, pages 22591–22612. PMLR, 2022.

[42] Rui Yang, Jie Wang, Qijie Peng, Ruibo Guo, Guoping Wu, and Bin Li. Learning robust representations with long-term information for generalization in visual reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[43] Sizhe Yang, Yanjie Ze, and Huazhe Xu. MoVie: Visual model-based policy adaptation for view generalization. *Advances in Neural Information Processing Systems*, 36:21507–21523, 2023.

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

[45] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022.

[46] Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering atari games with limited data. *Advances in neural information processing systems*, 34:25476–25488, 2021.

[47] Zhecheng Yuan, Zhengrong Xue, Bo Yuan, Xueqian Wang, Yi Wu, Yang Gao, and Huazhe Xu. Pre-trained image encoder for generalizable visual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:13022–13037, 2022.

[48] Zhecheng Yuan, Sizhe Yang, Pu Hua, Can Chang, Kaizhe Hu, and Huazhe Xu. RL-ViGen: A reinforcement learning benchmark for visual generalization. *Advances in Neural Information Processing Systems*, 36:6720–6747, 2023.

[49] Zhecheng Yuan, Tianming Wei, Shuiqi Cheng, Gu Zhang, Yuanpei Chen, and Huazhe Xu. Learning to manipulate anywhere: A visual generalizable framework for reinforcement learning. *arXiv preprint arXiv:2407.15815*, 2024.

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

[51] Di Zhang, Bowen Lv, Hai Zhang, Feifan Yang, Junqiao Zhao, Hang Yu, Chang Huang, Hongtu Zhou, Chen Ye, and Changjun Jiang. Focus on what matters: Separated models for visual-based RL generalization. *Advances in Neural Information Processing Systems*, 37:116960–116986, 2024.

[52] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2017.

[53] Xinning Zhou, Chengyang Ying, Yao Feng, Hang Su, and Jun Zhu. Self-consistent model-based adaptation for visual reinforcement learning. *arXiv preprint arXiv:2502.09923*, 2025.

[54] 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.

[55] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.

Appendix Contents

A Impact Statement **16**

B Extended Related Work **16**
B.1 Visual Generalization in MFRL 16
B.2 Extended Discussion of MBRL Visual Generalization 17

C Training Procedure **18**

D Discussion of Method Components **18**

E Implementation Details **20**
E.1 Code Base 20
E.2 Computational Resources 20
E.3 Hyperparameters 20

F Evaluation Details **21**
F.1 Environments and Tasks 21
F.2 Baseline Descriptions 21
F.3 Visual Distraction Categories 22
F.4 Evaluation Protocol 23
F.5 Evaluation Scenes 23

G Supplementary Results **25**
G.1 Zero-Shot Generalization 25
G.2 Sample Efficiency 26
G.3 DMC 100K 28
G.4 Wall-Clock Training Efficiency 29
G.5 Augmentation Ablation Studies 30
G.6 Latent-Space Consistency Analysis: DC and ES Ablation 33
G.7 Ablation Study on Design Choices 35
G.8 Latent-Space Consistency Analysis 37
G.9 Performance under Longer Planning Horizons 39
G.10 Generalization to Unseen Camera Viewpoints 39

H MBRL Baseline Analysis **41**

I VIGOR on DreamerV3 **43**
I.1 Overview 43
I.2 Notation 44
I.3 Where the Two-Level Vulnerability Appears in DreamerV3 45
I.4 Background: KL Balancing in DreamerV3 45
I.5 Integration 45
I.6 Experimental Setup 46
I.7 Results and Discussion 47
I.8 Summary 48

A Impact Statement

This work presents VIGOR, an MBRL framework that achieves zero-shot visual generalization to unseen distractions while preserving sample efficiency. By integrating AW, DC, and ES, VIGOR addresses a critical bottleneck in deploying visual RL agents to environments with visual distribution shifts.

The primary impact is methodological: VIGOR provides a robust approach to stabilize both encoder representations and latent dynamics under distribution shifts. These advances contribute to the broader goal of developing RL systems that are less reliant on task-specific data collection and more resilient to the visual variability in real-world environments.

As our evaluations are conducted within simulated benchmarks (DMC and Robosuite), this work does not pose immediate direct societal or ethical risks. However, as these techniques move toward real-world deployment in robotics and autonomous systems, we emphasize that reliability and safety become paramount. Future applications in safety-critical domains should undergo rigorous validation and incorporate safeguards to prevent catastrophic decision errors arising from misaligned latent representations or unexpected environmental shifts.

B Extended Related Work

B.1 Visual Generalization in MFRL

This subsection provides a detailed survey of MFRL visual generalization methods, complementing the summary in Section 2. Baseline selection criteria are detailed in Appendix F.2; methods not selected as baselines are discussed below for context.

Data-augmentation approaches. Data augmentation has emerged as a simple yet effective strategy for visual robustness in MFRL. RAD [26] systematically evaluates a suite of image transformations for RL, demonstrating that augmentations such as random crop and color jitter improve generalization. DrQ [44] and DrQ-v2 [45] integrate random-shift augmentation into off-policy actor–critic training, achieving strong sample efficiency on pixel-based control. SVEA [16] addresses the instability of augmented Q-learning by maintaining a weighted mixture of augmented and clean Q-targets, stabilizing training under diverse perturbations. SRM [20] applies spectral perturbations in the frequency domain to improve robustness to spatial corruptions. Network Randomization [28] passes observations through a randomly initialized convolutional layer, providing an implicit form of style augmentation; this principle inspires our *random-conv* ablation in Section 4.4. SimGRL [37] tackles imbalanced saliency via feature-level frame stacking and mitigates observational overfitting via a shifted random-overlay augmentation, which is applied uniformly to every sample without batch partitioning. DG2A [27] leverages a pretrained diffusion model (Stable Diffusion [33]) for adaptive data augmentation, generating augmented images guided by saliency maps.

Representation-learning approaches. A complementary direction aims to learn visual representations that are inherently robust to task-irrelevant variations. CURL [25] applies contrastive learning to augmented observations, encouraging the encoder to capture task-relevant features. SODA [14] aligns the encoding of an augmented observation with that of its non-augmented counterpart through a soft alignment loss, promoting augmentation-invariant features without contrastive negatives. VIGOR's ES component adapts SODA's alignment objective to a structurally different role: in MFRL, alignment promotes augmentation-invariant features at the encoder, whereas in VIGOR it stabilizes the encoder against the cross-augmentation supervision induced by DC's dynamics-level training (Section 3.3). SGQN [4] combines saliency-guided attention masks with self-supervised objectives, encouraging the encoder to focus on action-relevant regions. SMG [51] exploits an auxiliary image reconstruction task via separated model branches for encoders and decoders, forcing the encoder to extract task-relevant representations from foreground pixels. ALDA [3] disentangles the latent representation and projects OOD observations back to the training distribution using associative memory. ROUSER [42] learns robust action-value representations based on the information bottleneck framework, leveraging one-step rewards and compressed reward representations.

Pretrained-encoder approaches. Recent work leverages visual representations pretrained on large-scale datasets. PIE-G [47] uses frozen pretrained encoders to provide generalization-friendly features, bypassing the need for augmentation-based robustness. R3M [32] pretrains visual representations on egocentric video data using time-contrastive learning for downstream robotic manipulation. MoVie [43] adapts pretrained vision models for RL via multi-view representation learning. These methods are orthogonal to VIGOR and could potentially be combined with our framework to further improve robustness.

Gradient- and optimization-based approaches. CG2A [29] identifies conflicting gradients between augmented views and aggregates them via conflict-aware gradient surgery, improving generalization stability. PAD [15] uses a self-supervised auxiliary task at test time to adapt the encoder to new visual conditions, an approach that requires online adaptation and thus falls outside the zero-shot setting targeted by VIGOR.

Relationship to VIGOR. All MFRL methods above address robustness at the single-step level: the encoder feeds into a Q-value or policy without recursive dynamics. As identified in Section 1, MBRL exhibits a two-level vulnerability: visual distractions first perturb encoder outputs, and these perturbations then compound through recursive latent rollouts. This structural difference means that augmentation or regularization applied to the encoder alone is insufficient: the dynamics model must also produce consistent predictions under distribution shifts. VIGOR addresses both levels: ES stabilizes the encoder (level one), while DC constrains the recursive dynamics pathway (level two), with AW providing the paired latent structure that enables both constraints. The closest MFRL counterpart, SimGRL, uses random-overlay augmentation but applies it uniformly to every sample without batch splitting, leaving both the dynamics-level vulnerability and cross-augmentation supervision unaddressed.

B.2 Extended Discussion of MBRL Visual Generalization

Invariant MBRL. Several methods aim to learn distraction-invariant representations within the MBRL framework. DBC [50] uses bisimulation metrics to learn representations that group visually different but behaviorally equivalent states. Denoised MDP [41] identifies and removes noise variables from the latent representation. RePo [54] regularizes the encoder via a reconstruction-based objective that selectively preserves task-relevant information. HRSSM [39] learns a bisimulation-based hybrid RSSM in DreamerV3 [12] to extract endogenous, task-relevant latent representations without pixel-level reconstruction. SD [24] uses segmentation masks as a structural prior for robust representation learning, building upon the DreamerV3 architecture. While these methods improve robustness, several require test-time adaptation or additional environment interactions in the target domain, limiting their applicability in the zero-shot setting targeted by VIGOR.

Dr. G and its relationship to VIGOR. Dr. G [7] is the most directly comparable prior work, as it explicitly targets zero-shot visual generalization in MBRL. Built on Dreamer [10], Dr. G applies weak and strong augmentations to create two independent views of every sample and trains dual contrastive losses (reality–reality at the encoder level and dream–reality at the dynamics level) alongside a recurrent inverse-dynamics objective.

VIGOR differs from Dr. G in three key aspects: (i) *Efficiency*: AW partitions a single batch into weak-only and weak-to-strong subsets via asymmetric splitting, where the weak-only subset directly serves as the stable anchor for both DC's transition target and ES's alignment target. This anchor-and-augment structure requires only a single encoder pass per sample. Dr. G, by contrast, generates two fully independent views per sample to construct contrastive pairs, doubling the encoder computation. (ii) *Mechanism*: Both methods address visual robustness in the latent-space MBRL pipeline, but through different means. Dr. G uses contrastive losses requiring negative samples and an inverse-dynamics auxiliary loss. VIGOR enforces consistency through direct latent regression (DC) based on asymmetric batch splitting (AW), combined with encoder stabilization (ES), without contrastive negatives or auxiliary supervision. (iii) *Backbone*: VIGOR builds on TD-MPC2's decoder-free, MPC-based architecture. Dr. G builds on Dreamer's recurrent world model (RSSM) but deliberately omits the decoder, relying instead on contrastive and inverse-dynamics objectives to learn latent representations.

Our experiments show that VIGOR achieves stronger zero-shot generalization with higher sample efficiency (Section 4.2), suggesting that direct dynamics-level regression combined with encoder stabilization provides a more effective inductive bias than contrastive learning for this setting. This advantage in zero-shot generalization persists when VIGOR is instantiated on an RSSM-based backbone (DreamerV3+VIGOR; Appendix I, Table 9), indicating that it does not stem from the TD-MPC2 backbone alone.

C Training Procedure

This section details the training procedure of VIGOR. Since VIGOR leaves the planning mechanism unchanged, we focus on world model learning and omit planner updates in Algorithm 1.

For a trajectory segment $\Gamma = (x_i, a_i, r_i, x_{i+1})_{i=t}^{t+H}$ sampled from replay buffer $\mathcal{B}$, the reward and Q-value losses are defined as:

$$\mathcal{L}_{\text{rew}}(\theta) = \text{CE}\big(\hat{r}_t, r_t\big), \qquad \mathcal{L}_Q(\theta) = \text{CE}\big(\hat{q}_t, q_t\big), \tag{7}$$

where $\hat{r}_t = R_\theta(z_t, a_t)$ and $\hat{q}_t = Q_\theta(z_t, a_t)$ are the predicted reward and Q-value, respectively. The TD target is $q_t = r_t + \gamma \bar{Q}_\theta\big(z_{t+1}, \pi_\theta(z_{t+1})\big)$, where γ is the discount factor, $\bar{Q}_\theta$ is an exponential moving average of the Q-value, $z_t = h_\theta(x_t)$ is the latent representation, and π_θ is a policy trained via entropy maximization. CE denotes the cross-entropy loss. For details on the world model architecture and planning procedure, we refer readers to Hansen et al. [17].

Algorithm 1 World model learning in VIGOR

Input: Replay buffer $\mathcal{B}$; Horizon H; Weak and strong augmentation functions τ^w, τ^s; Temporal coefficient λ; Learning rate η; Momentum coefficient κ
while not converged **do**
  for gradient step $t_g = 1, 2, \ldots, T_g$ **do**
    $\Gamma = (x_i, a_i, r_i, x_{i+1})_{i=t}^{t+H} \sim \mathcal{B}$ {Sample a trajectory segment from the replay buffer}
    $L \leftarrow 0$ {Initialize cumulative loss}
    for $i = t, t+1, \ldots, t+H$ **do**
      $\upsilon^w \sim \Upsilon^w, \upsilon^s \sim \Upsilon^s$ {Sample augmentation parameters}
      $L \leftarrow L + \lambda^{i-t} \mathcal{L}_{\text{VIGOR}}(\theta; \upsilon^w, \upsilon^s)$ {Calculate the per-step VIGOR loss (Eq. (6))}
    end for
    $\theta \leftarrow \theta - \eta \frac{1}{H+1} \nabla_\theta L$ {Update online network parameters}
    $\theta^- \leftarrow \kappa\theta^- + (1-\kappa)\theta$ {Update target network parameters}
  end for
end while

As shown in Algorithm 1, integrating VIGOR into a latent-space MBRL framework is straightforward: only the steps highlighted in yellow need to be added. In the main paper, we instantiate and validate VIGOR within the TD-MPC2 framework, without modifying its planner or optimization procedures. Appendix I presents a second instantiation on DreamerV3, an RSSM-based world model, together with the corresponding training procedure (Algorithm 2).

D Discussion of Method Components

This section discusses why VIGOR's three components (AW, DC, and ES) require careful adaptation for the MBRL setting, how they are implemented, and how they interact. While related ideas have appeared in MFRL, directly applying them to latent-space MBRL is non-trivial due to the two-level vulnerability identified in Section 1: visual distractions first perturb encoder outputs, and these perturbations then compound through recursive world-model rollouts. We highlight the resulting design challenges and clarify how each component addresses them.

AW. The asymmetric batch-splitting design of AW, together with the rationale for routing every sample through the weak augmentation τ^w before applying τ^s to half the batch, is described in

Section 3.1. Two consequences are worth highlighting in the context of the two-level vulnerability. First, the paired weak-only and weak-to-strong latents produced within a single mini-batch are precisely the supervision signal consumed by DC; without this pairing, DC would require a second encoder pass per sample, doubling encoder computation. Second, by applying strong augmentation only to a subset, AW balances two opposing failure modes observed in our ablations (Section 4.3; Appendix G.5): weak-only ablations exhibit insufficient robustness due to limited augmentation diversity, while strong-only and weak-to-strong-only ablations exhibit optimization instability under heavy perturbation.

DC. DC directly addresses the dynamics level of the two-level vulnerability. Perturbation-induced drift in the latent state corrupts the entire synthetic trajectory produced by the world model, leading to erroneous planning decisions. This problem is amplified relative to MFRL because both Q-values and transition dynamics are conditioned on encoder outputs, so a single perturbed latent propagates errors across the full rollout horizon.

To mitigate this, VIGOR enforces latent consistency by anchoring the dynamics model to weakly augmented successor states. While the dynamics model is trained on perturbed mixed latents (z_t^m) generated by AW, the DC objective regresses these predictions toward targets produced by weak augmentations, which serve as nearly noise-free anchors. This ensures that the world model produces consistent next-state predictions regardless of input perturbations, preventing the compounding of rollout errors over long horizons. By aligning predictions from OOD inputs with their in-distribution counterparts, DC stabilizes the forward dynamics and supports robust planning under visual distribution shifts.

ES. ES addresses the encoder level of the two-level vulnerability. While ES adapts the alignment objective of SODA [14], originally proposed for MFRL representation learning, its role in VIGOR is tailored to a specific failure mode in latent-space MBRL: encoder drift induced by DC's cross-augmentation supervision. A single shared encoder produces the latent representations that concurrently feed the dynamics model, the reward predictor, Q-value networks, and the MPC planner. The cross-augmentation supervision imposed by DC, which trains the dynamics on mixed latents (z_t^m), can inadvertently distort the encoder's representation. Without explicit constraints, encoder outputs tend to drift under distribution shifts, causing the dynamics-level compounding that DC is designed to prevent.

To mitigate this, VIGOR introduces an auxiliary ES loss that aligns weak-only and weak-to-strong encodings of the same frame. In VIGOR, ES acts as a structural stabilizer for the shared encoder, enabling DC to operate on consistent representations without distorting the latent geometry. The encoder thereby preserves task-relevant features under both clean and perturbed inputs, ensuring that initial latent states remain grounded for long-horizon planning in unseen environments.

Summary. VIGOR's effectiveness arises from the synergy between its three components, each targeting a different level of the world-model pipeline: AW establishes an asymmetric batching structure that provides paired latents without additional overhead; DC stabilizes the forward dynamics by anchoring predictions from perturbed mixed latents to weakly augmented targets, mitigating compounding rollout errors; and ES prevents the shared encoder from drifting under cross-augmentation pressure, ensuring that DC's dynamics-level constraint operates on stable representations. The three components are complementary: our ablations (Sections 4.3–4.4; Appendix G.5–G.7) demonstrate that removing any single component results in significant performance degradation, with the largest drops observed on Robosuite manipulation tasks under severe distribution shifts.

E Implementation Details

E.1 Code Base

We evaluate VIGOR against four MFRL baselines (DrQ-v2 [45], SVEA [16], SRM [20], and SimGRL [37]) and four MBRL baselines (TD-MPC2 [17], DreamerV3 [12], HRSSM [39], and Dr. G [7]). We use the reference implementations from RL-ViGen [48][3] for DrQ-v2, SVEA, and SRM, and the official implementations from the respective authors for SimGRL[4], DreamerV3[5], HRSSM[6], and Dr. G[7]. VIGOR is built on the official TD-MPC2 repository[8], with modifications restricted to the additional components introduced in Section 3.

E.2 Computational Resources

All experiments were conducted on a single NVIDIA A5000 GPU. On average, a complete training and evaluation run took approximately 48 hours for the DMC tasks and 72 hours for the Robosuite tasks.

E.3 Hyperparameters

To ensure a fair comparison, we reuse common settings across all baselines whenever possible and follow each method's official configuration for method-specific values. Table 4 reports hyperparameters across baselines (with method-specific values listed where they differ), and Table 5 reports the full hyperparameter set for VIGOR. VIGOR inherits TD-MPC2 defaults [17] unless otherwise noted; VIGOR-specific parameters are highlighted in yellow.

