香港城市大學
City University of Hong Kong

Agent-Centric Observation Adaptation for Robust Visual Control under Dynamic Perturbations

Zhengru Fang[1,2,*], **Yu Guo**[1,*], **Fei Liu**[1], **Yuang Zhang**[2], **Yihang Tao**[1], **Senkang Hu**[1], **Wenbo Ding**[2], **Yuguang Fang**[1]

[1] City University of Hong Kong, [2] Tsinghua University

arXiv:2604.24661v3 [cs.RO] 8 May 2026

Real-world visual systems face time-varying perturbations, including weather, sensor noise, compression artifacts, and background distractions. Existing image restoration methods are typically designed for fixed corruption types and optimized for pixel-level fidelity, leaving open two questions: how restoration behaves under non-stationary corruption switching, and whether pixel-level fidelity preserves the task-relevant information needed by downstream models. To study this setting, we introduce the Visual Degraded Control Suite (VDCS), a benchmark that injects Markov-switching physical degradations into rendered scenes. We further identify a fundamental failure mode of reconstruction-based representations: faithfully reconstructing corrupted observations forces the latent state to encode corruption-specific nuisance information, thereby contaminating downstream models. From an information-bottleneck perspective, anchoring the representation to the clean foreground eliminates this contamination. Motivated by this analysis, we propose *Agent-Centric Observations with Mixture-of-Experts* (ACO-MoE), a frozen, plug-and-play observation adapter that combines a routed bank of restoration experts with a foreground-mask branch. ACO-MoE is pretrained entirely offline on synthetic rendered data with automatically generated degradation pairs and simulation-derived foreground masks, requiring no manual annotation. At inference time, it takes only corrupted RGB as input without corruption labels, clean reference frames, or foreground masks. Across VDCS, DMC-GB, and RoboSuite, ACO-MoE consistently improves downstream control with both model-free and model-based backbones, recovering 95.3% of clean-input performance under challenging Markov-switching corruptions. It also generalizes zero-shot to unseen visual perturbations excluded from adapter pretraining. Source code is available at https://github.com/fangzr/aco-moe-code.

1. Introduction

Modern visual systems are increasingly deployed in environments where the input stream is unreliable. Weather phenomena (rain, haze, snow), sensor noise, lossy compression, and changing scene context all corrupt observations in ways the downstream model was not trained to handle. Image restoration has made substantial progress on individual corruption types—denoising [8], deraining [61], and unified restorers [3, 33, 43, 46] that handle several degradations within a single network. However, two assumptions of the standard restoration setup limit its usefulness in real deployments. *First*, evaluation typically applies a single fixed corruption per clip, whereas real environments exhibit *non-stationary* corruption, where the active degradation type can switch within seconds. *Second*, restoration is optimized for pixel-level fidelity against a clean target, with no notion of which pixels actually matter to the downstream system that consumes the restored output.

These limitations become most consequential when the downstream task is closed-loop visual control, where every restored frame feeds directly into a learned policy or world model. Modern visual control agents [39, 62, 13, 38, 71, 48, 73, 21, 79, 77, 63, 23, 75, 58, 51, 22, 20] perform well on clean inputs

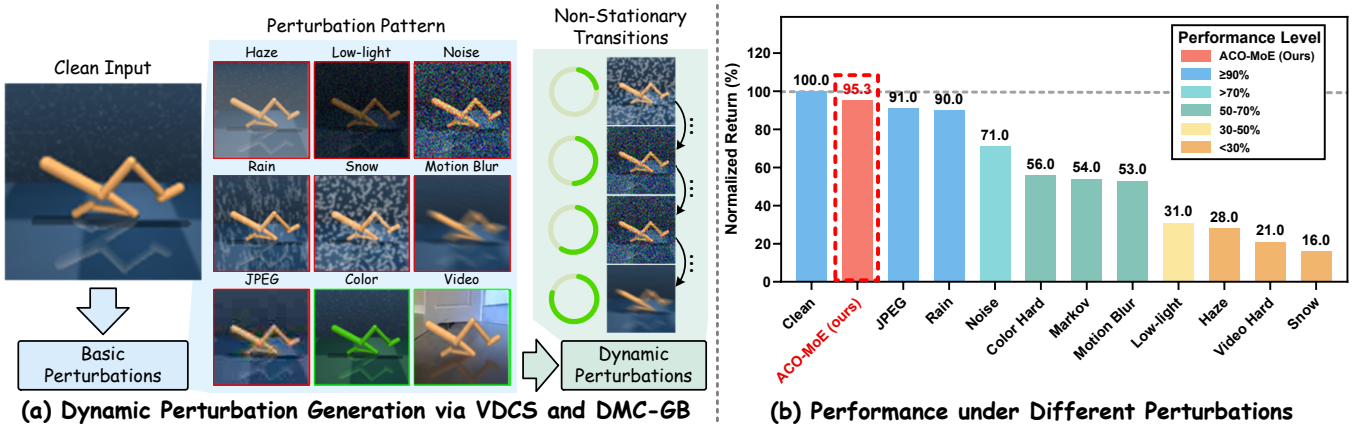


Figure 1: **Dynamic perturbation examples and performance degradation on DMControl [55]. Left:** Diverse perturbations including 7 VDCS physical degradations (red border) and 2 DMC-GB background distractions (green border); VDCS types switch dynamically via Markov chains during deployment. **Right:** Normalized returns of DreamerV3 [13] drop significantly under all disturbances, degrading to 54% under Markov switching.

but degrade catastrophically under even mild visual perturbation (Fig. 1). The failure mechanism varies with architecture: model-free encoders confuse task-relevant state with corruption artifacts [42, 54], while reconstruction-based world models such as DreamerV3 [13] are trained to faithfully predict corrupted observations, forcing the latent state to encode corruption-specific patterns rather than task content. Non-reconstructive planners such as TD-MPC2 [18] likewise consume pixel observations and propagate corruption errors through every imagined rollout. We formalize this in Sec. 5 and show that accurate reconstruction provably contaminates the representation with nuisance corruption information.

Two existing lines of work address parts of this problem, yet neither fully resolves it. *Generic image restoration* methods [33, 43, 46] handle diverse corruptions in a task-agnostic manner, but are evaluated on i.i.d. clip-level corruptions and have not been integrated as preprocessing for closed-loop systems under switching conditions. *Robust visual control* either applies data augmentation [17, 31, 30, 52] and domain randomization, or learns perturbation-invariant representations through contrastive or regularization objectives [1, 69, 74, 70, 14]. Both approaches retain corrupted pixels in the input of the policy encoder; when the foreground itself is degraded, no downstream invariance objective can recover information already lost at the pixel level. Notably, neither line of work performs *task-aware* restoration of the observation before the downstream model consumes it.

A widely adopted testbed for visual robustness is the DMControl Generalization Benchmark (DMC-GB) [15], which evaluates robustness to background distraction such as natural-video backgrounds and slight foreground color changes. However, DMC-GB does not assess the impact of physical image degradations that are commonplace in real deployments: weather phenomena alter scene geometry and texture; sensor-induced artifacts (Gaussian noise, motion blur, JPEG compression) corrupt pixel statistics globally; low-light conditions reduce perceptual contrast. Furthermore, DMC-GB applies a single fixed disturbance per episode, whereas real deployments exhibit non-stationary transitions between perturbation types. Vision corruption benchmarks [19, 6, 28] and test-time adaptation methods [2, 36] also assume a single fixed corruption per clip and have not been integrated with closed-loop visual systems. Taken together, these observations reveal three concrete challenges for robust visual perception under dynamic perturbations:

1. **Non-stationary benchmarking:** How can we evaluate *temporally switching* physical degradations beyond static background distractions?
2. **Heterogeneity:** How can a single module handle multiple corruptions with incompatible inverse mappings

without knowing the active corruption?
3. **Task-relevance:** How can we restore observations in a way that preserves the information downstream models actually need, while remaining stable as a frozen preprocessing module?

To address these challenges, we introduce the Visual Degraded Control Suite (VDCS) to generate Markov-switching corruption sequences (Fig. 1a), and present *Agent-Centric Observations with Mixture-of-Experts* (ACO-MoE), a frozen, plug-and-play observation adapter that pairs a routed bank of restoration experts with a foreground-mask branch. The router dynamically selects input-adaptive latent experts without requiring any oracle corruption label; the selected expert simultaneously repairs degraded RGB and segments the task-relevant foreground, compositing it onto a uniform black background. The entire module remains *frozen* during downstream backbone learning to prevent observation drift. As shown in Fig. 1b, this decoupling of perception from perturbation substantially mitigates the degradation suffered by downstream visual-control backbones across diverse corruption types. Beyond the corruption families used for adapter pretraining, ACO-MoE also maintains strong zero-shot performance under unseen OOD visual perturbations. Specifically, our contributions are summarized as follows:

(1) We introduce VDCS, a benchmark extending DeepMind Control Suite [55] with seven physical degradations under Markov-switching dynamics, enabling systematic evaluation of restoration and downstream control under non-stationary visual corruption.
(2) We formally show that reconstruction-based representations inevitably entangle corruption information in their latent state, and prove that foreground anchoring is a valid information-bottleneck surrogate that exactly eliminates this entanglement.
(3) We propose ACO-MoE, a dual-stream Mixture-of-Experts architecture that achieves task-aware restoration by simultaneously repairing corrupted pixels and extracting task-relevant foreground, requiring no oracle corruption labels at training or inference.
(4) Across VDCS, DMC-GB (Color/Video Hard), and RoboSuite, ACO-MoE achieves state-of-the-art performance with both model-free and model-based downstream backbones. Additional comparisons with frozen generic restorers further show that the gains arise from task-aware agent-centric adaptation rather than pixel restoration alone.