Table 4: **Hyperparameters across baselines.** Settings used in DMC and Robosuite experiments. Common values apply to all methods; method-specific values are listed per column where they differ. "N/A" denotes hyperparameters that do not apply to a given method (e.g., Dreamer-family-specific state dimensions).

Hyperparameter	DrQ-v2	SVEA	SRM	SimGRL	TD-MPC2	DreamerV3	HRSSM	Dr. G
Discount factor γ	0.99					0.997		0.99
Replay buffer size B	Unlimited (same as T_g)							
Action repeats	2			Task-specific	2	1	2	Task-specific
Frame stack k	3					1		
Pixel RGB image space	$o_t \in \mathcal{O}^{84\times84\times3}$				$o_t \in \mathcal{O}^{64\times64\times3}$			
Maximum episode length	1,000 (DMC); 500 (Robosuite)							
Batch size	512 (*walker-walk*, *walker-stand*); 256 (otherwise)			128	256	16		50
Total gradient steps T_g	1M							
Total seeding steps	4,000			1,000	2,500 (DMC); 1,250 (Robosuite)	1,024	2,500	2,500 (DMC); 1,250 (Robosuite)
Periodic evaluation steps during training	10k							
N-step TD target	3			1				
MLP hidden layer dimension	1,024				512	1,024	512	200
Latent dimension	50			100	512	1,024	512	1,024
Deterministic state dimension (for Dreamer family)	N/A					8,192	512	200
Stochastic state dimension/categories (for Dreamer family)	N/A					2,048	1,024	30
Activation function	ReLU				LayerNorm + Mish	RMSNorm + SiLU	LayerNorm + SiLU	ReLU + ELU
Target network EMA weight	1e-2			1e-2 (Critic); 5e-2 (Encoder)	1e-2	2e-2		5e-2

[3] https://github.com/gemcollector/RL-ViGen
[4] https://github.com/W-Song11/SimGRL-Code
[5] https://github.com/danijar/dreamerv3
[6] https://github.com/bit1029public/HRSSM
[7] https://github.com/JeongsooHa/DrG
[8] https://github.com/nicklashansen/tdmpc2

Table 5: **VIGOR hyperparameters.** Full set of hyperparameters used in VIGOR, which builds upon TD-MPC2 defaults. VIGOR-specific parameters are highlighted in yellow.

Hyperparameter	Value
MPC planning	
Planning Horizon H	3
Std. range	$\sigma \in [0.05, 2]$
Population size	512
Elite fraction	64
Iterations	6
Policy prior samples	24
Sampling temperature	0.5
Model learning	
Temporal coefficient λ	0.5
Reward loss coefficient c_1	0.1
Q-value loss coefficient c_2	0.1
Latent consistency loss coefficient c_3	20
Optimization	
Learning rate η	3e-4
Periodic target network (θ^-) update steps	1
Momentum coefficient for Q-network (κ)	0.99
Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Exploration schedule (std)	Linear (0.5, 0.05, 25,000 steps)
Planning horizon schedule	Linear (1, 5, 25,000 steps)
VIGOR-specific hyperparameters	
Weak augmentation τ^w	*random-shift*: `padding` $p = 4$
Strong augmentation τ^s	*random-overlay*: {`linear interpolation` $\delta = 0.5$; `Image dataset` $\mathcal{D} =$ `Places` [52]}
Augmentation ratio ζ	1.0 (half weak-only, half weak-to-strong)

F Evaluation Details

We follow the zero-shot evaluation protocol of Yuan et al. [48] and define seven distraction categories for DMC, each evaluated at two difficulty levels (*easy* and *hard*), resulting in a total of fourteen DMC evaluation settings. For Robosuite, we consider three difficulty levels (*easy*, *hard*, and *extreme*). These settings induce distribution shifts in appearance factors such as color, lighting, background, and camera position, providing a rigorous testbed for zero-shot visual generalization.

F.1 Environments and Tasks

We benchmark VIGOR on two widely used continuous-control suites: DMC and Robosuite. Representative environments are illustrated in Figure 9.

In **DMC** [40], we consider six visuomotor control tasks: *cartpole-swingup*, where the agent must swing up and balance a cartpole; *finger-spin*, which requires continuous spinning of a planar finger; *walker-walk*, which trains a bipedal walker to move forward stably; *walker-stand*, which focuses on maintaining balance in an upright posture; *cheetah-run*, where the agent learns high-speed forward locomotion; and *reacher-easy*, which involves controlling a 2-DoF arm to reach a target location.

For **Robosuite** [55], we evaluate four robotic manipulation tasks: *door-opening*, where the robot arm must open a hinged door; *nut-assembly*, which requires placing a round nut onto a peg; *peg-in-hole*, where a peg must be inserted into a narrow slot; and *lifting*, which involves grasping and lifting a cube.

F.2 Baseline Descriptions

This subsection provides detailed descriptions of the baselines used in our experiments, complementing the summary in Section 4.1.

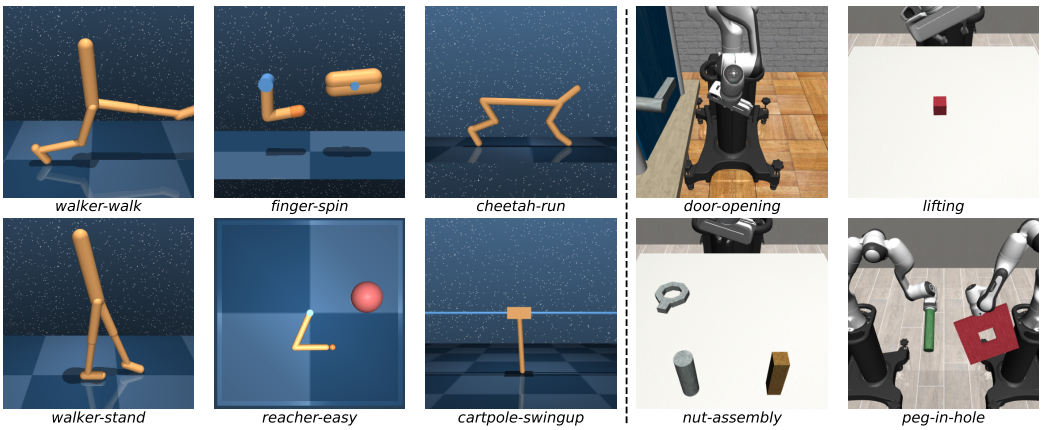


Figure 9: **Environments and tasks.** Six visuomotor control tasks from DMC [40] and four robotic manipulation tasks from Robosuite [55].

Selection criteria. We select baselines according to the following criteria: (i) The method must provide an official codebase in its original paper or in a recognized benchmarking framework (e.g., RL-ViGen [48]). We exclude methods whose scores can only be reproduced from unofficial or re-implemented code, as this may introduce unreliable or biased results. (ii) The method must explicitly target visual robustness or zero-shot visual generalization in RL. (iii) Reproduced scores must be broadly consistent with those reported in the original publication; methods with large unexplained discrepancies are excluded.

We additionally include three strong baselines (DrQ-v2, TD-MPC2, and DreamerV3) that do not explicitly target zero-shot visual generalization but serve essential reference roles: DrQ-v2 as a widely adopted MFRL baseline for sample efficiency, TD-MPC2 as VIGOR's backbone, and DreamerV3 as a state-of-the-art reconstruction-based MBRL method. Including these methods enables direct assessment of how much zero-shot robustness VIGOR adds over its backbone and how general-purpose methods perform under distribution shifts without explicit robustness mechanisms.

MFRL baselines. (i) DrQ-v2 [45] combines random-shift augmentation with an off-policy actor–critic architecture, achieving strong sample efficiency on pixel-based continuous control. (ii) SVEA [16] stabilizes off-policy Q-learning by maintaining a weighted mixture of augmented and clean Q-targets, preventing the value function from being corrupted by augmentation noise. (iii) SRM [20] applies spectrum-based augmentations in the frequency domain to enhance robustness to spatial corruptions. (iv) SimGRL [37] combines random-overlay with random-shift augmentation, injecting task-irrelevant background perturbations to mitigate observational overfitting.

MBRL baselines. (v) TD-MPC2 [17] achieves strong performance via decoder-free latent-space planning with temporal-difference learning. It serves as VIGOR's backbone. (vi) DreamerV3 [12] employs latent imagination with pixel-level reconstruction and achieves competitive performance across diverse control tasks. (vii) HRSSM [39] extends DreamerV3 with a bisimulation-based hybrid RSSM to extract task-relevant latent representations without pixel-level reconstruction. (viii) Dr. G [7] extends Dreamer [10] with dual contrastive objectives and a recurrent inverse-dynamics loss, explicitly targeting zero-shot visual generalization. It is the most directly comparable prior method.

F.3 Visual Distraction Categories

We consider seven types of distractions. For DMC, each type is evaluated separately at two difficulty levels (*easy* and *hard*); for Robosuite, the RL-ViGen evaluation splits (*easy*, *hard*, and *extreme*) combine these factors within predefined scenes (Appendix F.5). The seven types are defined as follows:

- *object-color*: Change the body color of the agent.
  - Uniformly sample the parameters of the color (i.e., *(r,g,b)*) from the pre-defined distribution for each difficulty.

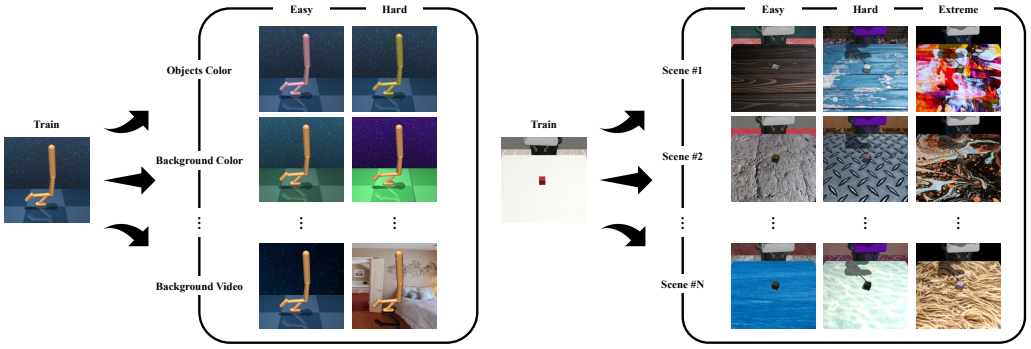
- *background-color*: Change the background color of the environment (e.g., terrain grid or sky).
  - Uniformly sample the parameters of the color (i.e., *(r,g,b)*) from the pre-defined distribution for each difficulty.
- *background-video*: Overlay the background with the randomly sampled natural video.
  - Sample a random video with the same width and height as the original image from a set of natural videos [38].
  - Overlay the video only to the background sky for *easy* and to all backgrounds, including the ground terrain other than the agent, for *hard* difficulty.
- *cam-pos*: Change the position of the tracking camera's focus by randomly adding noise offset.
  - Let the initial position of the tracking camera's focus be $X_{\text{cam}} = (c_x, c_y, c_z)$ in Euclidean space.
  - Sample a random offset $\delta \in \mathbb{R}^3$ from the uniform distribution with different bounds: $\mathcal{U}(-0.08, 0.08)$ for *easy* and $\mathcal{U}(-0.15, 0.15)$ for *hard* difficulty.
  - Inject the offset to the initial position of the camera; $X_{\text{cam}} = (c_x + \delta_x, c_y + \delta_y, c_z + \delta_z)$.
- *light-position*: Change the position and orientation of the tracking light of the agent.
  - Following the approach used in [38], the tracking light's coordinate is parameterized as the spherical coordinate; (ϕ, ϑ, ρ) where ϕ is azimuth, ϑ is inclination, and ρ is the radius of the sphere.
  - Sample ϕ from the normal distribution $\mathcal{N}(\pi/6, 1)$ for *easy* and $\mathcal{N}(\pi/3, 1)$ for *hard* difficulty.
  - Sample $\vartheta \sim \mathcal{N}(2\pi, 1)$ and transform the initial pose of the tracking light X_{light} to (ϕ, ϑ, ρ), where $\rho = \|X_{\text{light}}\|$ is the radius (Euclidean norm of the initial position).
- *light-color*: Change the color of the tracking light of the agent.
  - Uniformly sample the parameters of the color (i.e., *(r,g,b)*) from the pre-defined distribution for each difficulty.
- *moving-light*: Rotate the tracking light around the agent.
  - As in *light-position*, the spherical coordinate of the tracking light is randomly initialized as (ϕ, ϑ, ρ).
  - Let the speed of azimuth rotation as $\Delta_\phi = \pi/200$ for *easy* and $\Delta_\phi = \pi/100$ for *hard* difficulty.
  - Rotate the tracking light counterclockwise along the azimuth axis at every time step; $(\phi, \vartheta, \rho) \leftarrow (\phi, \vartheta, \rho) + (\Delta_\phi, 0, 0)$.

F.4 Evaluation Protocol

For DMC, each of the seven distraction types listed above includes two difficulty levels: *easy* and *hard*. Robosuite follows the predefined evaluation splits (*eval-easy*, *eval-hard*, and *eval-extreme*) as introduced in RL-ViGen [48].

F.5 Evaluation Scenes

Representative evaluation scenes for DMC and Robosuite are shown in Figures 10(a) and 10(b), respectively. Agents are trained in the default *clean* setting and then evaluated *zero-shot* under diverse unseen distractions. Figure 11 illustrates the four Robosuite tasks (*door-opening*, *nut-assembly*, *peg-in-hole*, and *lifting*) under increasingly severe visual distractions, from *eval-easy* to *eval-extreme*.



(a) Evaluation scenes in DMC (b) Evaluation scenes in Robosuite

Figure 10: **Evaluation scenes.** Agents are trained in clean environments and evaluated *zero-shot* on seven distraction types for DMC (two difficulty levels: *easy* and *hard*) and ten distraction scenes for Robosuite (three difficulty levels: *easy*, *hard*, and *extreme*).

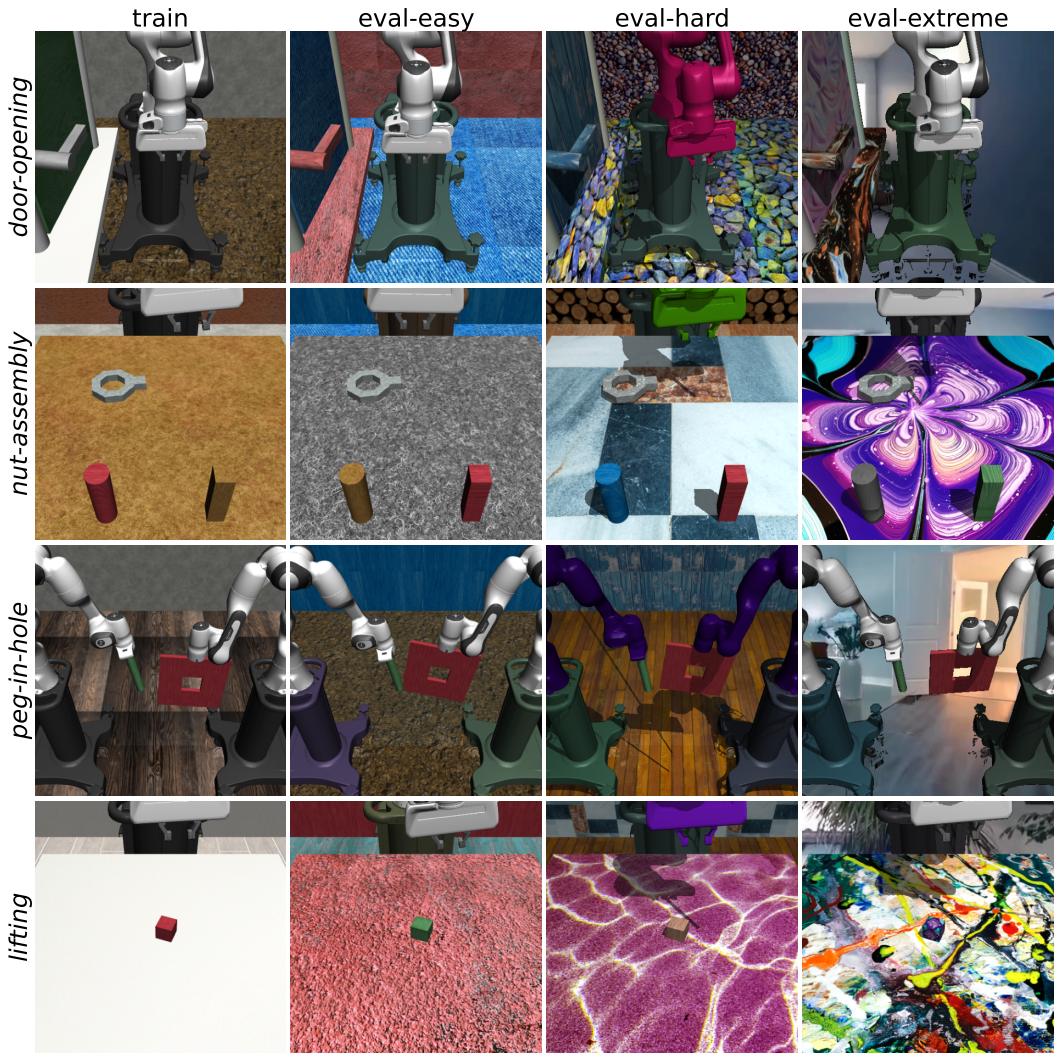


Figure 11: **Visual generalization evaluation in Robosuite.** Visual generalization setup in Robosuite, which includes four robotic manipulation tasks: *door-opening*, *nut-assembly*, *peg-in-hole*, and *lifting*. Each column represents a different level of visual distraction, from left to right: *train*, *eval-easy*, *eval-hard*, and *eval-extreme*. Each row shows a different task.

G Supplementary Results

This appendix complements Section 4.2 with additional figures, tables, and analyses.

G.1 Zero-Shot Generalization

This subsection provides a detailed breakdown of our zero-shot generalization results, complementing the aggregate scores in Section 4.2. Full results across distraction types and tasks are reported in Figures 12 and 13, with per-task averages summarized in Table 6.

Performance across distraction types and difficulty levels. Figure 12 reports zero-shot generalization on DMC across fourteen conditions (seven distraction types $\times$ two difficulty levels: *easy* and *hard*) and on Robosuite across three difficulty levels (*easy*, *hard*, and *extreme*); each panel additionally shows the *original* (distraction-free) condition as a reference. Across all methods, returns tend to degrade monotonically as distraction difficulty increases (*original* $\rightarrow$ *easy* $\rightarrow$ *hard* $\rightarrow$ *extreme*), reflecting the inherent challenge of distribution shifts in visual control. However, the magnitude of this degradation differs substantially across methods.

On DMC, baseline behavior diverges sharply across distraction types. TD-MPC2, despite being one of the strongest latent-space MBRL methods under clean settings, exhibits sharp drops under hard distractions on *background-color* ($\approx$0.32), *background-video* ($\approx$0.05), and *light-color* ($\approx$0.32), highlighting its sensitivity to severe visual shifts. DreamerV3 and HRSSM follow similar trajectories, with HRSSM degrading even under *easy cam-pos* perturbations. SimGRL maintains relatively stable performance across most distractions and achieves the highest score on *background-video* hard ($\approx$0.79), reflecting its training strategy that injects diverse background perturbations. In contrast, VIGOR sustains high returns across nearly all distraction types and difficulty levels, ranking first or within a narrow margin of the best method on every distraction type except *background-video*. On Robosuite, where most baselines collapse to near-zero performance under hard and extreme shifts, VIGOR maintains the highest returns at every difficulty level, demonstrating its robustness under severe distribution shifts.

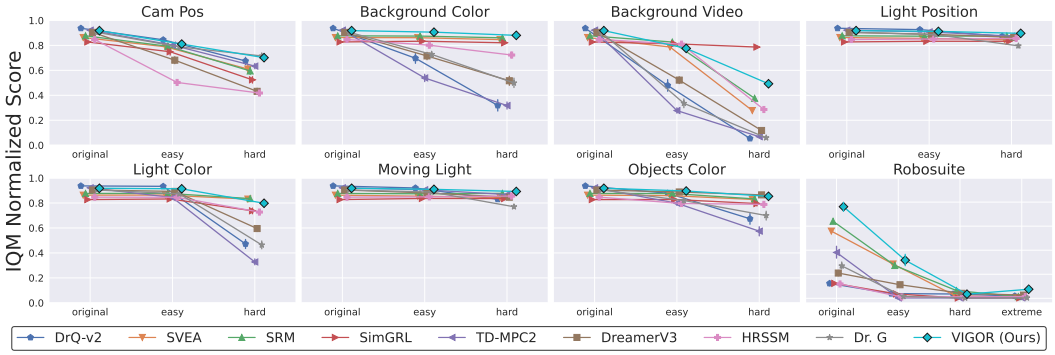


Figure 12: **Zero-shot generalization across distraction types and difficulty levels.** Each subplot reports IQM normalized scores for a single distraction type across difficulty levels. DMC: seven distraction types $\times$ three settings (*original*, *easy*, *hard*). Robosuite: four settings (*original*, *easy*, *hard*, *extreme*).

Task-level performance breakdown. Table 6 and Figure 13 provide a detailed breakdown of zero-shot generalization performance across all tasks in DMC and Robosuite. VIGOR ranks first among all baselines on every individual task in both benchmarks, demonstrating that its advantage is not driven by a few favorable cases but by uniformly strong performance across the task set.

On DMC, VIGOR achieves the highest average return (777.6), surpassing the second-best baseline (SRM, 752.0) by 3.4%. At the task level, VIGOR attains the strongest scores on all six tasks, exceeding the respective second-best methods by 2.0% on *walker-walk* (811.3 vs. DreamerV3, 795.5), 1.0% on *finger-spin* (809.1 vs. SRM, 801.5), 4.0% on *cheetah-run* (461.6 vs. SRM, 443.7), 0.5% on *walker-stand* (938.6 vs. SVEA, 933.5), 2.7% on *reacher-easy* (899.0 vs. SimGRL, 875.5), and 0.0% on *cartpole-swingup* (746.0 vs. SRM, 745.7).

Table 6: **Zero-shot generalization scores.** Mean episode returns ($\pm$ std) averaged over all distraction types and difficulty levels for each task. Boldface denotes the best score per task.

Env	Task	DrQ-v2	SVEA	SRM	SimGRL	TD-MPC2	DreamerV3	HRSSM	Dr. G	VIGOR (Ours)
DMC	walker-walk	731.5 ± 344.4	738.5 ± 177.0	730.2 ± 153.6	719.3 ± 128.5	642.2 ± 344.8	795.5 ± 279.0	775.3 ± 153.0	769.8 ± 331.9	$\mathbf{811.3 \pm 201.7}$
	finger-spin	721.6 ± 373.0	739.8 ± 246.4	801.5 ± 319.8	787.9 ± 242.5	643.9 ± 336.7	268.2 ± 301.6	734.6 ± 319.1	584.2 ± 321.9	$\mathbf{809.1 \pm 212.2}$
	cheetah-run	379.3 ± 295.5	429.1 ± 179.1	443.7 ± 228.9	206.1 ± 25.6	318.4 ± 243.4	364.3 ± 281.2	323.8 ± 85.7	413.7 ± 286.7	$\mathbf{461.6 \pm 176.7}$
	walker-stand	754.3 ± 313.7	933.5 ± 132.0	932.7 ± 105.0	929.5 ± 110.3	737.1 ± 320.5	891.9 ± 233.4	915.7 ± 116.0	834.6 ± 274.9	$\mathbf{938.6 \pm 72.9}$
	reacher-easy	501.8 ± 449.9	829.8 ± 322.1	858.2 ± 291.8	875.5 ± 286.8	621.9 ± 422.5	758.5 ± 345.7	571.2 ± 427.5	500.9 ± 388.8	$\mathbf{899.0 \pm 232.3}$
	cartpole-swingup	668.3 ± 289.2	686.5 ± 281.8	745.7 ± 143.4	742.7 ± 87.5	631.1 ± 241.7	733.5 ± 212.5	657.8 ± 192.1	638.7 ± 274.7	$\mathbf{746.0 \pm 173.8}$
	Average	626.1 ± 375.1	726.2 ± 279.4	752.0 ± 269.8	710.2 ± 292.9	599.1 ± 349.8	635.3 ± 363.2	663.1 ± 308.4	623.7 ± 347.5	$\mathbf{777.6 \pm 241.5}$
Robosuite	door-opening	7.6 ± 32.2	76.1 ± 98.1	92.5 ± 90.9	1.0 ± 0.4	1.0 ± 1.0	35.0 ± 41.5	6.8 ± 12.7	13.2 ± 69.0	$\mathbf{143.9 \pm 201.6}$
	lifting	27.0 ± 25.1	20.6 ± 58.9	25.5 ± 78.7	21.6 ± 74.1	1.8 ± 5.0	23.7 ± 27.7	1.5 ± 2.7	1.3 ± 2.5	$\mathbf{35.4 \pm 74.9}$
	nut-assembly	0.6 ± 1.6	11.8 ± 31.7	10.2 ± 26.8	1.3 ± 0.0	0.7 ± 1.1	3.4 ± 21.7	5.2 ± 3.8	0.7 ± 1.0	$\mathbf{15.5 \pm 39.3}$
	peg-in-hole	128.8 ± 45.0	222.6 ± 86.6	239.1 ± 114.2	137.3 ± 12.9	157.4 ± 35.9	195.2 ± 39.1	166.9 ± 24.1	200.8 ± 28.7	$\mathbf{332.6 \pm 114.2}$
	Average	41.0 ± 59.9	82.8 ± 111.9	91.8 ± 123.4	40.3 ± 68.0	40.2 ± 70.1	64.3 ± 83.5	45.1 ± 71.7	54.0 ± 92.8	$\mathbf{131.8 \pm 176.1}$

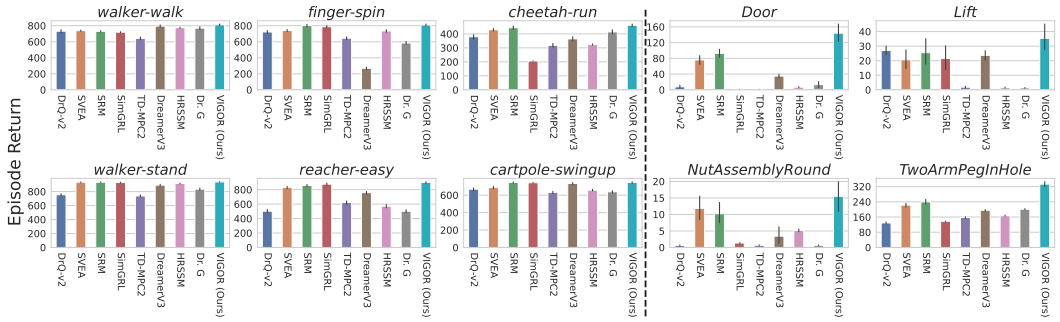


Figure 13: **Zero-shot generalization across tasks.** Mean episode returns across six DMC tasks and four Robosuite tasks, averaged over all distraction types and difficulty levels. Error bars denote standard deviations.

On Robosuite, the performance gains are substantially more pronounced. VIGOR achieves the highest average return (131.8), surpassing the second-best baseline (SRM, 91.8) by 43.6%, and attains the best scores on all four tasks: *door-opening* (143.9, +55.6% over SRM, 92.5), *lifting* (35.4, +31.1% over DrQ-v2, 27.0), *nut-assembly* (15.5, +31.4% over SVEA, 11.8), and *peg-in-hole* (332.6, +39.1% over SRM, 239.1). Many baselines degrade to near-random performance in this setting, whereas VIGOR maintains robustness under visually challenging manipulation conditions.

Summary. These findings confirm that VIGOR's three components (AW, DC, and ES) are essential for mitigating severe distribution shifts. By jointly promoting augmentation-invariant latent dynamics, stabilizing transition predictions under perturbed inputs, and preventing encoder drift, VIGOR achieves the highest zero-shot generalization on every task across both benchmarks, not only in aggregate but also at the individual task level. This consistent dominance corroborates that conventional latent-space MBRL is fragile to OOD perturbations, whereas VIGOR's latent-space consistency enables robust generalization without test-time adaptation.

G.2 Sample Efficiency

This subsection provides a detailed per-task comparison of sample efficiency, complementing the aggregate results in Section 4.2.

Training curves. Figure 14 presents per-task learning curves for DMC and Robosuite, reporting IQM normalized scores across all methods. While no single method dominates across all tasks, VIGOR consistently ranks among the top performers. Notably, VIGOR maintains this level of efficiency despite being trained with more challenging objectives that incorporate visual perturbations, confirming that robustness does not come at the cost of efficiency.

Evaluation protocol. To quantify sample efficiency, we adopt the evaluation protocol of Mai et al. [30]. We define TD-MPC2 as the oracle baseline, since it serves as VIGOR's backbone and provides the most direct reference for assessing the efficiency cost of the added robustness components. We

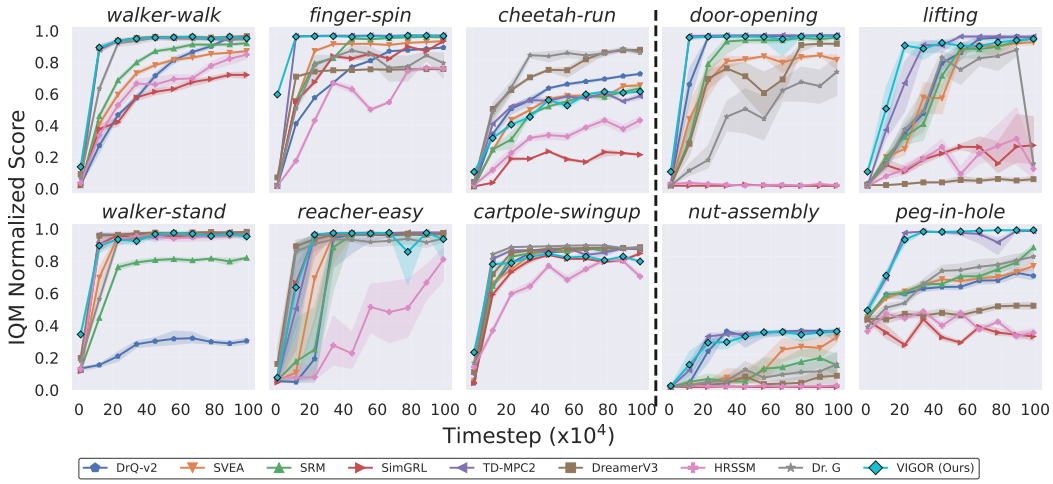


Figure 14: **Per-task sample efficiency.** Learning curves reported as IQM normalized scores across methods. Shaded regions denote 95% stratified bootstrap confidence intervals.

then measure the number of environment episodes required for each non-oracle method to reach 25%, 50%, and 75% of TD-MPC2's asymptotic return. If an agent fails to reach a threshold within the training budget, we assign the maximum episode count as a penalty (1,000 for DMC and 2,000 for Robosuite).

Quantitative results. As shown in Table 7, VIGOR reaches every threshold in the fewest episodes among all non-oracle baselines on both benchmarks, demonstrating that the added robustness components incur minimal cost in sample efficiency relative to the TD-MPC2 backbone.

On DMC, VIGOR reaches the 25%, 50%, and 75% thresholds in 395, 412, and 422 episodes, respectively. The second-best baseline at each threshold is Dr. G (415 at 25%; 540 at 50%) or DreamerV3 (553 at 75%); VIGOR requires 4.8% to 23.7% fewer episodes than the second-best baseline at each threshold. On Robosuite, the efficiency advantage is more pronounced: VIGOR requires 655, 720, and 740 episodes to reach the three thresholds, while the second-best baseline requires 1,240 (SRM), 1,340 (SRM), and 1,365 (DrQ-v2) episodes, corresponding to roughly 1.8–1.9× more interactions across all thresholds. Many baselines fail to reach the higher thresholds within the training budget, indicating that VIGOR's sample-efficiency advantage amplifies under visually complex manipulation tasks. These results confirm that AW, DC, and ES preserve the strong sample efficiency of the TD-MPC2 backbone while enabling robustness to unseen distractions.

Table 7: **Sample efficiency.** Mean number of episodes (lower is better) required to reach 25%, 50%, and 75% of the oracle baseline (TD-MPC2), following the protocol of Mai et al. [30]. Boldface denotes the best score per row (excluding the oracle).

Env	Percentile	DrQ-v2	SVEA	SRM	SimGRL	DreamerV3	HRSSM	Dr. G	VIGOR (Ours)
DMC	25%	772	493	623	625	437	802	415	**395**
	50%	883	605	702	642	550	833	540	**412**
	75%	895	640	722	657	553	847	617	**422**
Robosuite	25%	1,335	1,450	1,240	1,985	1,840	2,000	1,685	**655**
	50%	1,360	1,700	1,340	1,990	1,840	2,000	1,925	**720**
	75%	1,365	1,780	1,370	1,990	1,845	2,000	1,965	**740**

Statistical analysis. Figure 15 presents empirical cumulative distribution functions (ECDFs) of episode returns across tasks. ECDFs capture the full distribution of training efficiency and illustrate how quickly each method reaches target performance thresholds. VIGOR exhibits ECDF profiles that closely track TD-MPC2 while surpassing all other baselines, further validating its efficiency–robustness balance.

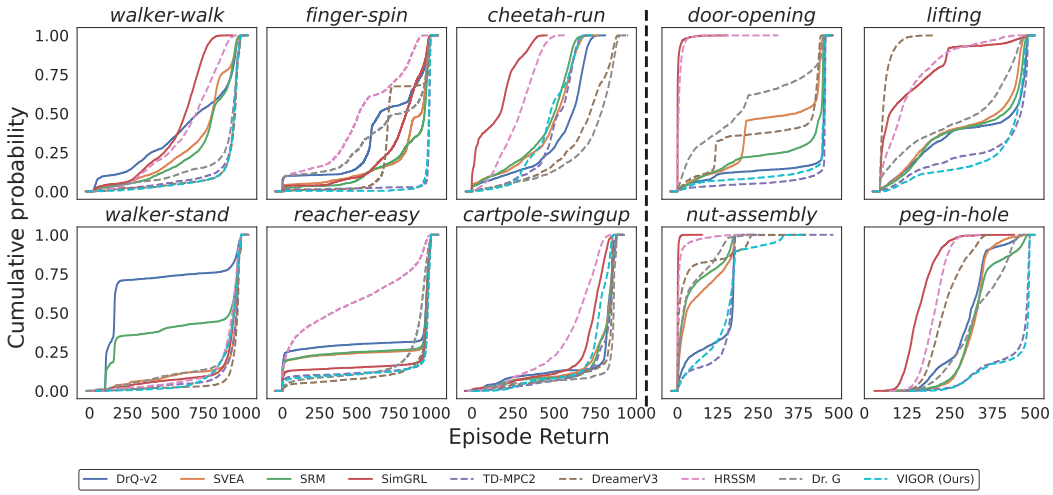


Figure 15: **Statistical comparison of sample efficiency.** ECDFs of episode returns across tasks, computed from training statistics. These distributions complement the threshold-based comparisons in Table 7.

Summary. These results demonstrate that AW, DC, and ES preserve the sample efficiency of latent-space MBRL while enabling robust zero-shot generalization, incurring only modest overhead relative to the TD-MPC2 backbone.

G.3 DMC 100K

The main experiments (Section 4.2) train all methods for 1M gradient steps, a budget sufficient for most methods to reach asymptotic performance. To stress-test VIGOR's efficiency–robustness trade-off under severe data constraints, we conduct an additional evaluation on DMC with only 100K gradient steps (10% of the standard budget), following prior work [2, 22, 25, 34, 35, 46].

Motivation. Evaluating sample efficiency requires assessing two conditions: (i) how *quickly* an agent reaches its asymptotic performance, and (ii) how *high* that asymptote is. Under the full 1M-step budget, most methods converge, making it difficult to distinguish whether strong generalization stems from robust representations or simply from prolonged training. The 100K budget isolates early-phase learning dynamics: methods that build robust latent structures efficiently should generalize well even before convergence, while methods that rely on extended training to compensate for fragile representations will degrade disproportionately. Furthermore, zero-shot generalization performance is bounded by the training-time return, so any efficiency gap directly tightens the robustness ceiling, exacerbating the efficiency–robustness trade-off.

Experimental setup. All methods are trained for 100K gradient steps on six DMC tasks under the same RL-ViGen protocol used in the main experiments (Section 4.1). Hyperparameters are identical to the 1M-step setting; no method-specific tuning is performed for the reduced budget. We evaluate both sample efficiency (IQM normalized scores as a function of training steps) and zero-shot generalization (IQM normalized scores under unseen visual distractions after 100K steps). All results are averaged over ten random seeds.

Results. Figure 16 reports sample-efficiency curves (left) and zero-shot generalization (right).