2. Related Work

Robust visual control under perturbations. Existing approaches mitigate visual perturbations through three complementary lines. Data augmentation [64, 17, 31, 30, 52] narrows the train–test gap via random shifts and color jitter, while invariant-representation methods [32, 74, 66, 1, 70, 62, 40, 14, 69] learn encoders that suppress corruption-specific features through contrastive or regularization objectives. Both keep corrupted pixels in the encoder's input; when foreground texture is physically destroyed, no downstream invariance objective can recover what is already lost. Foreground-aware adapters [59, 56, 29] address this with object-centric priors; the closest work, FTR [56], uses SAM-supervised masking with 200K test-time adaptation steps. On the model-based side, world models [11, 12, 38, 71, 48, 13, 18, 76, 77, 63, 65] share the same vulnerability: reconstruction-based latents encode nuisance corruption features [42, 54], and even non-reconstructive planners require clean visual input. ACO-MoE instead restores observations *before* the backbone consumes them, requires no test-time adaptation, and leaves the backbone unchanged.

Visual corruption benchmarks. ImageNet-C [19], RobustBench [6], and 3D Common Corruptions [28] stress-test vision models under fixed corruption types per image, while test-time adaptation methods [2, 36] address shift without explicit labels. For closed-loop control, the Distracting Control Suite [53], DMC-GB [15], and broader generalization benchmarks [67, 41, 9, 45] focus on background distractors and apply at most one fixed disturbance per episode. None evaluate intra-episode switching among heterogeneous physical

degradations, which our VDCS benchmark targets.

Image restoration and Mixture-of-Experts. Restoration architectures evolved from U-Nets [49] and transformers [68] to all-in-one restorers [33, 43, 60, 5, 10, 37, 72, 57, 46] that handle multiple corruption types in a single network, evaluated under i.i.d. clip-level corruptions with pixel-fidelity metrics. Sparse Mixture-of-Experts [25, 27, 50, 44, 78, 26, 35] routes inputs to specialized sub-networks; in vision it has been applied to restoration [46] and classification [47]. We extend these tools rather than compete with them: pairing restoration with a foreground-mask branch so fidelity is concentrated on task-relevant content, and routing at the frame level under Markov-switching corruption without any oracle corruption label.

3. Preliminaries

Visual control from pixels. We model pixel-based control as a Partially Observable Markov Decision Process (POMDP) $(\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \Omega, \gamma)$, where $s_t \in \mathcal{S}$ is the latent environment state, $a_t \in \mathcal{A}$ the action, $\mathcal{T}(s_{t+1} \mid s_t, a_t)$ the transition dynamics, and $r_t = R(s_t, a_t)$ the reward. The agent does not observe s_t directly; instead, it receives a pixel observation $x_t \in \Omega$ and learns a history-dependent policy $\pi(a_t \mid x_{\le t})$ maximizing $J(\pi) = \mathbb{E}_\pi\big[\sum_{t=0}^{\infty} \gamma^t r_t\big]$.

Observation model (clean vs. corrupted). Let $o_t^{\text{raw}} = \mathcal{R}(s_t) \in [0, 255]^{H \times W \times 3}$ denote the clean 8-bit RGB frame rendered by the simulator renderer $\mathcal{R}$. VDCS introduces an exogenous corruption mode index k_t at each time step: let $|\mathcal{K}|$ be the number of corruption types and define the index set $\mathcal{K} := \{1, \dots, |\mathcal{K}|\}$, with $k_t \in \mathcal{K}$. Given k_t, the observed 8-bit frame is generated by

$$x_t^{\text{raw}} = \mathcal{D}_{k_t}(o_t^{\text{raw}};\, \iota_t, \xi_t)\,, \tag{1}$$

where $\mathcal{D}_k$ is the k-th degradation operator, $\iota_t \in [0, 1]$ is its (mode-dependent) severity, and ξ_t denotes randomness. Backbones consume normalized pixels:

$$x_t \;=\; 2(x_t^{\text{raw}}/255) - 1 \;\in\; [-1, 1]^{H \times W \times 3}, \tag{2}$$

The VDCS benchmark specifies the dynamics of (k_t, ι_t), including Markov switching, severity sampling, and temporal correlation, in Section 4. The corruption index k_t is part of the benchmark generation process; it is not provided to ACO-MoE as a router target in our main configuration. We also decompose the clean observation as $O_t = (F_t, B_t)$, where F_t denotes the foreground pixels (agent and task-relevant objects) and B_t the background. In Appendix B, we use uppercase (S_t, O_t, X_t, K_t) to denote random variables and lowercase (s_t, o_t, x_t, k_t) for their realizations.

Backbone world models. ACO-MoE is evaluated with two complementary visual-control backbones while leaving their architectures and learning objectives unchanged.

DreamerV3 [13]. DreamerV3 learns a recurrent state-space model (RSSM) from pixels and trains an actor–critic purely in latent imagination. An encoder infers a stochastic latent $z_t \sim q_\phi(z_t \mid h_t, x_t)$ and a recurrent dynamics model updates $h_t = f_\phi(h_{t-1}, z_{t-1}, a_{t-1})$. The world-model objective is

$$\mathcal{L}_{\text{wm}} = \mathbb{E}\Big[\underbrace{-\log p_\phi(x_t \mid h_t, z_t)}_{\text{reconstruction}} + \underbrace{\beta_{\text{dyn}}\,\mathrm{KL}\big[\mathrm{sg}(q_\phi) \,\|\, p_\phi\big]}_{\text{dynamics}} + \underbrace{\beta_{\text{rep}}\,\mathrm{KL}\big[q_\phi \,\|\, \mathrm{sg}(p_\phi)\big]}_{\text{representation}} + \mathcal{L}_{\text{pred}}(r_t, c_t)\Big], \tag{3}$$

where $q_\phi \equiv q_\phi(z_t \mid h_t, x_t)$, $p_\phi \equiv p_\phi(z_t \mid h_t)$, $\mathrm{sg}(\cdot)$ is stop-gradient, h_t is the deterministic recurrent state, β_{dyn} and β_{rep} are KL balancing weights, $c_t \in \{0, 1\}$ is the episode continuation flag, and $\mathcal{L}_{\text{pred}}$ supervises reward

and continuation prediction. The reconstruction term can encourage the latent state to retain nuisance visual information when the input observation is corrupted. We discuss this failure mode in Appendix B.

TD-MPC2 [18]. TD-MPC2 performs MPC in a latent space learned *without* pixel reconstruction: observations are encoded as $z = h(x)$, trained via joint-embedding and reward/value prediction objectives. This sidesteps reconstruction-driven contamination, but the encoder $h(\cdot)$ still requires clean visual input to produce reliable latent states for planning.

ACO-MoE as clean-background observation adaptation. Following the policy-reuse perspective of visual adaptation methods such as FTR [56], we separate visual observation adaptation from downstream control learning. The downstream visual-control backbone is trained in simulation on clean-background observations, where the agent and task-relevant foreground are rendered against a uniform background. ACO-MoE is then pretrained offline to map corrupted RGB observations back to this clean-background observation space. During perturbation evaluation, ACO-MoE is inserted before the downstream encoder and the backbone architecture and learning objective are left unchanged. This design tests whether a clean-background policy or world model can be reused under dynamic visual perturbations through observation adaptation rather than through a new RL algorithm.

4. Visual Degraded Control Suite (VDCS)

VDCS extends DMControl with a *Markov-switching* visual corruption process to emulate time-varying disturbances in real control systems, such as robots and autonomous platforms, where weather, illumination, sensor noise, camera-shake motion blur, and bandwidth-driven compression typically *persist for a while* and *switch sporadically*. Concretely, VDCS corrupts the rendered RGB stream by applying the observation model in Eq. (1), where the corruption mode k_t evolves as a sticky Markov chain and the severity ι_t is temporally correlated within each persistent segment. The corruption index k_t is part of the VDCS data-generation process and is used to define the benchmark dynamics; it is not provided to ACO-MoE as a router target in our main configuration.

We consider seven physical degradations: rain, haze, snow, motion blur, Gaussian noise, low-light, and JPEG compression. VDCS is evaluated *online* by corrupting frames on the fly during downstream visual-control interaction. To pretrain ACO-MoE, we also generate a paired *offline* dataset of clean/corrupted frame pairs using the same operator family, together with pseudo foreground masks for compositing supervision. This offline dataset provides clean RGB targets and pseudo masks for observation adaptation, but no corruption-label supervision is used for router training. Full benchmark details, parameter values, and operator definitions are given in Appendix A.

5. ACO-MoE

As shown in Fig. 2, ACO-MoE is a frozen, plug-and-play observation adapter placed before a downstream policy or world model. It maps corrupted RGB inputs x_t to agent-centric observations $\tilde{x}_t$ that preserve task-relevant foreground information while suppressing nuisance visual variation. Following the policy-reuse perspective of visual adaptation methods such as FTR [56], we separate visual observation adaptation from downstream control learning: the visual-control backbone is trained in simulation on clean-background observations, while ACO-MoE is pretrained offline to map corrupted observations back to this clean-background observation space. During perturbation-time evaluation, ACO-MoE receives only corrupted RGB frames and the downstream backbone architecture and learning objective are left unchanged.

ACO-MoE consists of a shared encoder, a compact router, and N_e repair experts. The expert count N_e is an

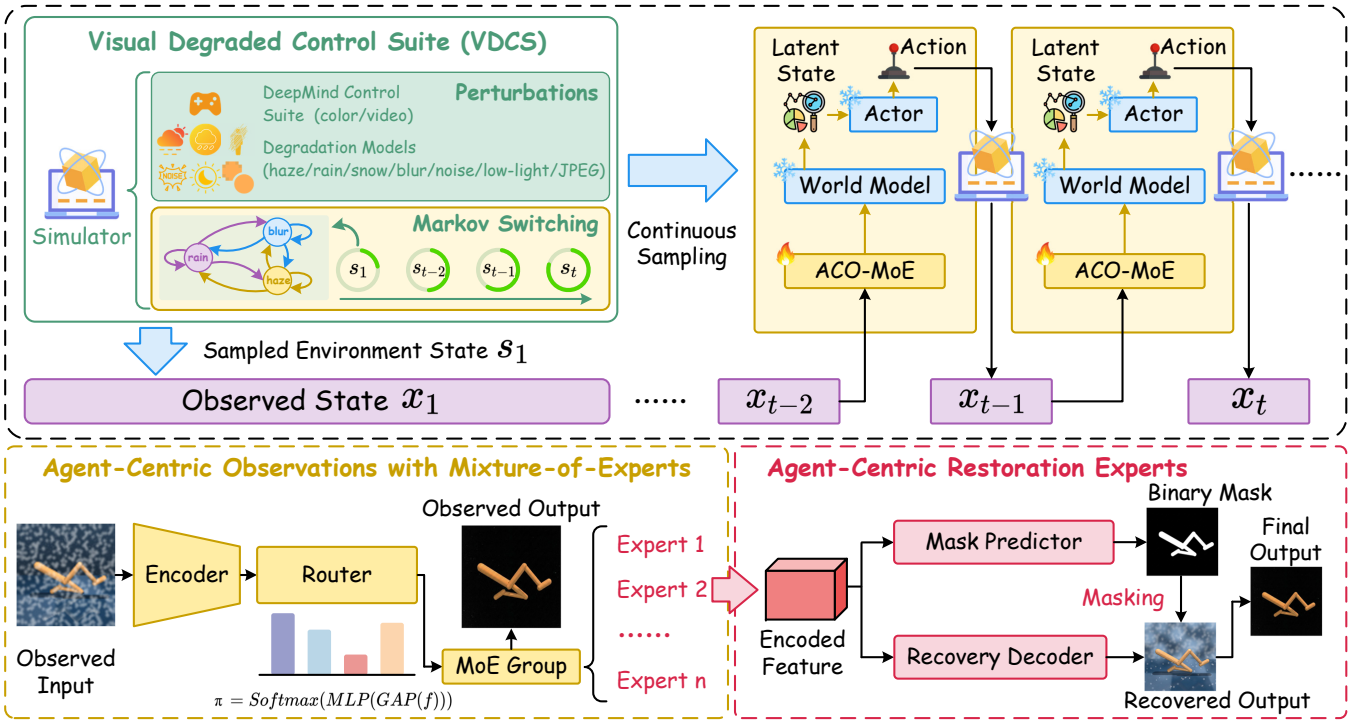


Figure 2: VDCS and ACO-MoE overview. VDCS applies Markov-temporal visual perturbations to DMControl to generate degraded RGB inputs x_t. ACO-MoE is a frozen observation adapter placed before a downstream visual-control backbone. Given only the corrupted RGB input, its learned router selects a repair expert that predicts an RGB residual and a foreground mask, and then composes an agent-centric observation on a clean background.

architectural capacity hyperparameter. Experts are not assigned to predefined corruption categories, and the router is not trained with corruption labels in our main configuration. Each expert predicts an RGB residual for corruption repair and a foreground mask for agent-centric composition. The final observation is obtained by compositing the repaired foreground onto a clean background.

Design motivation. ACO-MoE is motivated by a simple observation: visual fidelity is useful only insofar as it preserves task-relevant information. Dynamic corruptions alter foreground appearance, background statistics, and the latent representations consumed by downstream models. Rather than redesigning the downstream backbone, ACO-MoE transforms corrupted observations toward a canonical agent-centric form: it repairs foreground pixels when they are degraded and removes background variation through foreground composition. A more formal discussion of reconstruction-induced nuisance retention and foreground bottlenecks is given in Appendix B.

5.1. Agent-Centric Observation Construction

Given a corrupted observation x_t normalized by Eq. (2), ACO-MoE predicts a repaired RGB image $\hat{o}_t \in [-1,1]^{H\times W\times 3}$ and a foreground probability map

$$m_t = \mathrm{softmax}\left(\ell_t^{\mathrm{mask}}\right)_{[\mathrm{fg}]} \in [0,1]^{H\times W}.$$

The agent-centric observation is

$$\tilde{x}_t = \hat{o}_t \odot m_t + b \odot (1 - m_t), \tag{4}$$

where b is a clean background value, set to -1 under the normalized pixel range, and $\odot$ denotes element-wise multiplication with broadcasting across RGB channels.

The clean-background target is generated offline from simulation as $o_t^{\mathrm{cb}} = o_t \odot m_t^{\mathrm{ps}} + b \odot (1 - m_t^{\mathrm{ps}})$, where o_t is the normalized clean rendered frame and $m_t^{\mathrm{ps}} \in \{0,1\}^{H\times W}$ is a pseudo foreground mask obtained under a uniform-background rendering protocol via color-threshold segmentation. The same clean-background observation space is used to train the downstream visual-control backbone. At perturbation-time evaluation, however, ACO-MoE receives only the corrupted RGB observation x_t and predicts m_t from RGB alone; no clean frame, simulator mask, or corruption label is provided.

5.2. Agent-Centric Repair Experts

As shown in Fig. 2, ACO-MoE employs a shared encoder that extracts multi-scale features from x_t, a compact router, and N_e repair experts. Each expert $j \in \{1,\dots,N_e\}$ predicts an RGB residual $\Delta_t^{(j)} \in \mathbb{R}^{H\times W\times 3}$ and mask logits $\ell_t^{\mathrm{mask},(j)} \in \mathbb{R}^{H\times W\times 2}$. The router produces weights $\pi_t \in \Delta^{N_e-1}$ over experts. During pretraining, we use soft routing for differentiability:

$$\Delta_t = \sum_{j=1}^{N_e} \pi_{t,j}\Delta_t^{(j)}, \qquad \ell_t^{\mathrm{mask}} = \sum_{j=1}^{N_e} \pi_{t,j}\ell_t^{\mathrm{mask},(j)}. \tag{5}$$

The repaired RGB image is then computed as

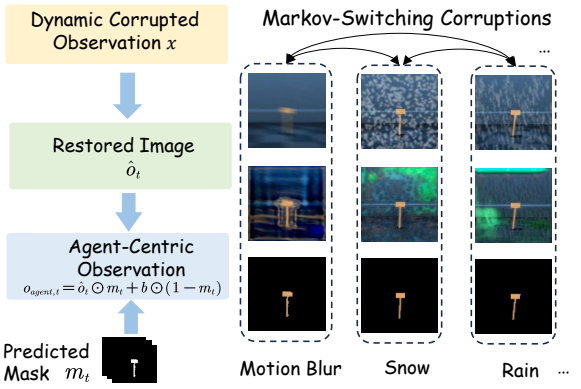


Figure 3: Task-oriented observation adaptation. The RGB branch repairs corrupted visual features, while the mask branch localizes the agent-centric foreground for control-oriented composition.

$$\hat{o}_t = \mathrm{clip}_{[-1,1]}(x_t + \tanh(\Delta_t)), \tag{6}$$

where $\mathrm{clip}_{[-1,1]}(\cdot)$ clips values element-wise to $[-1,1]$. At inference, we use hard top-1 routing, $j^\star = \arg\max_j \pi_{t,j}$, and evaluate only the selected expert. Fig. 3 visualizes the resulting RGB repair, mask prediction, and agent-centric observation. Unless otherwise stated, we use a small fixed expert budget N_e as an architectural capacity hyperparameter; N_e is not treated as the number of corruption modes and does not define any expert assignment.

5.3. Router and Training Objectives

Offline pretraining data. Following the clean-simulation policy-reuse setup in FTR [56], we construct paired offline data $(x_t, o_t, o_t^{\mathrm{cb}}, m_t^{\mathrm{ps}})$, where x_t is a corrupted RGB observation synthesized by VDCS or DMC-GB, o_t is the corresponding clean rendered frame, $m_t^{\mathrm{ps}} \in \{0,1\}^{H\times W}$ is the pseudo foreground mask from uniform-background rendering, and o_t^{cb} is the clean-background target defined in Sec. 5.1. The downstream backbone is trained on this clean-background observation space, while ACO-MoE is pretrained to map corrupted observations x_t back to it.

Training objective. The total pretraining loss is

$$\mathcal{L} = \lambda_{\mathrm{rgb}}\mathcal{L}_{\mathrm{rgb}} + \lambda_{\mathrm{mask}}\mathcal{L}_{\mathrm{mask}} + \lambda_{\mathrm{final}}\mathcal{L}_{\mathrm{final}}, \tag{7}$$

where $\mathcal{L}_{\mathrm{rgb}} = \|\hat{o}_t - o_t\|_1$, $\mathcal{L}_{\mathrm{mask}} = \mathrm{CE}(\ell_t^{\mathrm{mask}}, m_t^{\mathrm{ps}})$, and $\mathcal{L}_{\mathrm{final}} = \|\tilde{x}_t - o_t^{\mathrm{cb}}\|_1$. Here $\mathcal{L}_{\mathrm{rgb}}$ supervises RGB repair toward the clean rendered frame, $\mathcal{L}_{\mathrm{mask}}$ supervises foreground prediction, and $\mathcal{L}_{\mathrm{final}}$ supervises the

final clean-background agent-centric observation. The shared encoder, router, and per-expert decoders are optimized jointly under this objective. Pretraining hyperparameters and computational overhead are detailed in Appendix E.

6. Experiments

6.1. Experimental Setup

Benchmarks. **VDCS** extends DMControl with Markov-switching physical degradations (Section 4). In the Markov-temporal setting, k_t follows a sticky chain (p_s=0.8), creating temporally persistent, non-stationary corruptions that stress-test perception and world-model consistency. **DMC-GB** [15] applies background distractions (`color_hard`, `video_hard`) at the episode level, serving as a complementary test of generalization to a different perturbation regime.

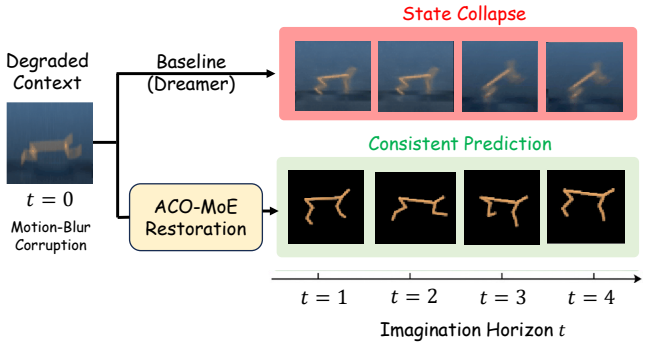


Figure 4: Robustness of Imagination. Under motion-blur corruption (t=0), DreamerV3 suffers state collapse on raw degraded input (red); ACO-MoE maintains consistent predictions with restored agent-centric observations (green).