- **Sample efficiency.** TD-MPC2 achieves the highest IQM normalized score ($\approx$0.80), with VIGOR closely following ($\approx$0.78). DreamerV3 ranks third ($\approx$0.61), followed by Dr. G ($\approx$0.58) and SVEA ($\approx$0.56). Two patterns are notable. First, the strongest sample-efficient methods are dominated by MBRL backbones (TD-MPC2, VIGOR, DreamerV3, and Dr. G), confirming that latent-space world models provide an efficiency advantage even under severe data constraints. However, not all MBRL methods benefit equally: HRSSM ($\approx$0.25) falls below all MFRL baselines under this tight budget, suggesting that its more complex world-model objective (bisimulation-based hybrid RSSM) is harder to optimize under limited

interaction. Second, TD-MPC2 and VIGOR substantially outpace DreamerV3, Dr. G, and HRSSM, highlighting the efficiency of TD-MPC2's MPC-based planning over recurrent world-model rollouts. VIGOR closely matches TD-MPC2's trajectory throughout training despite the addition of AW, DC, and ES, indicating that these components introduce minimal training overhead (Appendix G.4).

- **Zero-shot generalization.** VIGOR achieves the strongest zero-shot generalization, substantially outperforming all baselines. The contrast with the sample-efficiency ranking is striking: TD-MPC2, the most efficient method, drops to mid-tier generalization performance. This confirms that sample efficiency alone does not guarantee robustness: without explicit consistency mechanisms, the representations learned under a constrained budget remain fragile to unseen visual shifts. VIGOR closes this gap: it retains near-TD-MPC2 efficiency while achieving the highest generalization, demonstrating that latent-space consistency is established early in training rather than requiring extended training to converge.

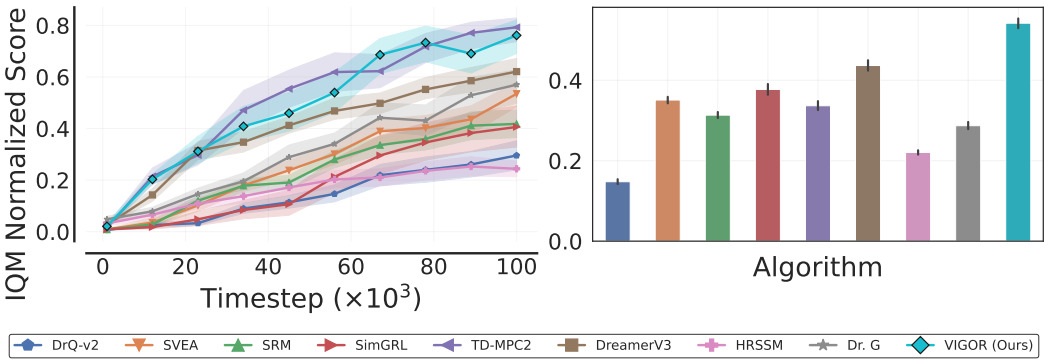


Figure 16: **DMC 100K results.** Left: Sample efficiency (IQM normalized scores) under a 100K-step training budget. Right: Zero-shot generalization (IQM) after 100K steps.

Summary. These results demonstrate that VIGOR's efficiency–robustness advantage is not an artifact of extended training. Even under a $10\times$ reduction in training budget, VIGOR achieves the strongest trade-off between sample efficiency and zero-shot generalization, confirming that latent-space consistency is built into the world-model representations from the early stages of training.

G.4 Wall-Clock Training Efficiency

To complement the sample-efficiency analysis, we evaluate VIGOR's wall-clock training efficiency, which is important for assessing practical usability.

Experimental setup. VIGOR and TD-MPC2 are trained under an identical RL-ViGen protocol on a single NVIDIA A5000 (24 GB) GPU, using the same dataloaders, environment settings, and hyperparameters described in Appendices E and F.

Overall results. As shown in Figure 17 (left), VIGOR closely matches TD-MPC2 in sample efficiency: IQM normalized scores as a function of environment steps are nearly identical, and both methods reach asymptotic performance at similar sample budgets. Figure 17 (right) analyzes performance as a function of wall-clock time aggregated across all tasks. TD-MPC2 trains faster during the early stages (the gap peaks at 0.37 around hours 3–4), but the difference narrows as training progresses. Over a 24-hour window, the average normalized-score gap is approximately 0.17, and it stays within 0.06–0.14 after hour 14.

Wall-clock overhead analysis. The wall-clock overhead of VIGOR is modest and stems primarily from strong augmentations, which introduce additional preprocessing (e.g., *background-video* composition and external image loading). VIGOR mitigates this cost through its batch-splitting design: AW partitions a single batch into weak-only and weak-to-strong subsets rather than creating duplicate views, and all augmentations are applied *before* encoding. As a result, the encoder processes the

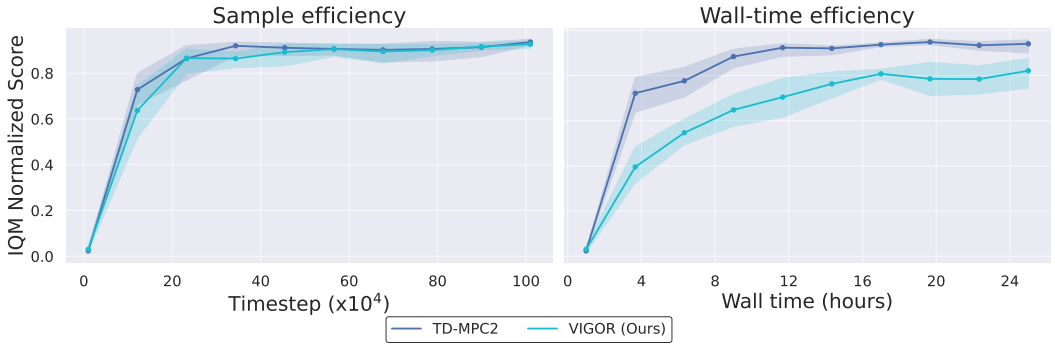


Figure 17: **Sample efficiency and wall-clock training efficiency.** Left: Learning curves as a function of environment steps. Right: Learning curves as a function of wall-clock time.

same total number of samples as in the original TD-MPC2 pipeline, and the DC and ES losses add only lightweight computations on top of the existing latent representations. This design explains why VIGOR incurs only a minor slowdown despite incorporating substantially stronger generalization mechanisms.

Per-task analysis. Figure 18 provides a per-task breakdown across all ten environments. TD-MPC2 maintains a small speed advantage in wall-clock time, but VIGOR remains close throughout training and the difference narrows as both methods converge. Table 8 summarizes the IQM normalized-score differences (TD-MPC2 minus VIGOR) at fixed wall-clock checkpoints. The differences are largest in the first several hours and diminish toward later stages (at hour 14 the gap drops to 0.06), confirming that VIGOR's overhead becomes less pronounced over time.

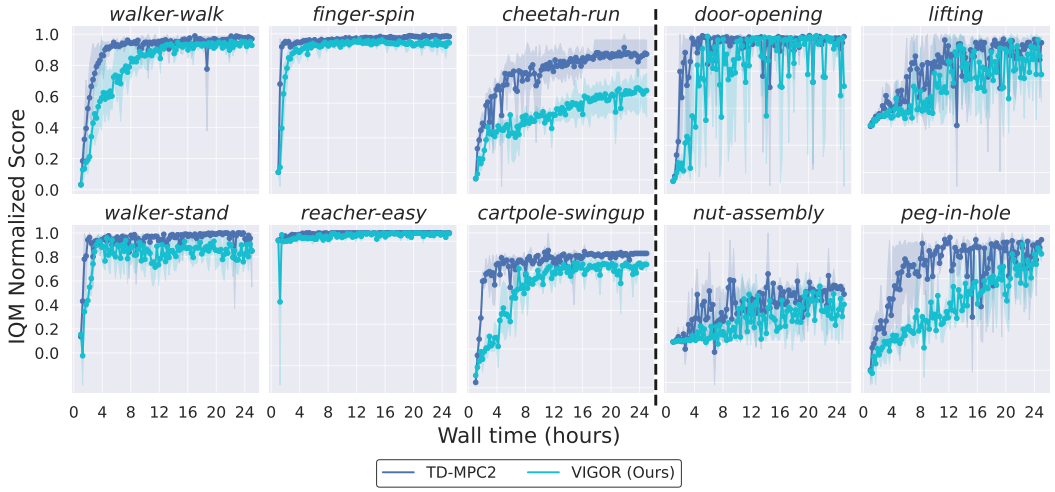


Figure 18: **Per-task wall-clock training efficiency.** IQM normalized scores as a function of wall-clock time for TD-MPC2 and VIGOR across all DMC and Robosuite tasks.

Summary. VIGOR introduces only a modest wall-clock overhead relative to TD-MPC2, which diminishes as training progresses. These results confirm that VIGOR preserves practical training efficiency while delivering substantially improved zero-shot robustness.

G.5 Augmentation Ablation Studies

This subsection provides supplementary analyses for the augmentation ablations in Sections 4.3 and 4.4, which respectively present the AW ablation results (zero-shot generalization in Table 2; sample efficiency in Figure 5) and the augmentation-family experiment (Figure 7). Here, we focus on patterns not discussed in the main text: per-difficulty performance and variance characteristics of each

Table 8: **Wall-clock training efficiency.** Difference in IQM normalized scores (TD-MPC2 minus VIGOR) at fixed wall-clock checkpoints across all tasks. Values near zero indicate matched training speed; positive values indicate a slight advantage for TD-MPC2.

WALL-CLOCK TIME (HOURS)	IQM NORMALIZED SCORE DIFFERENCE
1	−0.01
2	0.24
3	0.37
4	0.37
5	0.26
6	0.20
7	0.24
8	0.26
9	0.19
10	0.15
11	0.24
12	0.12
13	0.18
14	**0.06**
15	0.10
16	0.07
17	0.08
18	0.12
19	0.14
20	0.13
21	0.13
22	0.10
23	0.12
24	0.12
AVERAGE	0.17

AW variant, the ζ-sweep's cross-benchmark dynamics, sample-efficiency convergence patterns, and a qualitative comparison of the augmentation families used in the augmentation-agnostic experiment.

Per-difficulty and variance analysis of AW variants. The main text reports aggregate performance for each variant (Table 2). Here, we examine the per-difficulty and variance patterns that reveal *how* each variant fails.

- **DMC.** The gap between variants widens from easy to hard. `SO` achieves 423.8 ± 421.9 (easy) and 374.9 ± 401.0 (hard), with standard deviations nearly equal to the means throughout, indicating that learning is highly unstable and seed-dependent. `WTSO` shows a similar pattern (550.6 ± 377.6 easy; 481.3 ± 358.3 hard) with large variance, confirming that persistent strong exposure without a stable anchor produces unreliable training. `WO`, by contrast, maintains lower relative variance (773.9 ± 246.0 easy; 606.6 ± 326.8 hard) and converges fast (Figure 5, left), but its generalization ceiling is bounded by limited perturbation diversity. VIGOR achieves both the highest returns and the lowest variance at both levels (863.9 ± 155.4 easy; 781.1 ± 261.2 hard), demonstrating that AW's asymmetric splitting stabilizes optimization while maintaining robustness.
- **Robosuite.** The per-difficulty breakdown reveals a sharper separation, with all variants exhibiting monotonic degradation from easy to extreme. `WO` achieves moderate returns at the easy level (134.2 ± 143.8) but degrades steadily under harder shifts (112.6 ± 118.0 hard; 93.4 ± 96.9 extreme), confirming that weak augmentations alone provide insufficient robustness in visually complex manipulation. `SO` achieves higher easy-level returns (187.1 ± 182.2) but follows the same degradation pattern (137.3 ± 137.6 hard; 121.3 ± 122.6 extreme), with standard deviations consistently matching or exceeding the means, indicating highly unstable learning. `WTSO` improves at the easy level (297.6 ± 155.2) but similarly degrades under harder shifts (138.1 ± 131.3 hard; 130.1 ± 120.7 extreme; 79% overall). Only AW maintains the highest returns at every difficulty level (332.0 easy; 236.7 hard; 146.0 extreme) with the strongest overall performance.

The cross-benchmark contrast is instructive: on DMC, `WO` preserves training efficiency but undergeneralizes; on Robosuite, `WO` fails entirely because the visual complexity of manipulation scenes demands stronger perturbation exposure. Only AW's asymmetric splitting simultaneously provides both signals: stability from the weak-only partition and diversity from the weak-to-strong partition.

Mixture-ratio sensitivity (ζ-sweep). The mixture ratio $\zeta = |\mathcal{I}^w|/|\mathcal{I}^{ws}|$ controls the balance between weak-only and weak-to-strong samples within each batch. Table 2 shows that ζ=4, ζ=2, and ζ=0.5 all underperform ζ=1 on average, but the nature of the degradation differs across benchmarks:

- On DMC, increasing the weak-only fraction improves performance (ζ=4: $96\% > \zeta$=2: $94\% > \zeta$=0.5: 88%). Per-difficulty data show that this advantage stems primarily from the easy level, where ζ=4 (854.4 ± 176.5) nearly matches VIGOR (863.9 ± 155.4). At the hard level, the gap widens (736.3 ± 274.3 vs. 781.1 ± 261.2), indicating that excessive weak-only bias sacrifices robustness under stronger shifts.
- On Robosuite, the trend reverses: ζ=0.5 (87%) outperforms ζ=2 (80%) and ζ=4 (75%). The advantage of ζ=0.5 is most pronounced at the hard level (172.2 ± 157.4 vs. 144.7 ± 144.5 for ζ=2 and 151.8 ± 156.4 for ζ=4), confirming that severe shifts demand stronger perturbation exposure. Nevertheless, ζ=0.5 still falls short of ζ=1 at every level, indicating that excessive strong exposure impairs optimization even when robustness demands are high.

These results confirm that the balanced mixture (ζ=1) is a deliberate design choice. The optimal ratio reflects a trade-off between training stability (favoring more weak-only samples) and generalization coverage (favoring more weak-to-strong samples), with ζ=1 providing the best balance across both benchmarks.

Sample efficiency of AW variants. Figure 5 reveals convergence patterns that complement the generalization analysis.

For the AW variants (left panel), AW and `WO` converge at comparable rates, confirming that the weak-only partition in AW preserves training stability. `SO` exhibits wide confidence intervals throughout training and fails to reach meaningful returns, while `WTSO` converges but to a substantially lower asymptote.

For the ζ-sweep (right panel), all variants converge to similar asymptotic levels, but ζ=0.5 exhibits slower early-phase convergence due to its higher proportion of strongly perturbed samples. These patterns reinforce the main-text finding that AW uniquely balances efficiency and robustness: alternative strategies sacrifice one or the other.

Qualitative comparison of augmentation families. The augmentation-agnostic experiment in Section 4.4 replaces the default strong augmentation (*random-overlay*) with *random-conv* and *SRM*, two families from distinct perturbation domains. Figure 19 visualizes the qualitative differences on *walker-walk*:

- *Random-overlay* blends a randomly sampled natural image into the background, introducing contextual replacement that alters the global scene appearance while preserving the agent's shape and texture.
- *Random-conv* applies a random convolution kernel to each frame, producing spatially smooth color and texture shifts without modifying scene structure. The perturbation is local and frequency-preserving.
- *SRM (Spectrum Random Masking)* masks contiguous bands in the Fourier spectrum, removing mid- and high-frequency image components. The resulting perturbation is global and frequency-destructive, qualitatively distinct from both overlay and convolution.

The rightmost column of Figure 19 shows the absolute pixel-wise difference between *random-overlay* and *SRM*, highlighting their distinct spectral characteristics. Despite these fundamental differences, all three variants maintain strong zero-shot generalization (Figure 7), confirming that latent-space consistency, not the augmentation choice, is the source of robustness.

Summary. These supplementary analyses reinforce three conclusions from the main text. First, AW's asymmetric batch splitting is the only augmentation strategy that preserves both training stability and zero-shot robustness across benchmarks with fundamentally different visual complexities; per-difficulty and variance analysis reveals that alternative strategies fail through either instability (SO, WTSO) or insufficient diversity (WO). Second, the balanced mixture ratio (ζ=1) reflects a principled

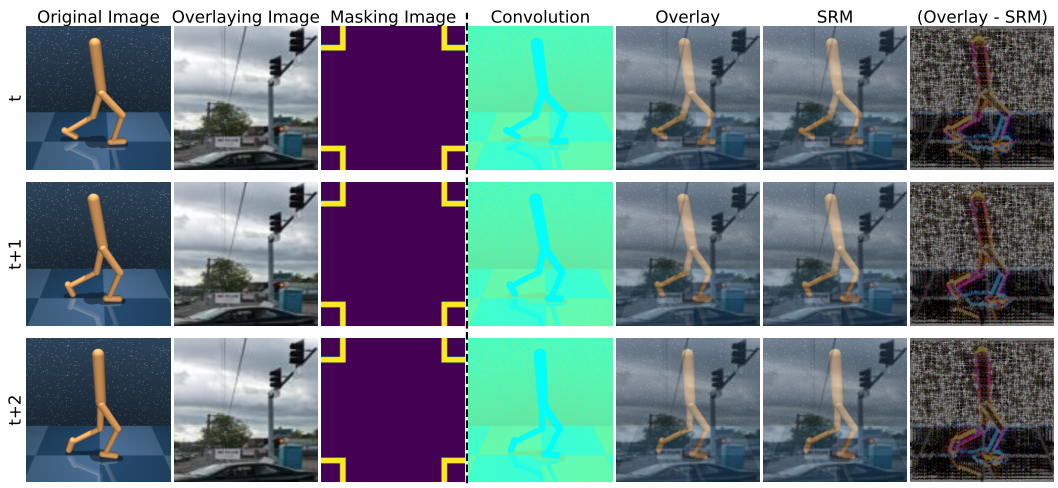


Figure 19: **Strong augmentation families.** Three augmentation types applied to *walker-walk*: *random-overlay* (contextual background replacement), *random-conv* (local texture and color perturbation), and *SRM* (Fourier-domain frequency masking). The rightmost column shows the absolute pixel-wise difference between *random-overlay* and *SRM*, illustrating their distinct spectral characteristics.

trade-off between stability and generalization coverage, with the optimal direction shifting across benchmarks but $\zeta{=}1$ providing the best compromise at every difficulty level. Third, VIGOR's robustness is augmentation-agnostic: three qualitatively distinct perturbation families (contextual, textural, and spectral) all yield strong generalization, confirming that latent-space consistency, not the augmentation, drives robustness.

G.6 Latent-Space Consistency Analysis: DC and ES Ablation

To complement the quantitative ablation in Section 4.3 (Table 3), we visualize how removing DC and ES affects the structure of latent rollouts under visual perturbations. This analysis isolates the qualitative role of each component beyond aggregate return statistics.

Experimental setup. We compare the full VIGOR with three ablated variants (w/o DC & ES, w/o DC, and w/o ES) on *walker-walk* under *background-color* and *object-color* distractions. For each variant, we roll out the learned world model from identical initial inputs and project the resulting latent states into 2D via UMAP [31]: the initial state z_t is encoded directly from the observation, while the subsequent states $z_{t+1:t+H}$ are produced by the dynamics model. To make this distinction visible, we display z_t with red-edged markers in Figure 20. Episode returns and environment snapshots are shown alongside.

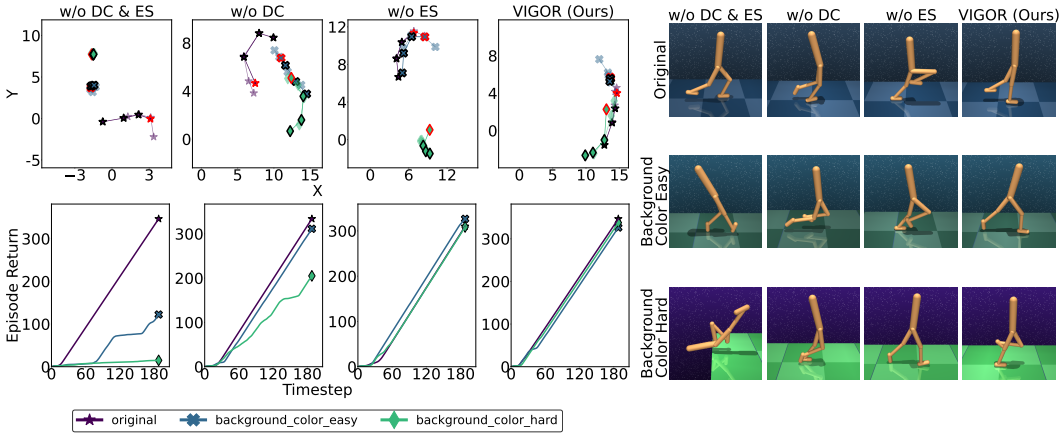
Results. Figure 20 reveals distinct failure modes for each ablation that align with the quantitative patterns in Table 3.

- **w/o DC & ES.** Without either consistency mechanism, latent trajectories scatter widely even at the encoder level: the initial states z_t under *easy* and *hard* settings are far from those under *original*, and the rollouts diverge sharply. The return curves drop substantially under perturbations, with episode failures visible in the snapshots.
- **w/o DC.** ES alone aligns the encoder representations to some extent, but the dynamics rollouts diverge across difficulty levels. Although the initial states z_t are partially clustered, the dynamics model produces inconsistent multi-step predictions, illustrating that encoder-level alignment alone cannot prevent the dynamics from amplifying residual discrepancies across recursive rollouts.
- **w/o ES.** DC alone yields more compact rollout trajectories than the previous two variants, but the latent geometry is shifted across difficulty levels: the initial states z_t under perturbations are visibly displaced from their clean counterparts. While returns are higher than the previous

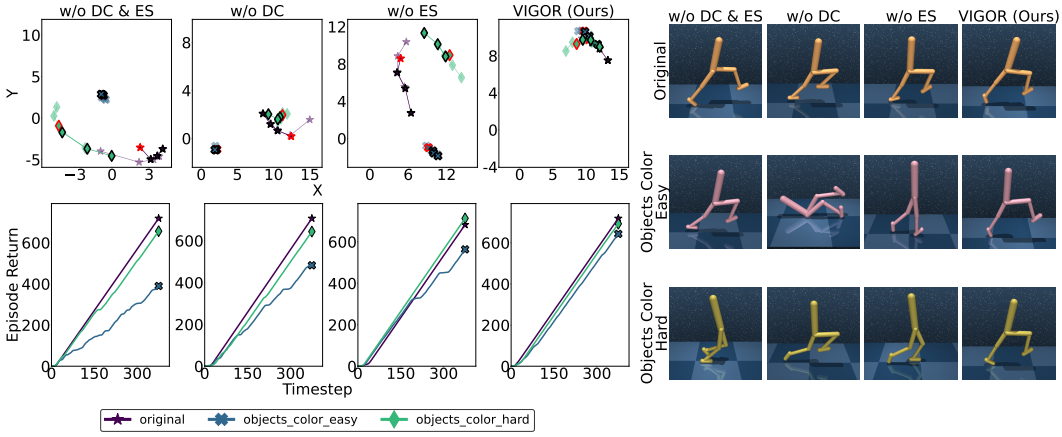
variants, the misaligned encoder representations limit the benefit of DC's dynamics-level constraint.

- **VIGOR (full).** Only the full model maintains both encoder-level alignment (initial states z_t across difficulty levels are tightly clustered) and dynamics-level consistency (subsequent rollouts $z_{t+1:t+H}$ remain aligned across the full horizon). This two-level stability translates directly into stable returns and successful task execution across all evaluated difficulty levels.

Summary. These visualizations corroborate the quantitative ablation (Table 3): ES alone leaves the rollouts divergent, DC alone leaves the initial encodings displaced, and only their combination yields the two-level stability that underlies VIGOR's robust returns.



(a) *background-color*



(b) *object-color*

Figure 20: **Latent-space consistency: ablation on DC and ES.** Left: UMAP projections of predicted latent states (top) and episode returns (bottom) for *walker-walk* under *background-color* and *object-color* distractions, comparing VIGOR with three ablated variants (w/o DC & ES, w/o DC, w/o ES). Right: Environment snapshots across difficulty levels. Markers denote difficulty (★: *original*, ×: *easy*, ◊: *hard*); arrows indicate temporal progression. Red-edged markers indicate the initial latent state z_t encoded from the observation; subsequent markers correspond to rolled-out states $z_{t+1:t+H}$ produced by the dynamics model.

G.7 Ablation Study on Design Choices

This subsection provides formal definitions of the ablated variants introduced in Section 4.4 and detailed results across distraction types, tasks, and efficiency metrics.

Dynamic vs. consistent augmentations. VIGOR applies augmentations at every step of the rollout horizon. For a trajectory segment $x_{t:t+H}$, the augmented views are:

$$x_i^w = \tau^w(x_i, \upsilon_i^w), \quad x_i^{ws} = \tau^s(\tau^w(x_i, \upsilon_i^w), \upsilon_i^s), \quad i = t, \ldots, t+H. \tag{8}$$

If $\upsilon_i^w = \upsilon_t^w$ and $\upsilon_i^s = \upsilon_t^s$ for all i, the scheme is *consistent*: a single set of augmentation parameters is sampled and reused across all rollout steps. Otherwise, the scheme is *dynamic*: new parameters are sampled independently at every step. VIGOR uses the dynamic setting by default; the consistent variant is `VIGOR_CONST_AUG`.

Choice of strong augmentation. VIGOR uses *random-overlay* by default. We compare against *random-conv*:

$$\tau^{s,\text{overlay}}(o, \tilde{o}) = (1-\delta)\, o + \delta\, \tilde{o}, \quad \tau^{s,\text{conv}}(o, w) = \text{CONV}(o, w), \tag{9}$$

where o is the input image, $\tilde{o} \sim \mathcal{D}$ is a task-irrelevant overlay, $\delta = 0.5$, and $w \sim \mathcal{N}(0, 1)$ is a random convolution kernel. Both transforms are applied to every frame in the stacked observation $x_t = \{o_t, o_{t-1}, \ldots, o_{t-k+1}\}$. We adopt *random-overlay* because it better approximates natural visual distractions at deployment, whereas *random-conv* applies synthetic pixel-level perturbations. The variant with *random-conv* is `VIGOR_CONV`. Figure 21 visualizes their qualitative differences.



Figure 21: **Strong augmentation examples.** Visualization of augmentation effects in the *walker-walk* task. Left: Original image. Center: *random-overlay* used in VIGOR. Right: *random-conv* used in `VIGOR_CONV`.

Choice of auxiliary objective for ES. VIGOR uses an alignment-based ES objective (adapted from SODA [14]) by default. As an alternative, we consider a contrastive ES objective (adapted from CURL [25]):

$$\mathcal{L}_{\text{contrast}} = -\log \frac{\exp(q^\top W k_+)}{\exp(q^\top W k_+) + \sum_{j=1}^{N} \exp(q^\top W k_j)}, \tag{10}$$

where $q = h_\theta(\tau^w(x_t, \upsilon^q))$, $k_+ = h_{\theta^-}(\tau^w(x_t, \upsilon^k))$, N is the mini-batch size (Section 3.1), and W is a learnable bilinear projection. The variant replacing alignment-based ES with contrastive ES is `VIGOR_CONTRAST`.

Zero-shot generalization. Figure 22 reports zero-shot generalization across all distraction types. VIGOR exhibits smoother and more monotonic degradation (*original* → *easy* → *hard*) compared with the ablated variants, indicating stronger robustness to distribution shifts.

Among the variants, `VIGOR_CONV` performs competitively on color-related distractions (*background-color*, *light-color*, *object-color*), consistent with the color-invariance bias of random convolution. However, its performance drops sharply on *background-video*, where perturbations alter spatial and contextual cues rather than color. `VIGOR_CONTRAST` consistently underperforms across most distraction types.

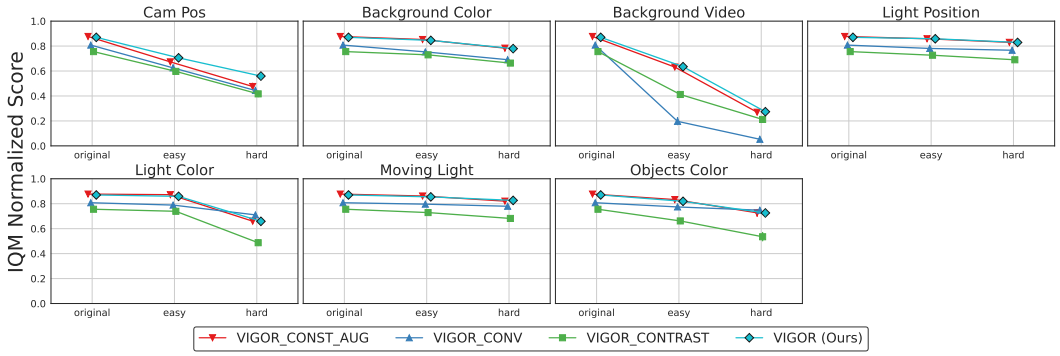


Figure 22: **Design-choice ablation: zero-shot generalization across distraction types.** Layout follows Figure 12, comparing VIGOR against its ablated variants. VIGOR shows more stable performance across difficulty levels.

Sample efficiency and aggregate metrics. Figure 23 provides a comprehensive comparison. The aggregate statistics (top row) show that VIGOR achieves the best median, IQM, mean, and optimality gap, confirming its overall advantage. The performance profile (bottom-left) shows VIGOR's dominance across the full distribution of normalized scores, and the learning curve (bottom-right) shows that VIGOR retains the sample efficiency of TD-MPC2 despite training under visual perturbations.

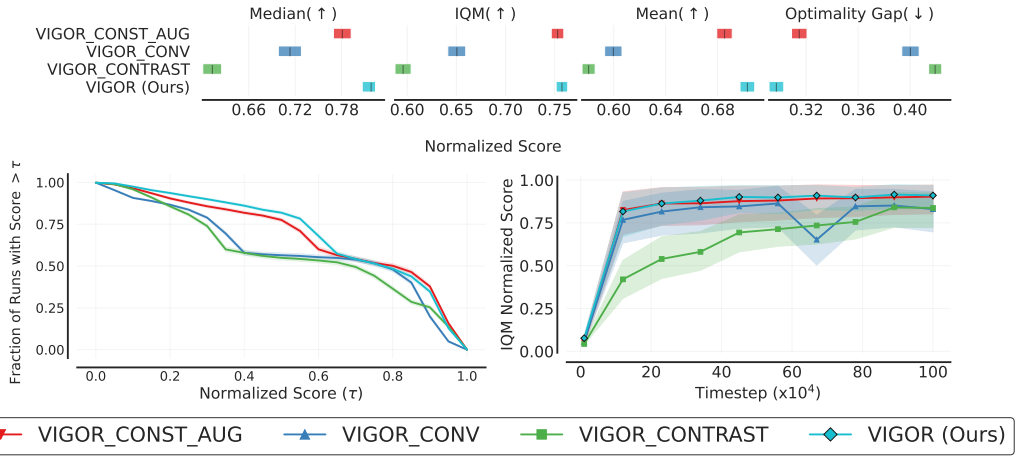


Figure 23: **Design-choice ablation: aggregate metrics and sample efficiency.** Top: Zero-shot generalization (median, IQM, mean, optimality gap). Bottom left: Performance profile. Bottom right: Learning curves. Shaded regions denote 95% stratified bootstrap confidence intervals.

Qualitative comparison of strong augmentations. Figure 21 illustrates the visual effects of the two strong augmentations on *walker-walk*. *Random-overlay* blends natural images into the background, introducing realistic domain shifts that approximate variations encountered in deployment. *Random-conv* applies convolutional filters that predominantly alter color statistics without structural changes. This distinction explains why `VIGOR_CONV` performs well on color-based perturbations but fails to generalize to shifts involving spatial or contextual variations.

Summary. These ablations confirm that each design choice (dynamic augmentation, the choice of strong augmentation, and the auxiliary loss formulation) contributes meaningfully to VIGOR's generalization. Altering any one of them leads to clear performance degradation, demonstrating that these choices collectively enable robust zero-shot generalization.

G.8 Latent-Space Consistency Analysis

This subsection provides detailed embedding visualizations that complement the latent-space analysis in Section 4.5. We examine whether the two-level consistency observed in the main text, at both the encoder and rollout levels, generalizes across multiple distraction types.

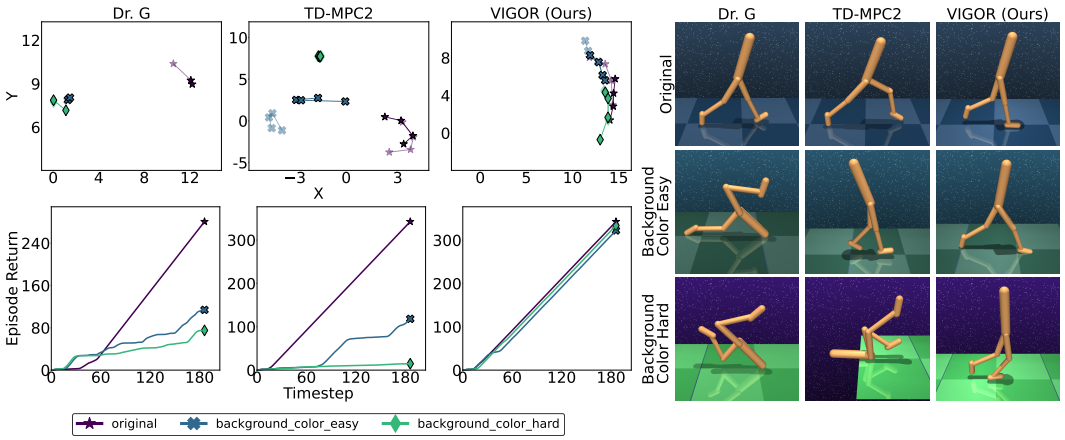
Experimental setup. We compare VIGOR with its backbone TD-MPC2 [17] and Dr. G [7], the most directly comparable prior method for zero-shot visual generalization in latent-space MBRL. All agents are trained in clean environments and evaluated *zero-shot* under unseen visual distractions. For each agent, we roll out the learned world model from identical initial inputs to obtain a predicted latent trajectory $z_{t:t+H} = (z_t, z_{t+1}, \ldots, z_{t+H})$, where each subsequent state is generated by the dynamics model as $z_{t+h+1} = d_\theta(z_{t+h}, a_{t+h})$, and we record the episode return accumulated during evaluation. To visualize how each world model evolves over consecutive timesteps, we additionally depict the preceding rollout $z_{t-1:t+H-1}$ as a lighter trajectory, providing temporal context. Comparing the two trajectories reveals whether the world model maintains consistent rollouts as the encoder processes successive observations under visual perturbations.

Results. We project the high-dimensional latent trajectories into 2D via UMAP [31], with faded markers indicating earlier time steps. Episode returns and environment snapshots across evaluation levels (*original*, *easy*, *hard*) are shown alongside.

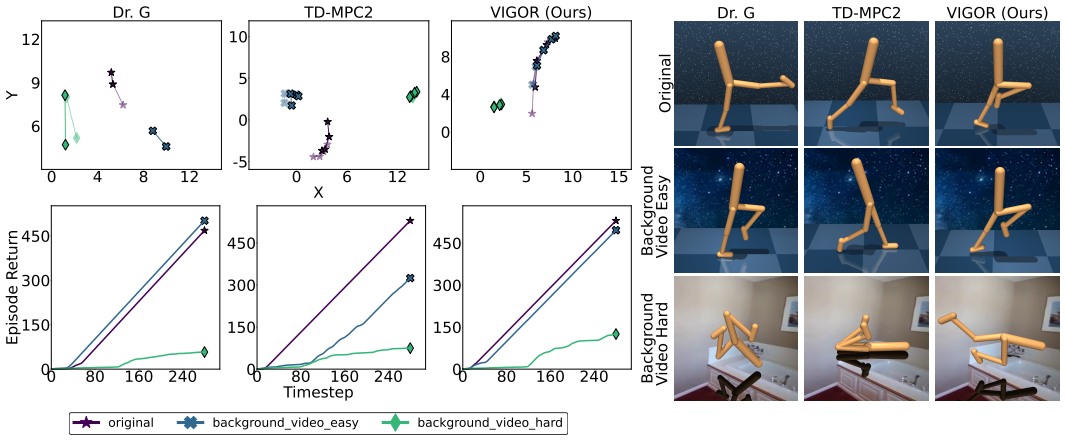
Figure 24 presents results for *walker-walk* under three perturbation types: *background-color*, *background-video*, and *light-color*. Across all three types, VIGOR exhibits consistency at both levels of the vulnerability identified in Figure 1. At the encoder level, the initial latent states extracted under *original*, *easy*, and *hard* settings are located within the same region of the latent manifold, indicating that the encoder maps visually distinct observations to similar task-relevant representations. At the rollout level, the subsequent latent trajectories $z_{t+1:t+H}$ produced by the dynamics model remain tightly aligned across difficulty levels, confirming that the dynamics model preserves this consistency throughout the planning horizon. This two-level stability translates directly into stable episode returns across all settings.

In contrast, TD-MPC2 and Dr. G exhibit divergent patterns at both levels. Their initial encoder outputs scatter under perturbations, and this divergence is amplified through latent rollouts, producing trajectories that deviate substantially from the clean setting. The resulting return degradation and task failures (visible in both the episode-return curves and the environment snapshots) illustrate the two-level vulnerability in practice.