Tasks. We select 8 DMControl tasks covering diverse morphologies and difficulties: from relatively simple locomotion (`cartpole_swingup`, `walker_walk`) to highly dynamic whole-body control (`hopper_hop`, `cheetah_run`) and fine-grained manipulation (`finger_spin`, `finger_turn_hard`). The 2 RoboSuite tasks (`Door`, `Lift`) extend evaluation to contact-rich manipulation with TD-MPC2 [18] as the backbone, validating plug-and-play transferability across RL algorithms.

Baselines. We compare against two categories of methods. *Data-augmentation methods*: DrQ-v2 [64], SVEA [17], SODA [15], SGQN [1], SimGRL [52], Q^2 [34], which represent the dominant paradigm of learning robust representations via augmentation, providing a strong baseline for model-free robustness. *Task-aware methods*: FTR [56] is the most relevant prior work, which also uses foreground-based adaptation but relies on SAM-supervised test-time fine-tuning, making it a direct point of comparison for our frozen preprocessing design. On DMC-GB we additionally include PAD [16], DrQ [30], SRM [24], and SMG [70]. In Appendix F.6, we further compare against frozen generic image restorers, including PromptIR [43], Restormer [68], AdaIR [7], and NAFNet [4], to test whether pixel-level restoration alone explains the gains.

Training settings. All methods train for 1M environment steps (DMControl) or 200K steps (RoboSuite). Evaluation uses 5 seeds with 10 episodes per seed, reporting mean $\pm$ std of episode returns. More implementation details are in Appendix E.

6.2. Main Results on VDCS

Table 1 reports performance under VDCS Markov-temporal perturbations. ACO-MoE achieves the highest average score of 736.0, outperforming the best baseline SimGRL (412.3) by 78.5% relative. Strong gains appear on tasks where physical corruptions most severely disrupt low-level texture cues: `finger_turn_hard` (+207.3%), `cheetah_run` (+126.5%), and `hopper_hop` (+105.5%).

Table 1: Performance comparison on VDCS Markov-temporal perturbations. Results show mean ± std over evaluation episodes. Red: best performance; Blue: second best.

Task	DrQ-v2	SVEA	SODA	SGQN	SimGRL	Q^2	FTR	**ACO-MoE (Ours)**	△
cartpole, swingup	49 ±79	649 ±96	615 ±149	354 ±61	613 ±176	683 ±230	342 ±53	864 ±75	+181 26.5%
finger, spin	2 ±1	961 ±32	307 ±120	948 ±87	778 ±69	379 ±63	545 ±54	492 ±102	−469 −48.8%
finger, turn hard	68 ±38	60 ±129	59 ±38	287 ±89	31 ±20	23 ±15	105 ±37	882 ±65	+595 207.3%
hopper, stand	4 ±3	69 ±89	475 ±73	5 ±3	603 ±80	310 ±178	659 ±75	910 ±50	+251 38.1%
hopper, hop	3 ±2	41 ±27	44 ±15	6 ±4	3 ±2	165 ±50	109 ±15	339 ±46	+174 105.5%
cheetah, run	6 ±3	221 ±43	158 ±62	166 ±35	237 ±85	309 ±67	4 ±2	700 ±95	+391 126.5%
walker, walk	45 ±24	837 ±42	171 ±58	792 ±87	845 ±97	443 ±85	532 ±38	963 ±87	+118 14.0%
walker, run	25 ±13	279 ±59	128 ±22	229 ±74	188 ±64	57 ±37	232 ±31	738 ±56	+459 164.5%
Average	25.3 ±25.2	389.6 ±335.7	244.6 ±211.0	348.4 ±333.1	412.3 ±314.9	296.1 ±202.9	316.1 ±226.5	736.0 ±205.6	+323.7 78.5%

Fig. 4 further illustrates why: without preprocessing, DreamerV3's reconstruction objective causes state collapse under corrupted context, while ACO-MoE produces consistent future predictions by decoupling perception from perturbation before world-model inference.

Across all eight tasks, ACO-MoE recovers 95.3% of DreamerV3's clean performance on average (Appendix F.1). The only clear exception is `finger_spin`, where the limitation comes mainly from the downstream backbone: DreamerV3 reaches only 543 even on clean observations, and ACO-MoE recovers 90.6% of this clean score. Replacing DreamerV3 with DrQ-v2 yields 95.6% recovery on the same task, supporting that the failure is not due to observation adaptation (Appendix F.4). Appendix F.6 further shows that ACO-MoE outperforms the strongest frozen generic restorer, AdaIR, by +220.3 average return (+40.3%), supporting the need for task-aware foreground composition. Beyond the VDCS corruption families used for adapter pretraining, ACO-MoE also shows meaningful zero-shot generalization to unseen OOD visual perturbations, with both the adapter and controller kept frozen and no test-time adaptation (Appendix F.7).

6.3. Ablations

Table 2: DMC-GB averaged results. Red: best; Blue: second best. $FTR^{\dagger}$ not evaluated on Color Hard.

Method	Video Hard	Color Hard
SGQN	590 ± 275	593 ± 223
SMG	749 ± 261	697 ± 232
$FTR^{\dagger}$	762 ± 209	—
ACO-MoE	**809 ± 194**	**818 ± 196**
vs. 2nd best	*+6%*	*+17%*

Fig. 5 isolates the main components of ACO-MoE on all 8 VDCS tasks. Replacing the expert router with a capacity-matched single U-Net (**Single Expert**) reduces average return by 10.9%, suggesting that input-adaptive expert selection is more effective than adding comparable capacity to one decoder. Removing RGB repair (**Mask-Only**) causes a smaller drop (10.6%), while removing foreground supervision (**RGB-Only**) is much more harmful (57.1%), indicating that foreground composition is the dominant factor and RGB repair provides complementary gains under pixel-level degradations. Removing the repair branch at inference (**w/o Repair**) also stays below the full model, confirming that masking alone is insufficient under strong physical corruptions. Additional ablations on freezing, no-preprocessing, and DMC-GB are reported in Appendix F.5.

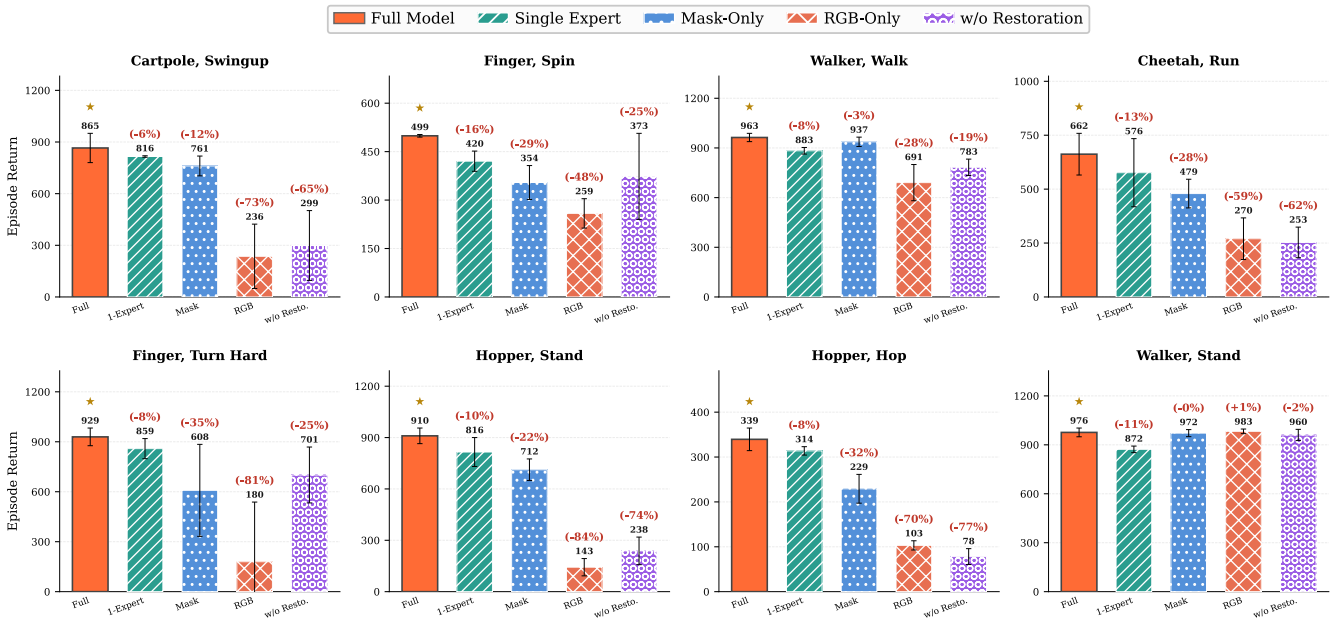
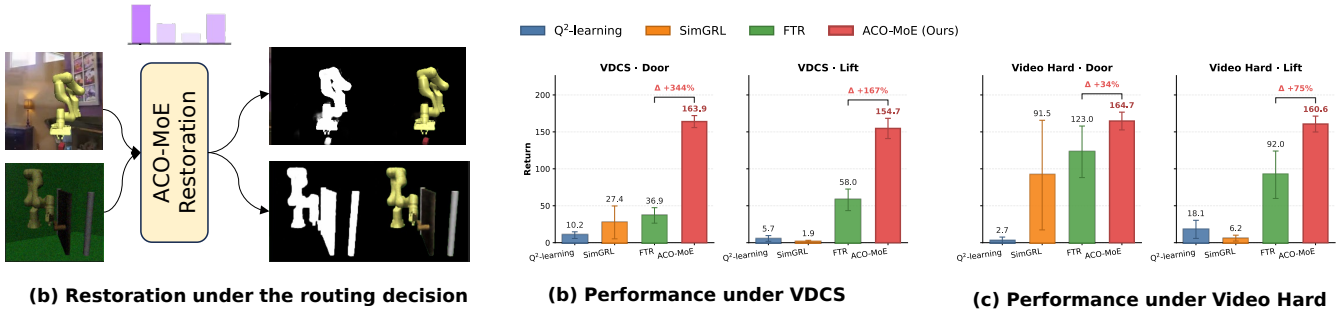


Figure 5: **Component ablation across 8 VDCS tasks. Single Expert**: replace ACO-MoE with a capacity-matched U-Net. **Mask-Only**: disable RGB repair loss. **RGB-Only**: disable mask loss. **w/o Repair**: remove RGB repair at inference while keeping foreground composition. Red numbers denote relative drops w.r.t. the full model.