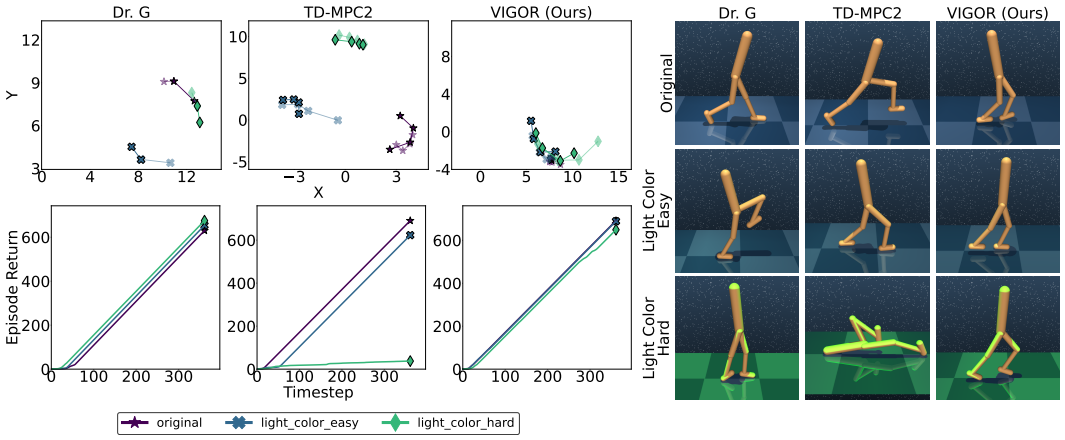
Summary. These visualizations provide further empirical support for the hypothesis (Figure 1) that mapping OOD inputs onto the task-relevant latent manifold is essential for zero-shot generalization. Across all three perturbation types, VIGOR consistently preserves consistency at both the encoder and rollout levels, a property absent in the baselines.



(a) *background-color*



(b) *background-video*



(c) *light-color*

Figure 24: **Latent-space consistency under visual perturbations.** Left: UMAP projections of predicted latent states (top) and episode returns (bottom) for *walker-walk* under three distraction types, including *background-color*, *background-video*, and *light-color*. Right: Environment snapshots across settings. Markers denote difficulty (★: *original*, ×: *easy*, ◊: *hard*); arrows indicate temporal progression. VIGOR maintains a consistent latent manifold and stable performance across levels, whereas TD-MPC2 and Dr. G exhibit divergent rollouts and degraded execution.

G.9 Performance under Longer Planning Horizons

Our default experiments adopt a short MPC planning horizon of $H=3$, following standard practice in latent-space MPC. To examine whether VIGOR's latent-consistency mechanism remains effective under deeper imagined rollouts, we conduct comparisons at extended planning horizons.

Experimental setup. Longer planning horizons substantially increase wall-clock time because each optimization step requires deeper latent rollouts. Running $H=5$ or $H=10$ across all DMC and Robosuite tasks would therefore incur prohibitive computational cost. We thus evaluate $H \in \{3, 5, 10\}$ on one representative task per benchmark: *finger-spin* on DMC and *door-opening* on Robosuite. All other hyperparameters remain fixed to isolate the effect of the planning horizon.

Results and discussion. Figure 25 shows learning curves for VIGOR under the three horizons. Across $H=3$, $H=5$, and $H=10$, VIGOR maintains stable optimization and reaches similar asymptotic returns. As expected, increasing the horizon slows convergence because compounding prediction errors accumulate over longer imagined trajectories, a well-documented limitation of MBRL. Nevertheless, VIGOR's latent-consistency regularization continues to stabilize the forward dynamics, preventing error accumulation from degrading policy learning even when the planning depth is tripled.

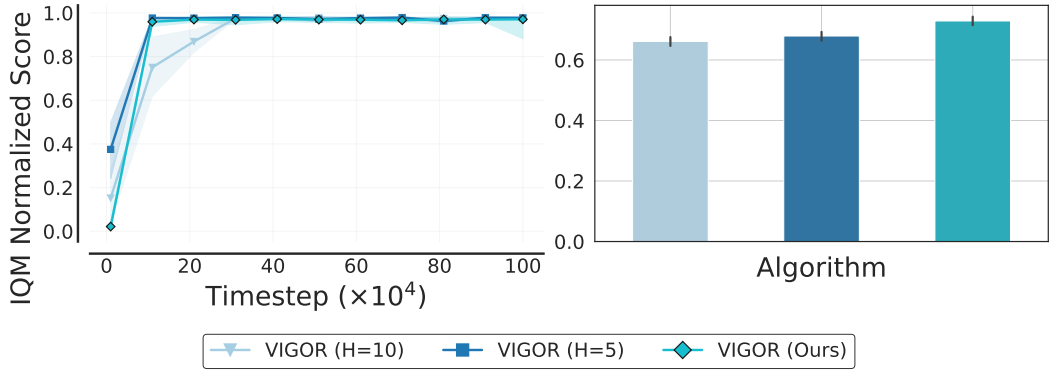


Figure 25: **Performance under extended planning horizons.** Learning curves for VIGOR on *finger-spin* (DMC) and *door-opening* (Robosuite) with $H \in \{3, 5, 10\}$. VIGOR maintains stable learning and reaches comparable asymptotic returns across all horizons, despite slower convergence at longer horizons.

Summary. These results confirm that VIGOR's latent-consistency mechanism generalizes beyond the default short-horizon setting. By mitigating the accumulation of prediction errors, VIGOR maintains stable learning and competitive performance under deeper imagined rollouts.

G.10 Generalization to Unseen Camera Viewpoints

While RL-ViGen includes camera-position shifts (translations), these preserve the base camera orientation. To investigate whether models trained with appearance-focused perturbations can generalize to fundamentally different geometric transformations, we evaluate all methods on unseen camera rotations that modify projection geometry and the relative orientation between the agent and the camera.

Viewpoint-rotation settings. We consider two rotational viewpoint distractions from DMControl-GB [14]:

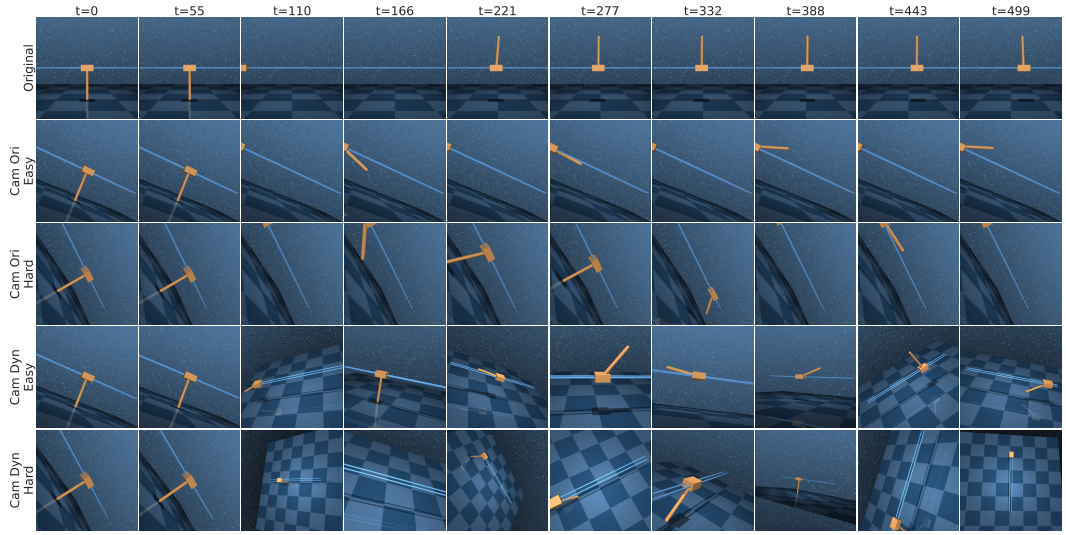
- `cam-ori` **(static rotation).** At the start of each episode, the tracking camera's spherical parameters $c = (\phi, \theta, r, \theta_{\text{roll}})$ are sampled uniformly within bounds controlled by a difficulty parameter $\beta_{\text{cam}} \in [0, 1]$ (easy: 0.3; hard: 0.8). Larger β_{cam} permits wider rotation and distance ranges ($-\frac{\pi}{2}\beta_{\text{cam}} \leqslant \phi, \theta, \theta_{\text{roll}} \leqslant \frac{\pi}{2}\beta_{\text{cam}}$ and $0.5\, r_{\text{original}}\beta_{\text{cam}} \leqslant r \leqslant 2.5\, r_{\text{original}}\beta_{\text{cam}}$), producing substantial viewpoint deviations without temporal variation.

- `cam-dyn` **(dynamic rotation).** Camera parameters are initialized as in `cam-ori` and then evolve via a bounded random walk. Camera velocities are sampled as

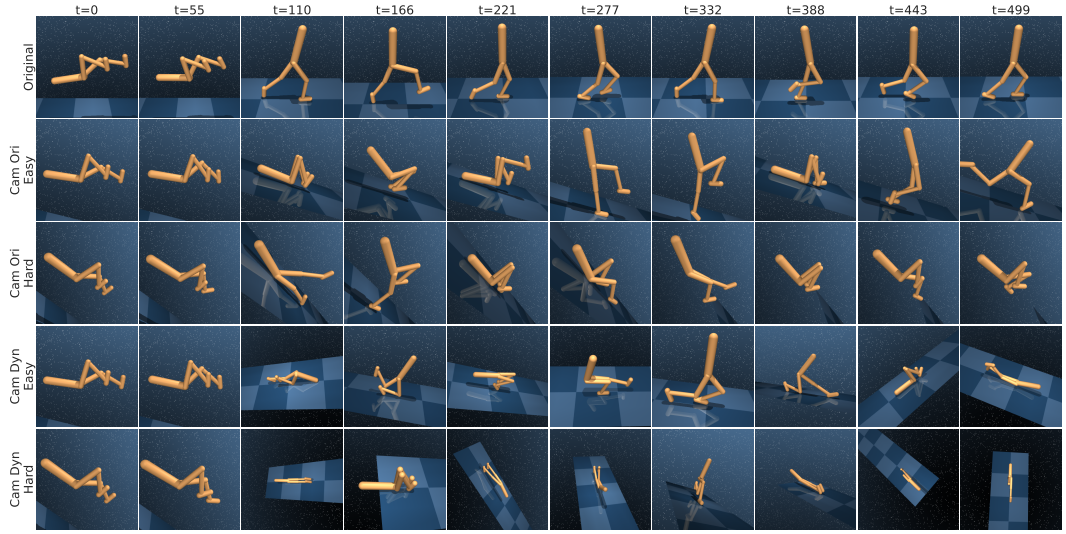
$$v_0 \sim \mathcal{U}(v_{\min}, v_{\max}), \quad v_n = v_0 + \sum_{j=1}^{n} \mathcal{N}(0, \sigma\Sigma), \tag{11}$$

where n indexes the simulation step within the episode, σ is a scalar noise scale, and Σ is the per-axis covariance, generating continuous, temporally coherent viewpoint drift throughout the episode. Motion magnitudes scale with β_{cam}, producing time-varying geometric distortions.

As illustrated in Figure 26, these viewpoint perturbations alter perspective, foreshorten limbs, reorder depth relationships, and in some cases cause the agent to leave the field of view entirely (e.g., *cartpole-swingup*).



(a) *cartpole-swingup* under `cam-ori` and `cam-dyn`.



(b) *walker-walk* under `cam-ori` and `cam-dyn`.

Figure 26: **Camera-viewpoint distractions.** Example frames from *cartpole-swingup* and *walker-walk* under `cam-ori` and `cam-dyn`. These settings introduce geometric distortions and projection changes that produce observations substantially different from training.

Zero-shot evaluation. All models are trained under the RL-ViGen protocol (*without* any viewpoint-rotation augmentations) and evaluated zero-shot on both settings.

Results and discussion. As shown in Figure 27, performance drops sharply across all methods, and none maintains reliable zero-shot control under camera rotations. This degradation arises because rotational viewpoint changes induce geometric transformations that differ fundamentally from appearance-based perturbations (e.g., lighting, background, or color). Such transformations violate the pixel-level and feature-level invariances implicitly learned by current encoder-based methods, including VIGOR.

While VIGOR is designed to improve robustness against appearance-based perturbations (background videos, lighting changes, color shifts, and camera translations), these results indicate that geometric viewpoint variations pose a more challenging generalization problem. Recent work has explored Spatial Transformer Networks (STNs) [21] for few-shot viewpoint adaptation in visual RL [43, 49], demonstrating improved spatial robustness by explicitly modeling spatial transformations. However, achieving simultaneous robustness to both appearance and geometric perturbations in a fully zero-shot setting remains largely unexplored. Developing geometry-aware latent representations, multi-view consistent world models, or scene-structured encoders constitutes an important direction for future work.

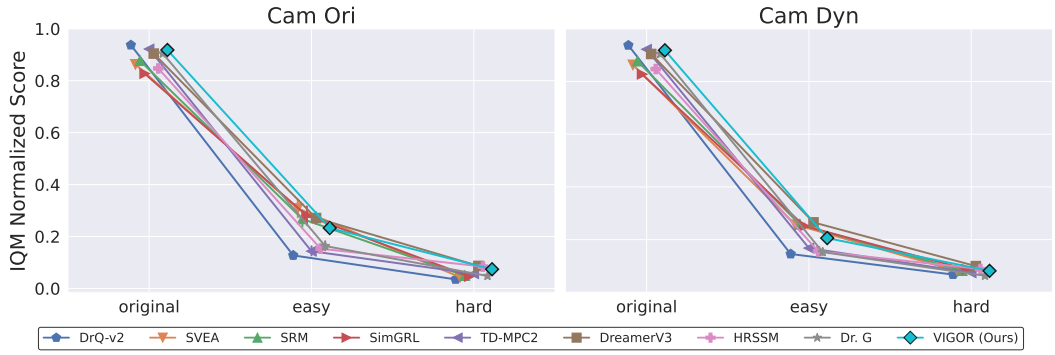


Figure 27: **Zero-shot generalization under camera-viewpoint rotations.** VIGOR and baselines evaluated on `cam-ori` and `cam-dyn` settings. All models are trained without viewpoint-rotation augmentations. Performance drops sharply for all methods, indicating that zero-shot generalization under geometric viewpoint changes remains an open challenge.

Summary. This evaluation reveals a clear limitation of current visual RL generalization methods. Although VIGOR significantly improves robustness to appearance-based perturbations, all evaluated methods struggle with the geometric projection changes induced by viewpoint rotation. These findings motivate future work on geometry-aware representations or multi-view consistent world models to achieve viewpoint-invariant control.

H MBRL Baseline Analysis

This section provides an extended discussion of the MBRL baselines used in our experiments. Since VIGOR is designed as a framework that can be integrated into existing latent-space MBRL methods, understanding the characteristics of representative backbones is important. We focus on two influential families of latent-space MBRL, **Dreamer** and **TD-MPC**, which are widely regarded as state-of-the-art for pixel-based visuomotor control.

Dreamer family. The Dreamer family [10–12] learns a *decoder-based* latent world model for long-horizon control from pixels. The model is trained on latent dynamics, reward prediction, and pixel-level reconstruction, forcing the latent state to retain sufficient information for observation recovery. The initial latent state is inferred through a recurrent state-space model (RSSM [9]) conditioned on past states, actions, and current observations. Actions are generated by a learned actor policy optimized via actor–critic RL in latent space, with the critic trained on imagined rollouts. DreamerV2 [11] and DreamerV3 [12] scale the architecture and introduce improved optimization

strategies, making the family increasingly competitive on high-dimensional pixel-based control tasks. More recently, DreamerV4 [13] extends the family with a scalable Transformer architecture for long-horizon, language-conditioned tasks; however, it is trained from purely offline interaction data and thus falls outside the online RL setting considered in this work.

TD-MPC family. The TD-MPC family [17, 18] takes a different approach by discarding reconstruction losses and adopting a *decoder-free* latent world model. It jointly learns latent dynamics, reward, value, and a policy under a temporal-difference objective, operating entirely in latent space. Unlike the Dreamer family, which trains a separate actor for long-horizon imagination, the TD-MPC family integrates its policy into an online planning scheme: at each step, a model predictive controller searches over action sequences guided by Q-value predictions, using the learned policy as a prior, and executes the first action of the best trajectory. TD-MPC2 improves upon this formulation with architectural refinements and multi-task scalability, achieving strong sample efficiency on continuous visuomotor control benchmarks.

Comparison of the two families. Both families share the core principle of learning a transition model over compact latent representations. Their fundamental difference lies in how the world model is trained and how actions are selected: the Dreamer family relies on a decoder-based model and learns an actor policy via RL, while the TD-MPC family uses a decoder-free model and selects actions through MPC planning guided by Q-values. A further distinction concerns the role of imagined rollouts at deployment: the Dreamer family generates synthetic latent trajectories only during training for actor–critic optimization, while the actor directly selects actions at test time without rollouts. The TD-MPC family, by contrast, generates latent rollouts at both training and test time: during training for world-model learning and at test time for MPC planning. This distinction reflects two design philosophies, *decoder-based actor learning* versus *decoder-free latent planning*, with implications for sample efficiency and generalization.

Rationale for backbone selection. Since VIGOR aims to enhance visual generalization without sacrificing sample efficiency, building upon an efficient backbone is essential. The Dreamer family's reconstruction-driven formulation increases model complexity and training cost, while the TD-MPC family's decoder-free design focuses directly on value learning and planning in latent space, yielding stronger sample efficiency in practice. As shown in Figure 28, TD-MPC2 outperforms both the Dreamer family and its predecessor TD-MPC across diverse tasks in DMC, consistent with prior findings [17]. We therefore adopt TD-MPC2 as VIGOR's backbone. Section 4.2 demonstrates that VIGOR preserves this efficiency advantage while substantially improving zero-shot generalization. Nevertheless, to examine whether VIGOR's benefits depend on this choice, we additionally instantiate it on DreamerV3 in Appendix I.

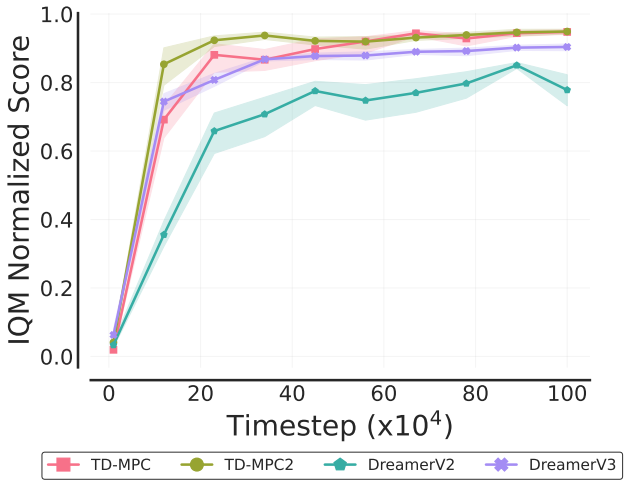


Figure 28: **Sample efficiency of MBRL baselines.** TD-MPC2 achieves the highest sample efficiency across six DMC tasks, outperforming other MBRL baselines.

I VIGOR on DreamerV3

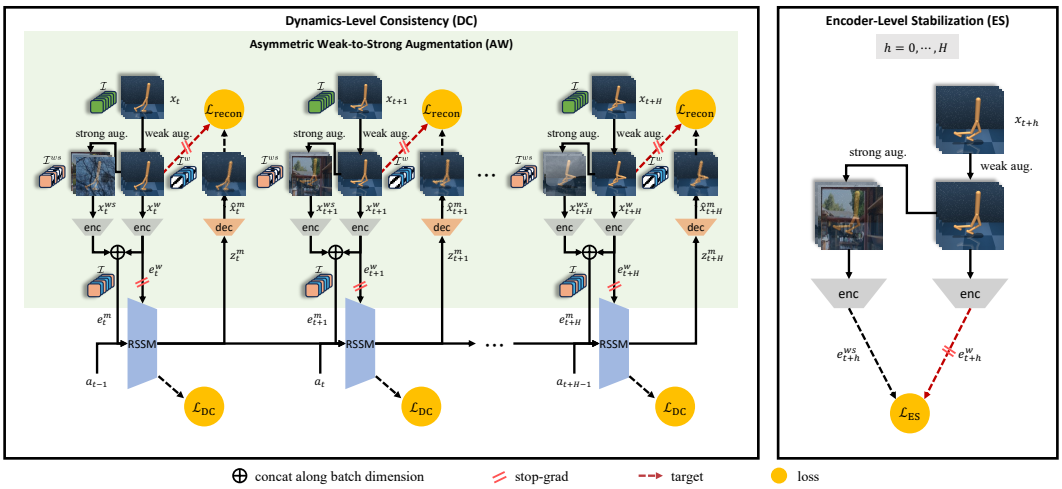
This section describes a second instantiation of VIGOR on DreamerV3 [12], denoted DreamerV3+VIGOR, which examines whether the benefits of VIGOR extend beyond the TD-MPC2 backbone. It is evaluated under the same protocol as the main experiments (Tables 9–10; Figures 30–32). Throughout this section, the default instantiation, referred to simply as VIGOR elsewhere in the paper, is denoted TD-MPC2+VIGOR. DreamerV3 provides a structurally distinct model-based RL backbone along several dimensions relevant to VIGOR. Unlike TD-MPC2 [17], which learns a decoder-free implicit latent world model and selects actions through model-predictive control, DreamerV3 learns a generative world model built around a recurrent state-space model (RSSM) with stochastic categorical latents, trains its latent dynamics through KL-balanced dynamics and representation losses, and learns an actor–critic from imagined latent trajectories. We retain DreamerV3's architecture, observation reconstruction, reward, and continuation objectives, as well as its actor–critic learning procedure; the only change to the prediction objectives is that the decoder reconstructs the weakly augmented observation from the mixed-branch state (Algorithm 2). VIGOR is incorporated only through the augmentation and consistency mechanisms described below: AW constructs the weak-only and weak-to-strong observation streams, DC re-targets the RSSM dynamics and representation KL terms across these streams, and ES adds an encoder-level consistency objective.

I.1 Overview

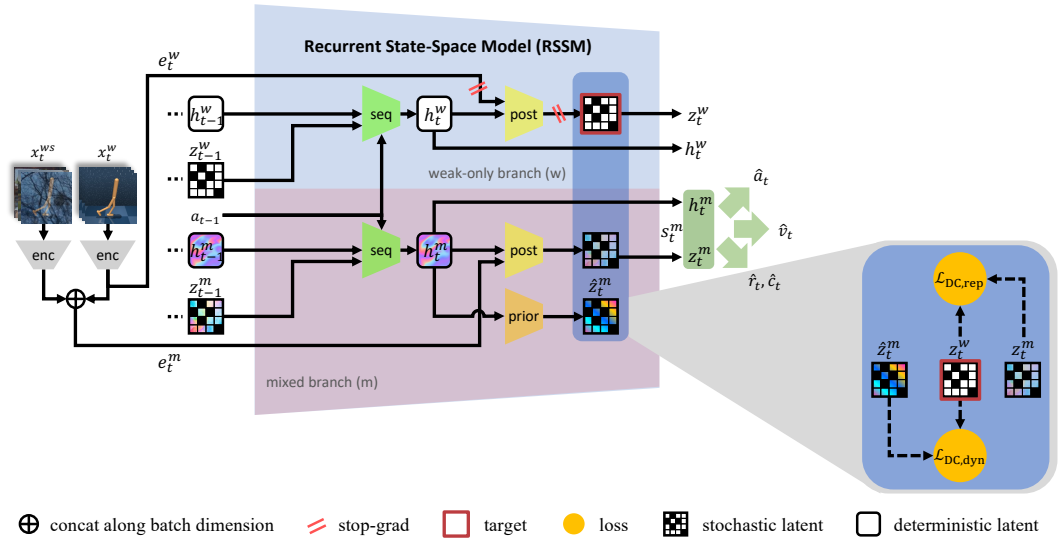
Figure 29 illustrates the integration, and Algorithm 2 summarizes world-model learning in DreamerV3+VIGOR; as in Algorithm 1, only the steps highlighted in yellow are added to (or modify) the original DreamerV3 procedure.

Algorithm 2 World model learning in DreamerV3+VIGOR

Input: Replay buffer $\mathcal{B}$; Sequence horizon H; Weak and strong augmentation functions τ^w, τ^s; Loss weights $\beta_{\text{dyn}}, \beta_{\text{rep}}, \beta_{\text{ES}}$; Learning rate η
while not converged **do**
  for gradient step $t_g = 1, 2, \ldots, T_g$ **do**
    $\Gamma = (x_i, a_{i-1}, r_i, c_i)_{i=t}^{t+H} \sim \mathcal{B}$ {Sample a sequence from the replay buffer}
    $\upsilon^w \sim \Upsilon^w, \upsilon^s \sim \Upsilon^s$ {Sample augmentation parameters per sequence}
    $L \leftarrow 0$ {Initialize cumulative loss}
    $(h^m_{t-1}, z^m_{t-1}) \leftarrow (h_{t-1}, z_{t-1})$ {Initialize from the latent stored in $\mathcal{B}$}
    $(h^w_{t-1}, z^w_{t-1}) \leftarrow (h^m_{t-1}, z^m_{t-1})$ {Start both branches from the same state}
    for $i = t, t+1, \ldots, t+H$ **do**
      $x^w_i \leftarrow \tau^w(x_i; \upsilon^w),\ x^{ws}_i \leftarrow \tau^s(x^w_i; \upsilon^s)$ {Build weak-only and weak-to-strong views}
      $e^m_i \leftarrow e_\phi(x^w_i) \oplus e_\phi(x^{ws}_i)$ {Form mixed encodings by batch-wise concat}
      $h^w_i \leftarrow f_\phi(h^w_{i-1}, z^w_{i-1}, a_{i-1})$ {Weak-only branch (stop-gradient)}
      $z^w_i \sim \text{sg}(q_\phi(\cdot \mid h^w_i, \text{sg}(e_\phi(x^w_i))))$
      $h^m_i \leftarrow f_\phi(h^m_{i-1}, z^m_{i-1}, a_{i-1})$ {Mixed branch}
      $z^m_i \sim q_\phi(\cdot \mid h^m_i, e^m_i)$
      $\mathcal{L}_{\text{pred}} \leftarrow -\ln p_\phi(x^w_i, r_i, c_i \mid h^m_i, z^m_i)$ {Calculate prediction loss}
      $\mathcal{L}_{\text{DC}} \leftarrow \beta_{\text{dyn}}\mathcal{L}_{\text{DC,dyn}} + \beta_{\text{rep}}\mathcal{L}_{\text{DC,rep}}$ {Replaces $\mathcal{L}_{\text{dyn}}, \mathcal{L}_{\text{rep}}$; Eq. (14) at step i}
      $\mathcal{L}_{\text{ES}} \leftarrow \|\bar{e}_\phi(x^{ws}_i) - \text{sg}(\bar{e}_\phi(x^w_i))\|_2^2$ {$\bar{e}_\phi = e_\phi/\|e_\phi\|_2$; e_ϕ: image encoder of q_ϕ}
      $L \leftarrow L + \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{DC}} + \beta_{\text{ES}}\mathcal{L}_{\text{ES}}$ {Accumulate loss}
    end for
    $\phi \leftarrow \phi - \eta \frac{1}{H+1} \nabla_\phi L$ {Update world model parameters}
  end for
end while



(a) Overview of DreamerV3+VIGOR



(b) RSSM of DreamerV3+VIGOR

Figure 29: **Overview of integrating VIGOR with DreamerV3.** (a) At every step of a sampled training sequence (steps $t, \dots, t+H$ are shown), AW constructs the weak-only and weak-to-strong views of the observation (subsets $\mathcal{I}^w$ and $\mathcal{I}^{ws}$, as in Figure 2) and concatenates them along the batch dimension; the shared encoder produces the mixed embedding e_t^m and, under stop-gradient, the weak-only embedding e_t^w, both of which enter the RSSM together with the previous action a_{t-1}. The decoder reconstructs the weakly augmented observation from the mixed-branch state s_t^m ($\mathcal{L}_{\text{recon}} = -\ln p_\phi(x_t^w \mid h_t^m, z_t^m)$, the image term of $\mathcal{L}_{\text{pred}}$ in Algorithm 2), DC is applied inside the RSSM at every step ($\mathcal{L}_{\text{DC}}$; detailed in (b)), and ES aligns the weak-to-strong embedding e_{t+h}^{ws} with its weak-only counterpart e_{t+h}^{w} at every step $h = 0, \dots, H$. (b) One time step of the RSSM, with the weak-only branch (top, stop-gradient) and the mixed branch (bottom). Each branch updates its recurrent state from its previous state and the action, $h_t = f_\phi(h_{t-1}, z_{t-1}, a_{t-1})$ (seq), and infers its posterior latent z_t from h_t and the embedding e_t (post); the mixed branch additionally predicts the prior latent $\hat{z}_t^m$ from h_t^m alone (prior). The weak-only posterior z_t^w serves as the reference toward which DC trains the mixed-branch prior $\hat{z}_t^m$ ($\mathcal{L}_{\text{DC,dyn}}$) and regularizes the mixed-branch posterior z_t^m ($\mathcal{L}_{\text{DC,rep}}$). The full mixed-branch state $s_t^m = (h_t^m, z_t^m)$ is passed to the decoder in (a) and to the reward, continuation, actor, and critic heads ($\hat{r}_t, \hat{c}_t, \hat{a}_t, \hat{v}_t$).

I.2 Notation

Following the DreamerV3 notation, x_t denotes the visual observation (here a single frame o_t, since DreamerV3 uses no frame stacking, whereas x_t denotes the frame stack $\{o_t, o_{t-1}, o_{t-2}\}$ in Section 3; see Table 4), h_t denotes the deterministic recurrent state of the RSSM, and z_t denotes its stochastic

categorical latent; note that z_t here is the stochastic state of the RSSM, whereas in Section 3 it denotes the encoder output of TD-MPC2. We write $e_t = e_\phi(x_t)$ for the encoder embedding supplied to the posterior, with e_t^w, e_t^{ws}, and e_t^m denoting the embeddings of the respective views (Figure 29(a)); ES operates on its ℓ_2-normalized version $\bar{e}_\phi(x_t) = e_\phi(x_t)/\|e_\phi(x_t)\|_2$ (Algorithm 2). The posterior $q_\phi(z_t \mid h_t, e_t)$ infers the stochastic latent from the current observation together with the recurrent state, whereas the prior $p_\phi(z_t \mid h_t)$ predicts the latent from the recurrent state without access to the current observation. Throughout this section, h_t denotes the RSSM recurrent state and should not be confused with the visual encoder h_θ introduced in Section 3; likewise, $s_t = (h_t, z_t)$ in Figure 29(b) denotes the full RSSM state. Figure 29(a) illustrates consecutive steps $t, \ldots, t+H$ of a sampled training sequence ($H+1$ steps), which Algorithm 2 indexes as $i = t, \ldots, t+H$; as in Algorithm 1, t is the first step of the sampled sequence and i the running index. As in Section 3, superscripts w, ws, and m denote quantities computed from the weak-only sequence, the weak-to-strong sequence, and the mixed (weak-only $\oplus$ weak-to-strong) batch, respectively.

I.3 Where the Two-Level Vulnerability Appears in DreamerV3

The recursive pathway differs from that of TD-MPC2. At deployment, DreamerV3 does not perform online planning; instead, the actor selects actions from the current RSSM state (h_t, z_t), where z_t is inferred from the current observation by the posterior. A visual perturbation can therefore alter the posterior state used for the current action and, because z_t contributes to the recurrent update, can also influence subsequent deterministic states and latent predictions. Unlike in TD-MPC2, where representation errors can compound along a finite-horizon planning rollout, this second level of sensitivity in DreamerV3 arises through temporal recurrence of the RSSM across environment steps. During training, the actor–critic is optimized on imagined trajectories generated by the prior dynamics, so robustness of both posterior inference and prior prediction is important.

The DC objective introduced below (Appendix I.5) targets both pathways: its representation term constrains posterior inference from perturbed observations, and its dynamics term constrains the prior dynamics propagated from them, with both anchored to the weak-only posterior at every time step of the sequence.

I.4 Background: KL Balancing in DreamerV3

DreamerV3 already regularizes consistency between its posterior and prior latent distributions through two KL terms that share the same posterior–prior pair but place the stop-gradient on opposite sides:

$$\mathcal{L}_{\mathrm{dyn}} = D_{\mathrm{KL}}\big[\,\mathrm{sg}(q_\phi(z_t \mid h_t, e_t)) \,\big\|\, p_\phi(z_t \mid h_t)\big]\,, \tag{12}$$

$$\mathcal{L}_{\mathrm{rep}} = D_{\mathrm{KL}}\big[\,q_\phi(z_t \mid h_t, e_t) \,\big\|\, \mathrm{sg}(p_\phi(z_t \mid h_t))\big]\,. \tag{13}$$

The dynamics loss trains the prior to predict the stop-gradient posterior, whereas the representation loss regularizes the posterior toward the stop-gradient prior, making the inferred representations easier for the dynamics model to predict. DreamerV3 weights these terms by $\beta_{\mathrm{dyn}} = 1$ and $\beta_{\mathrm{rep}} = 0.1$ and applies a 1-nat free-bits threshold to both KL divergences. This asymmetric posterior–prior optimization follows the KL-balancing principle introduced in DreamerV2 [11], while DreamerV3 combines it with free bits to prevent the KL objective from forcing the representations toward an overly predictable but uninformative solution.

These KL terms differ from the DC objective of Eq. (4) in two important respects. First, they operate on stochastic categorical latent distributions rather than deterministic latent vectors. Second, they impose consistency between the posterior and prior computed within the same observation stream; they do not directly constrain corresponding latent distributions obtained from different augmented views to agree. Consequently, the standard DreamerV3 KL objective provides no explicit cross-augmentation invariance signal, motivating the cross-view re-targeting introduced by VIGOR's DC objective.

I.5 Integration

The three components are integrated as follows.

AW. AW follows the construction in Section 3.1. Each sampled training sequence is transformed according to the weak-only branch (τ^w) or the weak-to-strong branch ($\tau^s \circ \tau^w$) with $\zeta = 1$, and

the two views are encoded by the shared CNN encoder; their embeddings are concatenated along the batch dimension into the mixed embedding e^m (Figure 29(a)), which is then processed by the RSSM. Because the DreamerV3 posterior is conditioned on the encoded observation at every time step, the augmented observation stream directly influences posterior inference throughout the training sequence. This differs from the TD-MPC2 instantiation of Eq. (4), where the mixed representation is used only to construct the initial latent state of the rollout segment.

DC. We instantiate dynamics-level consistency in DreamerV3 by re-targeting its two KL-balancing terms across the weak-only and mixed branches. Specifically, the dynamics loss trains the prior predicted from the mixed sequence to match the stop-gradient posterior from the weak-only sequence, while the representation loss regularizes the mixed posterior toward the same stop-gradient posterior from the weak-only sequence:

$$\mathcal{L}_{\mathrm{DC,dyn}} = D_{\mathrm{KL}}\big[\,\mathrm{sg}(q_\phi(z_t^w \mid h_t^w, e_t^w)) \,\big\|\, p_\phi(\hat{z}_t^m \mid h_t^m)\big]\,, \tag{14}$$

$$\mathcal{L}_{\mathrm{DC,rep}} = D_{\mathrm{KL}}\big[\,q_\phi(z_t^m \mid h_t^m, e_t^m) \,\big\|\, \mathrm{sg}(q_\phi(z_t^w \mid h_t^w, e_t^w))\big]\,. \tag{15}$$

Here, the weak-only and mixed sequences are processed by the same RSSM, producing their respective posterior and prior distributions at each time step. Stop-gradient is applied only to the reference distribution in each KL term, preserving the directional optimization of DreamerV3's original KL balancing. We replace the corresponding standard DreamerV3 dynamics and representation KL terms with Eqs. (14)–(15), while retaining the original loss weights, $\beta_{\mathrm{dyn}} = 1$ and $\beta_{\mathrm{rep}} = 0.1$, as well as the 1-nat free-bits threshold.

This construction transfers the consistency principle of Eq. (4) to DreamerV3 while respecting the structure of its stochastic recurrent state-space model. Rather than regressing deterministic latent vectors with an ℓ_2 distance, DreamerV3+VIGOR aligns categorical latent distributions through KL divergence. Moreover, because the RSSM operates recurrently over sequence batches, the cross-view alignment is imposed at every time step, with the weak-only posterior serving as a shared stop-gradient reference for both KL terms; this encourages the latent dynamics and representations induced by perturbed observations to remain consistent with those induced by the weakly augmented sequence. In contrast to the original DreamerV3 representation loss, the representation term of the DC objective therefore no longer regularizes the posterior toward the prior. As in the TD-MPC2 instantiation, we use the weakly augmented view, rather than an unaugmented clean view, as the reference branch for the reasons discussed in Section 3.2.

ES. We apply the encoder-level stabilization objective of Eq. (5) to the observation embedding e_t supplied to the RSSM posterior. At each time step, the encoder embedding obtained from the weak-to-strong view (e_t^{ws}) is aligned with the stop-gradient embedding obtained from the corresponding weak-only view (e_t^w). This imposes augmentation invariance before the observation enters the recurrent state-space model, complementing DC, which enforces consistency at the level of the stochastic latent dynamics. By reducing augmentation-induced variation in the encoder output, ES provides the posterior with a more stable observation representation across the two views, including at the beginning of a sequence where recurrent context is limited.