(b) Restoration under the routing decision **(b) Performance under VDCS** **(c) Performance under Video Hard**

Figure 6: **RoboSuite results.** (a) Restored RGB, mask, and agent-centric observation under Video Hard and Low Light. (b) VDCS: +344% on Door, +167% on Lift vs. Q^2. (c) Video Hard: +34% on Door, +75% on Lift vs. FTR.

6.4. Analysis

Generalization on RoboSuite. Fig. 6 evaluates contact-rich RoboSuite tasks with TD-MPC2 as the downstream backbone. ACO-MoE improves over Q^2 by +344% on `Door` and +167% on `Lift` under VDCS, and over FTR by +34% and +75% under Video Hard. These results show that the same frozen observation adapter can transfer from DreamerV3-style DMControl evaluation to TD-MPC2-based manipulation without changing the downstream control backbone.

Generalization on DMC-GB. Table 2 evaluates generalization on DMC-GB [15] across five tasks (`cartpole_swingup`, `finger_spin`, `walker_stand`, `walker_walk`, `cheetah_run`). ACO-MoE achieves state-of-the-art averages of **809** (video-hard) and **818** (color-hard), outperforming the second-best method FTR by **6.2%** on video-hard and SMG by **17.4%** on color-hard, respectively.

These corruptions differ fundamentally from VDCS: background pixels are replaced while foreground

remains intact. Yet ACO-MoE benefits from the same mechanism: compositing the agent foreground onto a uniform black background eliminates nuisance variation regardless of its origin. The `finger_spin` underperformance persists for the backbone reason identified in Section 6.2. Full per-task results against 10+ recent baselines are reported in Appendix F.2.

7. Conclusion, Limitation, and Future Work

We studied visual RL under dynamic corruptions and showed that reconstruction-based world models entangle corruption factors in latent space, motivating agent-centric preprocessing. We introduced ACO-MoE, a plug-and-play dual-stream Mixture-of-Experts that restores task-relevant RGB and masks nuisance background, producing corruption-invariant observations for model-free and model-based backbones. Our analysis formalizes reconstruction-induced contamination and foreground anchoring as a valid design principle. Across VDCS, DMC-GB, RoboSuite, and unseen OOD perturbations, ACO-MoE improves performance, outperforms frozen generic restorers, generalizes across regimes, and transfers from DreamerV3 to TD-MPC2 and DrQ-v2. Ablations confirm the importance of input-adaptive expert capacity, dual-stream outputs, and frozen training.

However, ACO-MoE still relies on offline-trained experts, and severe unseen perturbations that destroy spatial structure or occlude task-relevant foreground remain challenging. Foreground extraction also degrades on fine-grained tasks with sub-pixel-scale targets under strong pixel-level degradations. While robust observation adaptation may improve visual-control reliability under adverse sensing, simulation robustness should not be viewed as a deployment-ready safety guarantee. Future work includes open-set routing, real-world pseudo masks/pairs (Appendix D.2), physical-robot validation, failure monitoring, and conservative fallback mechanisms.

References

[1] 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.

[2] Malik Boudiaf, Romain Mueller, Ismail Ben Ayed, and Luca Bertinetto. Parameter-free online test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8344–8353, 2022.

[3] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *European Conference on Computer Vision (ECCV)*, volume 13667, pages 17–33, 2022.

[4] Xiaojie Chu, Liangyu Chen, and Wenqing Yu. Nafssr: Stereo image super-resolution using nafnet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1239–1248, June 2022.

[5] Marcos V Conde, Gregor Geigle, and Radu Timofte. InstructIR: High-quality image restoration following human instructions. In *European Conference on Computer Vision*, pages 1–21. Springer, 2024.

[6] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. RobustBench: A standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

[7] Yuning Cui, Syed Waqas Zamir, Salman Khan, Alois Knoll, Mubarak Shah, and Fahad Shahbaz Khan. AdaIR: Adaptive all-in-one image restoration via frequency mining and modulation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[8] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

[9] Bryan LM de Oliveira, Luana GB Martins, Bruno Brandão, Murilo L da Luz, Telma W de L Soares, and Luckeciano C Melo. Sliding puzzles gym: A scalable benchmark for state representation in visual reinforcement learning. *arXiv preprint arXiv:2410.14038*, 2024.

[10] Hang Guo, Jinmin Li, Tao Dai, Zhihao Ouyang, Xudong Ren, and Shu-Tao Xia. MambaIR: A simple baseline for image restoration with state-space model. In *European Conference on Computer Vision (ECCV)*, pages 222–241. Springer, 2024.

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

[12] 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*, volume 97, pages 2555–2565. PMLR, 2019.

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

[14] Xudong Han, Senkang Hu, Yihang Tao, Yu Guo, Philip Birch, Sam Tak Wu Kwong, and Yuguang Fang. Dsp-reg: Domain-sensitive parameter regularization for robust domain generalization, 2026.

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

[16] 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.

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

[18] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

[19] Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[20] Senkang Hu, Zhengru Fang, Zihan Fang, Yiqin Deng, Xianhao Chen, and Yuguang Fang. Agentscodriver: Large language model empowered collaborative driving with lifelong learning, 2024.

[21] Senkang Hu, Zhengru Fang, Yiqin Deng, Xianhao Chen, Yuguang Fang, and Sam Kwong. Toward full-scene domain generalization in multi-agent collaborative bird's eye view segmentation for connected and autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 26(2):1783–1796, 2025.

[22] Senkang Hu, Zhengru Fang, Zihan Fang, Yiqin Deng, Xianhao Chen, Yuguang Fang, and Sam Tak Wu Kwong. Agentscomerge: Large language model empowered collaborative decision making for ramp merging. *IEEE Transactions on Mobile Computing*, 24(10):9791–9805, 2025.

[23] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17853–17862, 2023.

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

[25] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[26] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lelio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Theophile Gervet, Thibaut Lavril, Thomas J. Wang, Timothee Lacroix, and William El Sayed. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

[27] Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6(2):181–214, 1994.

[28] Oğuzhan Fatih Kar, Teresa Yeo, Andrei Atanov, and Amir Zamir. 3D common corruptions and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18963–18974, 2022.

[29] Kyungmin Kim, JB Lanier, Pierre Baldi, Charless Fowlkes, and Roy Fox. Make the pertinent salient: Task-relevant reconstruction for visual control with distractions. *arXiv preprint arXiv:2410.09972*, 2024.

[30] Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

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

[32] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, volume 119, pages 5639–5650. PMLR, 2020.

[33] Boyun Li, Xiao Liu, Peng Hu, Zhongqin Wu, Jiancheng Lv, and Xi Peng. All-in-one image restoration for unknown corruption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17452–17462, 2022.

[34] Weijian Liao, Zongzhang Zhang, and Yang Yu. Policy-independent behavioral metric-based representation for deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8746–8754, 2023.

[35] Bin Lin, Zhenyu Tang, Yang Ye, Jinfa Huang, Junwu Zhang, Yatian Pang, Peng Jin, Munan Ning, Jiebo Luo, and Li Yuan. MoE-LLaVA: Mixture of experts for large vision-language models. *IEEE Transactions on Multimedia*, 2026.

[36] Yuejiang Liu, Parth Kothari, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. TTT++: When does self-supervised test-time training fail or thrive? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 21808–21820, 2021.

[37] Ziwei Luo, Fredrik K. Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B. Schön. Controlling vision-language models for multi-task image restoration. In *International Conference on Learning Representations (ICLR)*, 2024.

[38] Vincent Micheli, Eloi Alonso, and Francois Fleuret. Transformers are sample-efficient world models. *arXiv preprint arXiv:2209.00588*, 2022.

[39] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[40] Mitsuhiko Nakamoto, Oier Mees, Aviral Kumar, and Sergey Levine. Steering your generalists: Improving robotic foundation models via value guidance. *arXiv preprint arXiv:2410.13816*, 2024.

[41] Joseph Ortiz, Antoine Dedieu, Wolfgang Lehrach, J Swaroop Guntupalli, Carter Wendelken, Ahmad Humayun, Sivaramakrishnan Swaminathan, Guangyao Zhou, Miguel Lázaro-Gredilla, and Kevin P Murphy. Dmc-vb: A benchmark for representation learning for control with visual distractors. *Advances in Neural Information Processing Systems*, 37:6574–6602, 2024.

[42] Minting Pan, Xiangming Zhu, Yitao Zheng, Yunbo Wang, and Xiaokang Yang. Model-based reinforcement learning with isolated imaginations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):2788–2803, 2024.

[43] Vaishnav Potlapalli, Syed Waqas Zamir, Salman Khan, and Fahad Shahbaz Khan. PromptIR: Prompting for all-in-one blind image restoration. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023.

[44] Joan Puigcerver, Carlos Riquelme, Basil Mustafa, and Neil Houlsby. From sparse to soft mixtures of experts. *arXiv preprint arXiv:2308.00951*, 2023.

[45] Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. *arXiv preprint arXiv:2402.08191*, 2024.

[46] Yulin Ren, Xin Li, Bingchen Li, Xingrui Wang, Mengxi Guo, Shijie Zhao, Li Zhang, and Zhibo Chen. MoE-DiffIR: Task-customized diffusion priors for universal compressed image restoration. In *European Conference on Computer Vision (ECCV)*, volume 15067, pages 116–134, 2024.

[47] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In *Advances in Neural Information Processing Systems*, volume 34, pages 8583–8595, 2021.

[48] Jan Robine, Marc Hoftmann, Tobias Uelwer, and Stefan Harmeling. Transformer-based world models are happy with 100k interactions. *arXiv preprint arXiv:2303.07109*, 2023.

[49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, volume 9351, pages 234–241. Springer, 2015.

[50] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

[51] Chen Shi, Shaoshuai Shi, Kehua Sheng, Bo Zhang, and Li Jiang. DriveX: Omni scene modeling for learning generalizable world knowledge in autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 28599–28609, 2025.

[52] Wonil Song, Hyesong Choi, Kwanghoon Sohn, and Dongbo Min. A simple framework for generalization in visual RL under dynamic scene perturbations. volume 37, pages 121790–121826, 2024.

[53] 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.

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

[55] 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.

[56] Jiahui Wang, Chao Chen, Jiacheng Xu, Zongzhang Zhang, and Yang Yu. Focus-Then-Reuse: Fast adaptation in visual perturbation environments. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[57] Tao Wang, Kaihao Zhang, Ziqian Shao, Wenhan Luo, Bjorn Stenger, Tong Lu, Tae-Kyun Kim, Wei Liu, and Hongdong Li. GridFormer: Residual dense transformer with grid structure for image restoration in adverse weather conditions. *International journal of computer vision*, 132(10):4541–4563, 2024.

[58] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. DriveDreamer: Towards real-world-driven world models for autonomous driving. In *European Conference on Computer Vision (ECCV)*, pages 55–72. Springer, 2024.

[59] Ziyu Wang, Yanjie Ze, Yifei Sun, Zhecheng Yuan, and Huazhe Xu. Generalizable visual reinforcement learning with segment anything model. *arXiv preprint arXiv:2312.17116*, 2023.

[60] Bin Xia, Yulun Zhang, Shiyin Wang, Yitong Wang, Xinglong Wu, Yapeng Tian, Wenming Yang, and Luc Van Gool. DiffIR: Efficient diffusion model for image restoration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13095–13105, 2023.

[61] Jie Xiao, Xueyang Fu, Aiping Liu, Feng Wu, and Zheng-Jun Zha. Image de-raining transformer. *IEEE transactions on pattern analysis and machine intelligence*, 45(11):12978–12995, 2022.

[62] Guowei Xu, Ruijie Zheng, Yongyuan Liang, Xiyao Wang, Zhecheng Yuan, Tianying Ji, Yu Luo, Xiaoyu Liu, Jiaxin Yuan, Pu Hua, Shuzhen Li, Yanjie Ze, Hal Daume, Furong Huang, and Huazhe Xu. DrM: Mastering visual reinforcement learning through dormant ratio minimization. *arXiv preprint arXiv:2310.19668*, 2023.

[63] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023.

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

[65] Xinlei Yu, Zhangquan Chen, Yongbo He, Tianyu Fu, Cheng Yang, Chengming Xu, Yue Ma, Xiaobin Hu, Zhe Cao, Jie Xu, et al. The latent space: Foundation, evolution, mechanism, ability, and outlook. *arXiv preprint arXiv:2604.02029*, 2026.

[66] 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.

[67] 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.

[68] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5728–5739, 2022.

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

[70] Di Zhang, Bowen Lv, Hai Zhang, Feifan Yang, Junqiao Zhao, Hang Yu, Chang Huang, Hongtu Zhou, Chen Ye, et al. Focus On What Matters: Separated models for visual-based rl generalization. *Advances in Neural Information Processing Systems*, 37:116960–116986, 2024.

[71] Weipu Zhang, Gang Wang, Jian Sun, Yetian Yuan, and Gao Huang. STORM: Efficient stochastic transformer based world models for reinforcement learning. *Advances in Neural Information Processing Systems*, 36:27147–27166, 2023.

[72] Xu Zhang, Jiaqi Ma, Guoli Wang, Qian Zhang, Huan Zhang, and Lefei Zhang. Perceive-IR: Learning to perceive degradation better for all-in-one image restoration. *IEEE Transactions on Image Processing*, 2025.

[73] Yixian Zhang, Shu'ang Yu, Tonghe Zhang, Mo Guang, Haojia Hui, Kaiwen Long, Yu Wang, Chao Yu, and Wenbo Ding. SAC Flow: Sample-efficient reinforcement learning of flow-based policies via velocity-reparameterized sequential modeling. In *International Conference on Learning Representations (ICLR)*, 2026.

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

[75] Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. OccWorld: Learning a 3D occupancy world model for autonomous driving. In *European Conference on Computer Vision (ECCV)*, pages 55–72. Springer, 2024.

[76] Gaoyue Zhou, Haizhou Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

[77] Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, DitYan Yeung, and Chuang Gan. RoboDreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024.

[78] Jinguo Zhu, Xizhou Zhu, Wenhai Wang, Xiaohua Wang, Hongsheng Li, Xiaogang Wang, and Jifeng Dai. Uni-Perceiver-MoE: Learning sparse generalist models with conditional moes. *Advances in Neural Information Processing Systems*, 35:2664–2678, 2022.

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

Appendix Contents

A. VDCS: Full Degradation Operator Definitions
B. Theoretical Analysis and Proofs
C. ACO-MoE Architecture Details
D. Offline Dataset Generation
E. Training Settings and Computational Cost
F. Additional Experimental Results and Ablations

A. VDCS: Full Degradation Operator Definitions

This appendix provides the full VDCS benchmark, parameter settings, and mathematical definitions of the seven degradation operators used in Section 4. We follow the notation in Section 3: the clean frame is $o_t^{\text{raw}} = \mathcal{R}(s_t) \in [0,255]^{H\times W\times 3}$, and the corrupted frame is generated as $x_t^{\text{raw}} = \mathcal{D}_{k_t}(o_t^{\text{raw}}; \iota_t, \xi_t)$ (Eq. (1)). Operators are defined in the 8-bit space; normalization to $[-1,1]$ is done by Eq. (2). We use $\odot$ for element-wise multiplication (broadcast across channels).

A.1. Markov-Switching Settings and Severity Dynamics

Mode switching. Let $|\mathcal{K}|$ be the number of corruption types and $\mathcal{K} := \{1,\dots,|\mathcal{K}|\}$. The corruption mode $k_t \in \mathcal{K}$ follows a finite-state Markov chain with transition matrix $\Pi \in [0,1]^{|\mathcal{K}|\times|\mathcal{K}|}$, where $\Pi_{ij} = \Pr(k_t{=}j \mid k_{t-1}{=}i)$ and $\sum_{j=1}^{|\mathcal{K}|} \Pi_{ij} = 1$. We use a sticky chain with $\Pi_{ii} = p_s$ and $\Pi_{ij} = (1-p_s)/(|\mathcal{K}|-1)$ for $j \neq i$ (default $p_s = 0.8$).

Severity sampling. Each type $k \in \mathcal{K}$ has a base severity $\bar{\iota}_k \in [0,1]$ and a jitter band $\delta = 0.1$. When a mode is (re)selected (at $t=0$ or when $k_t \neq k_{t-1}$), we sample

$$\iota_t \sim \mathcal{U}\big[\iota_{k_t,\min},\ \iota_{k_t,\max}\big], \tag{8}$$

where $\mathcal{U}[\cdot]$ denotes the uniform distribution on an interval.

Temporal correlation within a segment. If the mode persists ($k_{t+1} = k_t$), severity follows a small random walk:

$$\iota_{t+1} = \text{clip}(\iota_t + \eta_t,\ \iota_{k_t,\min},\ \iota_{k_t,\max}), \qquad \eta_t \sim \mathcal{N}(0, 0.02^2), \tag{9}$$

with mode-specific bounds

$$\iota_{k,\min} = \max\big(0.1,\ (1-\delta)\bar{\iota}_k\big), \qquad \iota_{k,\max} = \min\big(1,\ (1+\delta)\bar{\iota}_k\big), \tag{10}$$

and $\text{clip}(u,a,b) = \min\{\max\{u,a\},b\}$.

A.2. VDCS Degradation Operators

Below we provide full operator definitions. In each case, the operator randomness (e.g., sampled masks, noise, kernel angle, JPEG quality) is absorbed into ξ_t in Eq. (1). We use $\text{clip}_{[0,255]}(u) = \min\{\max\{u,0\},255\}$ applied element-wise.

Table 3: VDCS physical degradations and base severities $\bar{\iota}_k$.

Degradation	Physical meaning	$\bar{\iota}_k$
Rain	Streak overlay (weather)	0.6
Haze	Atmospheric scattering / haze	0.6
Snow	Flake overlay (weather)	0.6
Motion blur	Directional blur kernel	0.35
Gaussian noise	Additive sensor noise	0.5
Low-light	Exposure reduction + noise	0.7
JPEG compression	Lossy block artifacts	0.7

Rain and snow (alpha blending). Rain overlays stochastic streaks and snow overlays stochastic flakes, producing a normalized weather mask $M_t \in [0,1]^{H\times W\times 1}$. For rain, draw $N_r = \lfloor 500\,\iota_t \rfloor$ line segments with length $\ell \in [\ell_{\min}, \ell_{\max}]$ (we use $\ell_{\min} = 3$, $\ell_{\max} = 10$) and angles near vertical, then blur to form M_t. For snow, draw $N_s = \lfloor 1000\,\iota_t \rfloor$ circular flakes with radius $r \in \{1,2\}$ and brightness $b \in [200, 255]$, then blur to form M_t. Both use alpha blending:

$$x_t^{\text{raw}} = (1 - \gamma M_t) \odot o_t^{\text{raw}} + \gamma M_t \odot c_w, \tag{11}$$

where $c_w = (255, 255, 255)$ is the weather color, and γ is opacity (we use $\gamma = 0.3$ for rain and $\gamma = 0.5$ for snow).

Haze (atmospheric scattering). Haze simulates atmospheric scattering with a vertical density profile. For row index $y \in \{0, \ldots, H-1\}$, define $\rho(y) = (y/H)^{1/2}$ and broadcast it to obtain $\rho_t \in [0,1]^{H\times W\times 1}$. Using haze color $c_f = (200, 200, 200)$ and scale $\alpha = 0.7\,\iota_t$, we render

$$x_t^{\text{raw}} = (1 - \alpha\rho_t) \odot o_t^{\text{raw}} + \alpha\rho_t \odot c_f. \tag{12}$$

Motion blur (directional convolution). Motion blur models camera/object motion as a directional convolution. Sample an angle $\theta \sim \mathcal{U}[0, 2\pi)$ and define a normalized line kernel $\mathbf{k}_{\theta,\ell}$ with length

$$\ell = \ell_{\min} + \iota_t(\ell_{\max} - \ell_{\min}), \qquad \ell_{\min} = 5,\ \ell_{\max} = 25. \tag{13}$$

Then

$$x_t^{\text{raw}} = \mathbf{k}_{\theta,\ell} * o_t^{\text{raw}}, \tag{14}$$

where $*$ denotes per-channel 2D convolution.