I.6 Experimental Setup

DreamerV3+VIGOR is trained and evaluated under the protocol of the main experiments (Section 4.1; Appendix F): agents are trained in clean environments on the six DMC and four Robosuite tasks and evaluated zero-shot under the seven RL-ViGen distraction types at every difficulty level, with the same metrics as in Section 4.2. The DreamerV3 baseline of the main experiments serves as the backbone reference, so the comparison between DreamerV3 and DreamerV3+VIGOR isolates the effect of VIGOR. DreamerV3+VIGOR uses the same architecture, optimization settings, sequence length, batch size, and world-model loss scales as the DreamerV3 baseline (DreamerV3 column of Table 4; official defaults otherwise), including the KL loss weights and free-bits threshold stated above. No backbone-specific hyperparameters are changed. The VIGOR-specific hyperparameters (the weak and strong augmentations τ^w and τ^s and the mixture ratio ζ) are shared with the TD-MPC2 instantiation and correspond to the highlighted rows of Table 5. The ES loss weight is set to $\beta_{\mathrm{ES}} = 1$. All results are averaged over ten random seeds.

I.7 Results and Discussion

Table 9 reports per-task zero-shot generalization for DreamerV3+VIGOR alongside all baselines and TD-MPC2+VIGOR, and Figure 30 breaks the results down by distraction type and difficulty level. Table 10 and Figure 31 isolate the two backbones and their VIGOR instantiations, and Figure 32 reports the corresponding learning curves.

Zero-shot generalization. DreamerV3+VIGOR improves the average zero-shot return of its backbone from 635.3 to 767.8 on DMC (+20.9%) and from 64.3 to 98.1 on Robosuite (+52.6%). On DMC, the improvement holds on every task, and DreamerV3+VIGOR attains the best score among all methods on five of the six tasks; its average is second only to TD-MPC2+VIGOR (777.6) and exceeds every baseline, including the strongest baseline, SRM (752.0). The exception is *finger-spin* (458.3 ± 441.4), where the DreamerV3 backbone itself performs poorly (268.2) and the large variance indicates seed-dependent training. On Robosuite, the gains are concentrated on *door-opening* (35.0 → 175.2) and *nut-assembly* (3.4 → 12.6), whereas *lifting* decreases (23.7 → 12.6) and *peg-in-hole* is essentially unchanged (195.2 → 192.0); the average again exceeds every baseline (SRM: 91.8) but remains well below TD-MPC2+VIGOR (131.8).

Figures 30 and 31 show that the gains arise where the backbone is most fragile. On DMC, DreamerV3 degrades sharply under hard *cam-pos*, *background-color*, *background-video*, and *light-color* distractions, whereas DreamerV3+VIGOR degrades gracefully; on *background-video* hard, it even retains a higher IQM normalized score than TD-MPC2+VIGOR. On Robosuite, DreamerV3+VIGOR scores considerably lower than TD-MPC2+VIGOR in the *original* setting but maintains a nearly flat profile from *easy* to *extreme* and attains the highest IQM normalized score among all methods under the *hard* and *extreme* settings.

Table 9: **Zero-shot generalization scores including DreamerV3+VIGOR.** Mean episode returns ($\pm$ std) averaged over all distraction types and difficulty levels for each task. Baseline columns are identical to Table 6. Boldface denotes the best score per task.

Env	Task	MFRL				MBRL				VIGOR (Ours)	
		DrQ-v2	SVEA	SRM	SimGRL	TD-MPC2	DreamerV3	HRSSM	Dr. G	TD-MPC2+VIGOR (default)	DreamerV3+VIGOR
DMC	walker-walk	731.5 ± 344.4	738.5 ± 177.0	730.2 ± 153.6	719.3 ± 128.5	642.2 ± 344.8	795.5 ± 279.0	775.3 ± 153.0	769.8 ± 331.9	811.3 ± 201.7	$\mathbf{927.7 \pm 144.7}$
	finger-spin	721.6 ± 373.0	739.8 ± 246.4	801.5 ± 319.8	787.9 ± 242.5	643.9 ± 336.7	268.2 ± 301.6	734.6 ± 319.1	584.2 ± 321.9	$\mathbf{809.1 \pm 212.2}$	458.3 ± 441.4
	cheetah-run	379.3 ± 295.5	429.1 ± 179.1	443.7 ± 228.9	206.1 ± 25.6	318.4 ± 243.4	364.3 ± 281.2	323.8 ± 85.7	413.7 ± 286.7	461.6 ± 176.7	$\mathbf{490.9 \pm 167.2}$
	walker-stand	754.3 ± 313.7	933.5 ± 132.0	932.7 ± 105.0	929.5 ± 110.3	737.1 ± 320.5	891.9 ± 233.4	915.7 ± 116.0	834.6 ± 274.9	938.6 ± 72.9	$\mathbf{979.7 \pm 81.8}$
	reacher-easy	501.8 ± 449.9	829.8 ± 322.1	858.2 ± 291.8	875.5 ± 286.8	621.9 ± 422.5	758.5 ± 345.7	571.2 ± 427.5	500.9 ± 388.8	899.0 ± 232.3	$\mathbf{925.2 \pm 187.5}$
	cartpole-swingup	668.3 ± 289.2	686.5 ± 281.8	745.7 ± 143.4	742.7 ± 87.5	631.1 ± 241.7	733.5 ± 212.5	657.8 ± 192.1	638.7 ± 274.7	746.0 ± 173.8	$\mathbf{825.2 \pm 138.3}$
	Average	626.1 ± 375.1	726.2 ± 279.4	752.0 ± 269.8	710.2 ± 292.9	599.1 ± 349.8	635.3 ± 363.2	663.1 ± 308.4	623.7 ± 347.5	$\mathbf{777.6 \pm 241.5}$	767.8 ± 309.7
Robosuite	door-opening	7.6 ± 32.2	76.1 ± 98.1	92.5 ± 90.9	1.0 ± 0.4	1.0 ± 1.0	35.0 ± 41.5	6.8 ± 12.7	13.2 ± 69.0	143.9 ± 201.6	$\mathbf{175.2 \pm 77.7}$
	lifting	27.0 ± 25.1	20.6 ± 58.9	25.5 ± 78.7	21.6 ± 74.1	1.8 ± 5.0	23.7 ± 27.7	1.5 ± 2.7	1.3 ± 2.5	$\mathbf{35.4 \pm 74.9}$	12.6 ± 6.1
	nut-assembly	0.6 ± 1.6	11.8 ± 31.7	10.2 ± 26.8	1.3 ± 0.0	0.7 ± 1.1	3.4 ± 21.7	5.2 ± 3.8	0.7 ± 1.0	$\mathbf{15.5 \pm 39.3}$	12.6 ± 15.1
	peg-in-hole	128.8 ± 45.0	222.6 ± 86.6	239.1 ± 114.2	137.3 ± 12.9	157.4 ± 35.9	195.2 ± 39.1	166.9 ± 24.1	200.8 ± 28.7	$\mathbf{332.6 \pm 114.2}$	192.0 ± 24.8
	Average	41.0 ± 59.9	82.8 ± 111.9	91.8 ± 123.4	40.3 ± 68.0	40.2 ± 70.1	64.3 ± 83.5	45.1 ± 71.7	54.0 ± 92.8	$\mathbf{131.8 \pm 176.1}$	98.1 ± 95.3

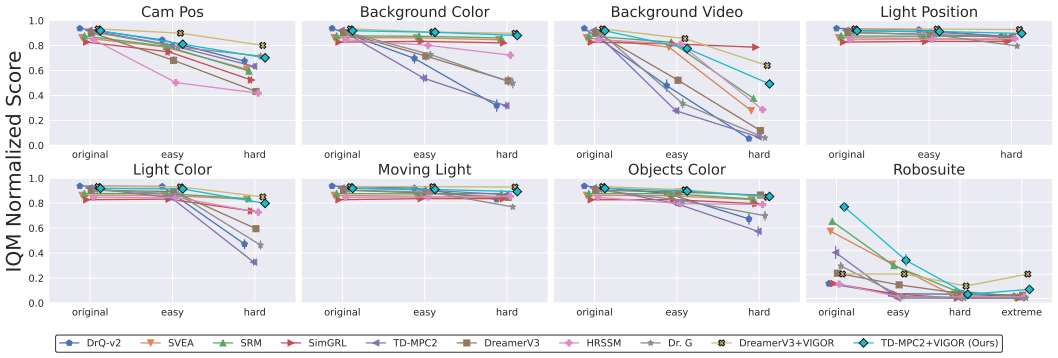


Figure 30: **Zero-shot generalization across distraction types and difficulty levels, including DreamerV3+VIGOR.** Layout follows Figure 12: each subplot reports IQM normalized scores for a single distraction type across difficulty levels. DMC: seven distraction types × three settings (*original*, *easy*, *hard*). Robosuite: four settings (*original*, *easy*, *hard*, *extreme*).

Table 10: **Zero-shot generalization scores for DreamerV3 and TD-MPC2 backbones.** Mean episode returns ($\pm$ std) averaged over all distraction types and difficulty levels for each task. Boldface denotes the best score per task.

Env	Task	DreamerV3	DreamerV3+VIGOR	TD-MPC2	TD-MPC2+VIGOR (Ours)
DMC	walker-walk	795.5 ± 279.0	$\mathbf{927.7 \pm 144.7}$	642.2 ± 344.8	811.3 ± 201.7
	finger-spin	268.2 ± 301.6	458.3 ± 441.4	643.9 ± 336.7	$\mathbf{809.1 \pm 212.2}$
	cheetah-run	364.3 ± 281.2	$\mathbf{490.9 \pm 167.2}$	318.4 ± 243.4	461.6 ± 176.7
	walker-stand	891.9 ± 233.4	$\mathbf{979.7 \pm 81.8}$	737.1 ± 320.5	938.6 ± 72.9
	reacher-easy	758.5 ± 345.7	$\mathbf{925.2 \pm 187.5}$	621.9 ± 422.5	899.0 ± 232.3
	cartpole-swingup	733.5 ± 212.5	$\mathbf{825.2 \pm 138.3}$	631.1 ± 241.7	746.0 ± 173.8
	Average	635.3 ± 363.2	767.8 ± 309.7	599.1 ± 349.8	$\mathbf{777.6 \pm 241.5}$
Robosuite	door-opening	35.0 ± 41.5	$\mathbf{175.2 \pm 77.7}$	1.0 ± 1.0	143.9 ± 201.6
	lifting	23.7 ± 27.7	12.6 ± 6.1	1.8 ± 5.0	$\mathbf{35.4 \pm 74.9}$
	nut-assembly	3.4 ± 21.7	12.6 ± 15.1	0.7 ± 1.1	$\mathbf{15.5 \pm 39.3}$
	peg-in-hole	195.2 ± 39.1	192.0 ± 24.8	157.4 ± 35.9	$\mathbf{332.6 \pm 114.2}$
	Average	64.3 ± 83.5	98.1 ± 95.3	40.2 ± 70.1	$\mathbf{131.8 \pm 176.1}$

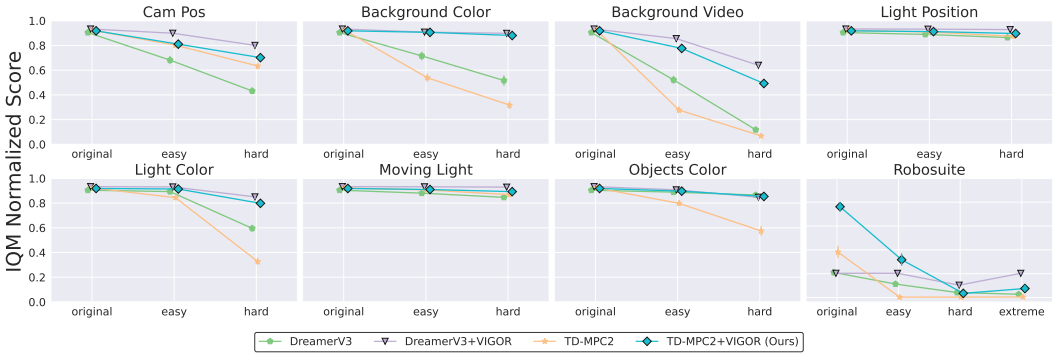


Figure 31: **Zero-shot generalization across distraction types and difficulty levels for DreamerV3 and TD-MPC2 backbones.** Each subplot reports IQM normalized scores for a single distraction type across difficulty levels. DMC: seven distraction types $\times$ three settings (*original*, *easy*, *hard*). Robosuite: four settings (*original*, *easy*, *hard*, *extreme*).

Sample efficiency. Figure 32 shows that DreamerV3+VIGOR tracks the learning curves of DreamerV3 on most tasks and learns faster on *peg-in-hole*, but converges to noticeably lower training returns on *cheetah-run* and *door-opening*. The added augmentation and consistency objectives thus incur a task-dependent cost in training return on the DreamerV3 backbone, in contrast to the near-lossless integration observed on TD-MPC2 (Section 4.2; Appendix G.2).

I.8 Summary

The DreamerV3 instantiation demonstrates that VIGOR can be applied to a structurally distinct RSSM-based world model while preserving the latent-space consistency principle underlying VIGOR. AW retains the same weak-only and weak-to-strong construction, DC re-targets DreamerV3's KL-balancing terms toward the weak-only posterior across the two augmentation branches, and ES imposes consistency at the encoder level. Without backbone-specific tuning, DreamerV3+VIGOR improves the aggregate zero-shot generalization of DreamerV3 on both benchmarks and exceeds every baseline in mean return, although it remains below TD-MPC2+VIGOR, does not improve every Robosuite task, and incurs lower training returns on some tasks. The two instantiations thus differ in absolute performance and in their efficiency–robustness profiles, reflecting differences between the underlying backbones. Taken together, the results suggest that the benefits of VIGOR are not limited to the TD-MPC2 backbone, while the TD-MPC2 instantiation remains the stronger of the two.

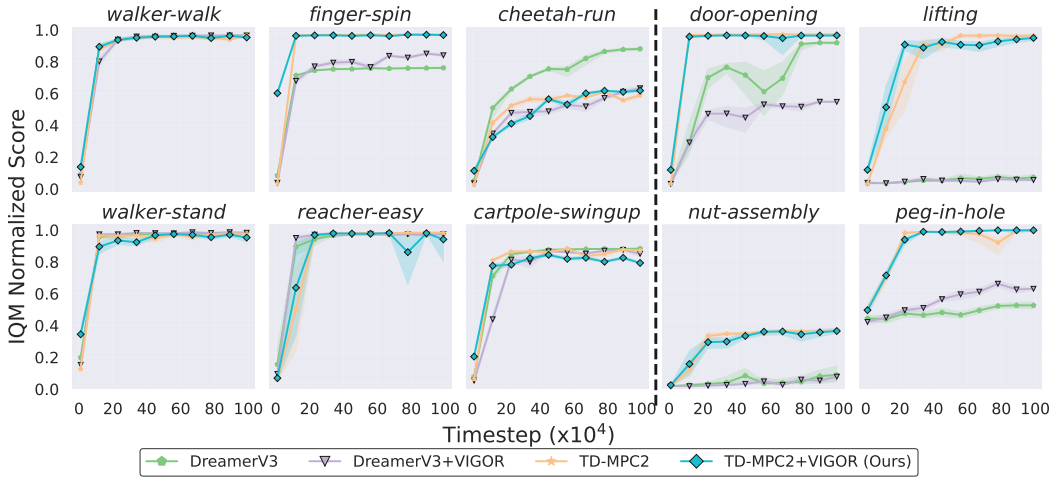


Figure 32: **Per-task sample efficiency for DreamerV3 and TD-MPC2 backbones.** Learning curves reported as IQM normalized scores across methods. Shaded regions denote 95% stratified bootstrap confidence intervals.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The introduction states three contributions: the two-level vulnerability analysis (Section 1; Figure 1), the VIGOR framework with three interdependent components (AW, DC, and ES) (Section 3), and state-of-the-art zero-shot generalization with augmentation-agnostic robustness. All claims are supported by experiments in Section 4.2 and ablations in Sections 4.3 and 4.4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 acknowledges that VIGOR targets appearance-based perturbations, and Appendix G.10 explicitly evaluates and discusses VIGOR's limitations under geometric viewpoint shifts.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not include theoretical results; VIGOR is an empirical contribution.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3 describes the VIGOR framework, Algorithm 1 specifies the training procedure, Appendix E reports all hyperparameters and the code base used for each baseline, Appendix F details the environments, distraction categories, and evaluation protocol, and Appendix I specifies the DreamerV3 instantiation (Algorithm 2).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The source code is publicly available via our project page (`https://vigor-gen.github.io/`), which links to the GitHub repository, and includes training scripts, hyperparameter configurations, and evaluation protocols to reproduce the reported experiments for both the TD-MPC2 and DreamerV3 instantiations; all environments and datasets used (DMC and Robosuite under the RL-ViGen protocol, and the Places dataset for augmentation) are publicly available.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 and Appendix F describe the experimental setting, including environments, baselines, evaluation metrics, and evaluation protocol. Appendix E provides comprehensive hyperparameter tables for VIGOR and all baselines.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All results are averaged over ten random seeds. Table 1 reports 95% stratified bootstrap confidence intervals, learning curves show 95% bootstrap confidence intervals as shaded regions, and ablation tables report mean $\pm$ standard deviation.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix E reports the compute resources: all experiments were conducted on a single NVIDIA A5000 GPU, with average training times of approximately 48 hours on DMC tasks and 72 hours on Robosuite tasks per run. Appendix G.4 provides a detailed wall-clock training efficiency analysis.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research has been conducted in accordance with the NeurIPS Code of Ethics. We use publicly available simulation benchmarks (DMC, Robosuite) and the Places dataset [52] for augmentation, and the work does not involve human subjects, sensitive data, or content that raises ethical concerns.

   Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Appendix A discusses positive impacts (improving robustness of visual RL agents to distribution shifts) and acknowledges that real-world deployment in safety-critical robotics applications requires additional validation and safeguards.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: VIGOR is a training framework evaluated on standard simulation benchmarks (DMC, Robosuite) and does not release pre-trained models, scraped datasets, or other assets with high risk for misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All baselines, benchmarks (DMC, Robosuite), and external assets (e.g., Places dataset) are properly cited. Appendix E provides URLs to the official implementations used. All assets are used in accordance with their licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The source code released alongside the paper is documented in the public GitHub repository linked from our project page (`https://vigor-gen.github.io/`) and is released under the MIT license. The repository includes a README with setup and reproduction instructions, hyperparameter configurations for the reported experiments, training and evaluation scripts for both the TD-MPC2 and DreamerV3 instantiations, and details on dependencies, hardware requirements, and expected runtime. No new datasets or pre-trained models are introduced.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs were used only for writing refinement (clarity, conciseness, and grammar) and were not involved in the core methodology, experimental design, or analysis.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.