Gaussian noise (sensor noise). Gaussian noise injects pixel-wise sensor-level stochasticity. Sample $n_t \sim \mathcal{N}(0, \sigma^2 I)$ with

$$\sigma = \iota_t \sigma_{\max}, \qquad \sigma_{\max} = 25, \tag{15}$$

where I is the identity over pixel dimensions. Then

$$x_t^{\text{raw}} = \text{clip}_{[0,255]}(o_t^{\text{raw}} + n_t)\,. \tag{16}$$

Low-light (exposure reduction + noise). Low-light reduces exposure and amplifies sensor noise. Define a brightness factor

$$\nu = 1 - \iota_t(1 - \nu_{\min}), \qquad \nu_{\min} = 0.2 \tag{17}$$

and sample $n_t \sim \mathcal{N}(0, (\iota_t \sigma_\ell)^2 I)$ with $\sigma_\ell = 15$. Then

$$x_t^{\text{raw}} = \text{clip}_{[0,255]}(\nu\, o_t^{\text{raw}} + n_t)\,. \tag{18}$$

JPEG compression (lossy encoding). JPEG compression introduces block artifacts via encode–decode. Let q be the JPEG quality:

$$q = q_{\max} - \iota_t(q_{\max} - q_{\min}), \qquad q_{\min} = 10,\ q_{\max} = 90, \tag{19}$$

and denote by $\mathcal{J}_q(\cdot)$ the JPEG encode–decode operator at quality q. Then

$$x_t^{\text{raw}} = \mathcal{J}_q(o_t^{\text{raw}})\,. \tag{20}$$

B. Theoretical Analysis and Proofs

B.1. Formal Setup and Main Results

Setup. We use the random-variable notation introduced in Section 3: uppercase denotes random variables and lowercase their realizations. Let $\mathcal{X} := \{0, 1, \ldots, 255\}^{H \times W \times 3}$ be the finite 8-bit image alphabet. We assume the corruption process is *exogenous*: $K_t \perp S_t$, meaning the active corruption type (e.g., rain, haze, or camera-shake blur) is determined by external environmental conditions independently of the task state. This assumption holds by design in VDCS, where all corruptions are synthetic and scene-agnostic. In particular, object-motion blur, where the blur kernel depends on the agent's velocity and therefore on S_t, is excluded from the VDCS corruption set.[1] We further assume balanced modes with $p(K_t = k) = 1/|\mathcal{K}|$.

A (measurable) mode predictor is a function $\varphi : \mathcal{X} \to \mathcal{K}$. We assume *mode-identifiability*: for any state realization s,

$$P_e(s) \triangleq \inf_{\varphi} \Pr\big(\varphi(X_t) \neq K_t \mid S_t = s\big) \;<\; 1 - \tfrac{1}{|\mathcal{K}|}, \tag{21}$$

i.e., the Bayes error of predicting the active corruption from the corrupted observation is strictly better than random guessing.

Reconstruction contaminates representations. Intuitively, accurately reconstructing corrupted pixels forces the latent to encode corruption-specific patterns, not just task content. Formally, reconstruction-based world models learn an encoder $Z_t = f_\phi(X_t)$ and decoder $\hat{X}_t = g_\psi(Z_t)$ by minimizing a reconstruction distortion $\mathbb{E}[d(X_t, \hat{X}_t)] \leq \epsilon$, where $d : \mathcal{X} \times \mathcal{X} \to [0, 1]$ is a normalized distortion (e.g., per-pixel 8-bit error normalized to $[0, 1]$) and ϵ is a target distortion budget. The following proposition shows that accurate reconstruction *forces* the latent to retain corruption information, regardless of the encoder architecture.

Proposition B.1 (Representation Contamination). *Under (21), $K_t \perp S_t$, $p(K_t = k) = 1/|\mathcal{K}|$, and 8-bit observations, if $\mathbb{E}[d(X_t, \hat{X}_t)] \leq \epsilon$ with $\epsilon \in (0, \frac{1}{2}]$, then*

$$I(Z_t; K_t \mid S_t) \;\geq\; I(X_t; K_t \mid S_t) \;-\; C(\epsilon), \tag{22}$$

[1] VDCS motion-blur corruption is implemented as a synthetic global blur kernel with randomly chosen direction and severity, independent of any object's motion in the scene.

where $I(X_t; K_t \mid S_t)$ quantifies how much corruption-mode information is present in the input beyond the clean state (and is strictly positive by (21)), while $C(\epsilon) = \epsilon \log |\mathcal{K}| + h(\epsilon)$ is a reconstruction slack term that vanishes as $\epsilon \to 0$. Here $h(p) = -p \log p - (1-p)\log(1-p)$ is the binary entropy function (base-2 logs). Hence $I(Z_t; K_t \mid S_t) > 0$ for sufficiently small ϵ: ***any accurate reconstruction latent inevitably encodes the active corruption mode.***

The proof is in Appendix B.2. Proposition B.1 identifies a structural conflict: the smaller the reconstruction error ϵ, the smaller $C(\epsilon)$ and thus the tighter the lower bound in (22), i.e., *better reconstruction implies more corruption entanglement*.

Information-bottleneck (IB) objective and its connection to Proposition B.1. Let $Y_t \triangleq \arg\max_a \pi^\star(a \mid S_t)$ denote the optimal action label. Proposition B.1 shows reconstruction pushes $I(Z_t; K_t \mid S_t)$ *up*. The IB principle instead seeks to push nuisance dependence *down* by compressing X_t while preserving Y_t:

$$\min_{p(Z_t|X_t)} \; I(Z_t;\, X_t) \quad \text{s.t.} \quad I(Z_t;\, Y_t) \geq \beta. \tag{23}$$

For a deterministic encoder $Z_t = f_\phi(X_t)$, $H(Z_t \mid X_t) = 0$ and thus $I(Z_t; Y_t \mid X_t) = 0$; the chain rule yields

$$I(Z_t;\, X_t) \;=\; I(Z_t;\, Y_t) \;+\; I(Z_t;\, X_t \mid Y_t). \tag{24}$$

The residual $I(Z_t; X_t \mid Y_t)$ upper-bounds corruption entanglement. Since the encoder is deterministic, $H(Z_t \mid X_t, Y_t) = 0$, so $I(Z_t; X_t \mid Y_t) = H(Z_t \mid Y_t)$. Then, because $H(Z_t \mid K_t, Y_t) \geq 0$, we have

$$I(Z_t;\, X_t \mid Y_t) \;=\; H(Z_t \mid Y_t) \;\geq\; H(Z_t \mid Y_t) - H(Z_t \mid K_t, Y_t) \;=\; I(Z_t;\, K_t \mid Y_t) \;\geq\; 0. \tag{25}$$

Since $K_t \perp S_t$ and Y_t is a function of S_t, we have $K_t \perp Y_t$, so $I(Z_t; K_t \mid Y_t)$ measures corruption dependence without task confounding. Minimizing $I(Z_t; X_t)$ under the task constraint therefore minimizes an upper bound on $I(Z_t; K_t \mid Y_t)$, directly counteracting Proposition B.1.

Corollary B.2 (Foreground as an Approximate IB Anchor)**.** *Let F_t denote the ideal clean* foreground *observation (agent/task-relevant pixels) and B_t the corresponding background, so that O_t can be viewed as a foreground–background decomposition. Let Y_t be a discrete policy target (e.g., an optimal action label), and define $\eta \triangleq H(Y_t \mid F_t) \geq 0$. Assume $K_t \perp S_t$ and that F_t and Y_t are deterministic functions of S_t. Setting $Z_t = F_t$ achieves: 1)* ***Exact nuisance invariance:*** *$I(Z_t;\, K_t \mid Y_t) = 0$ (independent of η); 2)* ***Approximate task sufficiency:*** *$I(Z_t;\, Y_t) = H(Y_t) - \eta$.*

Specifically, $Z_t = F_t$ exactly eliminates corruption entanglement and meets the IB task constraint $I(Z_t; Y_t) \geq \beta$ with $\beta = H(Y_t) - \eta$. Empirically, foreground-only and full-background DreamerV3 training yield comparable average returns (769 vs. 767, Table 6), supporting $\eta \approx 0$ on this benchmark suite, though task-level variance exists. (Proof in Appendix B.2.)

From anchor to ACO-MoE. Physical degradations affect foreground pixels, so the ideal F_t is not directly observable from X_t. ACO-MoE approximates it via two input-adaptive steps implementing the IB surrogate in (24): (i) latent-expert RGB repair, which learns to map corrupted foreground appearance toward clean rendered targets without observing K_t or any expert–corruption label; and (ii) foreground compositing onto a black background, which removes variation in B_t that is independent of Y_t, suppressing $I(Z_t; X_t \mid Y_t)$ and tightening (25). Together, these steps push ACO-MoE's output toward the ideal IB anchor F_t.

B.2. Proofs of Theoretical Analysis

B.2.1. Notation

All logarithms are base-2. For a discrete random variable U with finite alphabet $\mathcal{U}$, $H(U) = -\sum_u p(u)\log p(u)$ denotes entropy. Mutual information is $I(U;V) = H(U) - H(U \mid V)$ and conditional mutual information is $I(U;V \mid W) = H(U \mid W) - H(U \mid V,W)$. The binary entropy function is

$$h(p) = -p\log p - (1-p)\log(1-p).$$

Random variables (fixed time index). We fix the time index and suppress the subscript t. Let S be the task state and let K be the corruption-mode random variable. Let K be the corruption-mode random variable taking values in $\mathcal{K} = \{1,\dots,|\mathcal{K}|\}$, where $|\mathcal{K}|$ denotes the number of corruption types (written as K when used as a scalar in the main text). We assume the corruption process is exogenous: $K \perp S$. Let $O = \mathcal{R}(S)$ be the clean observation (an 8-bit image). The corrupted observation is $X = \mathcal{D}_K(O)$, where $\mathcal{D}_k$ may be randomized (e.g., through the VDCS severity/randomness in Eq. (1), which we absorb into $\mathcal{D}_k$). The representation and reconstruction are $Z = f_\phi(X)$ and $\hat{X} = g_\psi(Z)$.

Mode-identifiability. Let $\mathcal{X}$ denote the finite 8-bit image alphabet of X. A mode predictor is a measurable function $\varphi : \mathcal{X} \to \mathcal{K}$. For every s in the support of S, the conditional Bayes error

$$P_e(s) \triangleq \inf_{\varphi} \Pr(\varphi(X) \neq K \mid S = s)$$

satisfies $P_e(s) < 1 - \frac{1}{|\mathcal{K}|}$.

Discrete 8-bit observations and distortion. X takes values in a finite alphabet (e.g., 8-bit images). Since the decoder output $g_\psi(Z)$ may be continuous-valued, we define the *quantized* reconstruction $\hat{X} \triangleq Q(g_\psi(Z))$, where $Q(\cdot)$ rounds each pixel/channel to the nearest 8-bit level in $\{0,\dots,255\}$. Thus both X and $\hat{X}$ are discrete 8-bit images on the same alphabet. We introduce a distortion measure $d : \mathcal{X} \times \mathcal{X} \to [0,1]$ that satisfies the mismatch property

$$\mathbf{1}\{X \neq \hat{X}\} \;\leq\; d(X,\hat{X}).$$

This property holds, for example, for normalized per-pixel ℓ_1 distortion $d(X,\hat{X}) = \frac{1}{HWC}\sum_i \frac{|X_i - \hat{X}_i|}{255}$ on 8-bit images; we keep d abstract in the analysis. For notational simplicity, we write $\hat{X} = g_\psi(Z)$ throughout, with the understanding that $\hat{X}$ denotes the quantized reconstruction.

Foreground decomposition and policy target. We decompose the clean observation as $O = (F,B)$, where F denotes the foreground (agent and task-relevant objects, treated as a discrete 8-bit image consistent with the above) and B the background. The optimal policy target is $Y \triangleq \arg\max_a \pi^\star(a \mid S)$, the greedy action label under the optimal policy; Y is a deterministic function of S, takes values in the finite action space $\mathcal{A}$, and its entropy $H(Y)$ is finite.

B.2.2. Proof of Proposition B.1

We fix the time index and suppress the subscript throughout. The proof shows that the Markov chain structure of reconstruction *preserves* corruption information, irrespective of the encoder architecture, so that a small reconstruction distortion forces $I(Z;K \mid S)$ to be large.

Proof. **Step 1: Data-processing inequality along the encoding chain.** *(Goal: transfer the problem from Z to $\hat{X}$, which is easier to connect to the reconstruction distortion.)*

Conditioned on $S = s$, the generative process yields the Markov chain

$$K \;\to\; X \;\to\; Z \;\to\; \hat{X},$$

because X is generated from $(K, S = s)$, $Z = f_\phi(X)$ is a deterministic function of X, and $\hat{X} = g_\psi(Z)$ is a deterministic function of Z (after quantization). Since $\hat{X}$ is a function of Z, the data-processing inequality (DPI) gives

$$I(Z; K \mid S = s) \;\geq\; I(\hat{X}; K \mid S = s).$$

Averaging over S yields

$$I(Z; K \mid S) \;\geq\; I(\hat{X}; K \mid S). \tag{26}$$

It remains to lower-bound $I(\hat{X}; K \mid S)$ in terms of $I(X; K \mid S)$ and the reconstruction quality ϵ.

Step 2: From reconstruction distortion to a mismatch probability. *(Goal: convert the distortion bound into an event amenable to information-theoretic reasoning.)*

Define $\epsilon_s \triangleq \mathbb{E}[d(X, \hat{X}) \mid S = s]$. By the mismatch property of the distortion (see Notation), $\mathbf{1}\{X \neq \hat{X}\} \leq d(X, \hat{X})$, so taking conditional expectations gives

$$q_s \;\triangleq\; \Pr(X \neq \hat{X} \mid S = s) \;\leq\; \epsilon_s.$$

Let $\bar{\epsilon}_s = \min\{\epsilon_s, \frac{1}{2}\}$. Since $q_s \leq \epsilon_s$ and h is non-decreasing on $[0, \frac{1}{2}]$, we have $h(q_s) \leq h(\bar{\epsilon}_s)$.

Step 3: Bounding the loss in mutual information via coupling. *(Goal: quantify how much $I(\hat{X}; K \mid S)$ can fall below $I(X; K \mid S)$ by using the reconstruction mismatch event as a handle.)*

Fix s and introduce the indicator $E = \mathbf{1}\{X \neq \hat{X}\}$, which satisfies $\Pr(E = 1 \mid S = s) = q_s$. We bound $I(X; K \mid S = s) - I(\hat{X}; K \mid S = s)$ directly:

$$\begin{aligned} I(X; K \mid S = s) - I(\hat{X}; K \mid S = s) &= H(K \mid \hat{X}, S = s) - H(K \mid X, S = s) \\ &\leq H(K \mid \hat{X}, S = s) - H(K \mid X, \hat{X}, S = s) \\ &= I(K;\, X \mid \hat{X}, S = s), \end{aligned} \tag{27}$$

where the inequality uses $H(K \mid X, \hat{X}, S = s) \leq H(K \mid X, S = s)$ (conditioning on $\hat{X}$ cannot increase entropy). We now bound $I(K; X \mid \hat{X}, S = s)$ by conditioning on E:

$$I(K;\, X \mid \hat{X}, S = s) = I(K;\, E \mid \hat{X}, S = s) + I(K;\, X \mid E, \hat{X}, S = s). \tag{28}$$

For the first term, $I(K; E \mid \hat{X}, S = s) \leq H(E) \leq h(q_s)$. For the second term: when $E = 0$ (i.e., $X = \hat{X}$), X is determined by $\hat{X}$, so $I(K; X \mid E = 0, \hat{X}, S = s) = 0$; when $E = 1$, $I(K; X \mid E = 1, \hat{X}, S = s) \leq H(K) \leq \log |\mathcal{K}|$. Therefore,

$$I(K;\, X \mid E, \hat{X}, S = s) = \Pr(E = 1 \mid S = s) \cdot I(K;\, X \mid E = 1, \hat{X}, S = s) \leq q_s \log |\mathcal{K}|.$$

Substituting into (28) and using $h(q_s) \leq h(\bar{\epsilon}_s)$ and $q_s \leq \bar{\epsilon}_s$:

$$I(X; K \mid S = s) - I(\hat{X}; K \mid S = s) \leq h(\bar{\epsilon}_s) + \bar{\epsilon}_s \log |\mathcal{K}|.$$

Step 4: Average over S and collect $C(\epsilon)$. *(Goal: convert the per-state bound into a single clean bound in terms of the global distortion ϵ.)*

Taking expectation over S:

$$I(\hat{X};K \mid S) \geq I(X;K \mid S) - \mathbb{E}[h(\bar{\epsilon}_S) + \bar{\epsilon}_S \log |\mathcal{K}|]\,.$$

Since h is concave on $[0,1]$, Jensen's inequality gives $\mathbb{E}[h(\bar{\epsilon}_S)] \leq h(\mathbb{E}[\bar{\epsilon}_S])$. Moreover, $\mathbb{E}[\bar{\epsilon}_S] \leq \mathbb{E}[\epsilon_S] \leq \epsilon \leq \frac{1}{2}$, so $h(\mathbb{E}[\bar{\epsilon}_S]) \leq h(\epsilon)$ (as h is non-decreasing on $[0, \frac{1}{2}]$), and $\mathbb{E}[\bar{\epsilon}_S] \leq \epsilon$. Therefore

$$I(\hat{X};K \mid S) \geq I(X;K \mid S) - \big(\epsilon \log |\mathcal{K}| + h(\epsilon)\big).$$

Combining with (26) establishes

$$I(Z;K \mid S) \geq I(X;K \mid S) - C(\epsilon), \qquad C(\epsilon) = \epsilon \log |\mathcal{K}| + h(\epsilon).$$

Step 5: Positivity of $I(X;K \mid S)$. *(Goal: confirm the lower bound is non-trivially positive, completing the proof.)*

Since $K \perp S$ and $p(K=k) = 1/|\mathcal{K}|$ (balanced modes), we have $H(K \mid S=s) = H(K) \leq \log |\mathcal{K}|$ for every s, and in fact $H(K) = \log |\mathcal{K}|$ under the uniform prior. By the mode-identifiability assumption, $P_e(s) < 1 - \frac{1}{|\mathcal{K}|}$ for every s. Fano's inequality applied conditionally on $S=s$ then gives

$$I(X;K \mid S=s) \geq \log |\mathcal{K}| - h(P_e(s)) - P_e(s) \log(|\mathcal{K}| - 1) \;>\; 0,$$

where positivity follows because the right side equals 0 exactly at $P_e(s) = 1 - \frac{1}{|\mathcal{K}|}$ (by direct substitution) and is strictly decreasing in $P_e(s)$ for $P_e(s) \in [0, 1 - \frac{1}{|\mathcal{K}|}]$. Averaging over S yields $I(X;K \mid S) > 0$. □

□

Remark. The bound $C(\epsilon) = \epsilon \log |\mathcal{K}| + h(\epsilon)$ satisfies $C(\epsilon) \to 0$ as $\epsilon \to 0$, so better reconstruction *tightens* the lower bound in (22). This is the key structural conflict: **accurate reconstruction necessarily drives the latent to encode the corruption mode.**

B.2.3. Proof of Corollary B.2

Proof. Fix the time index and suppress subscript. Let $Y \triangleq \arg\max_a \pi^\star(a \mid S)$ be the optimal action label (a deterministic function of S, consistent with the main text), $\eta \triangleq H(Y \mid F) \geq 0$, and assume $K \perp S$. We treat F as a discrete 8-bit image (finite alphabet), consistent with Appendix B.2.1. We write $F = h_1(S)$ and $Y = h_2(S)$ for deterministic functions h_1, h_2; determinism of Y follows from the arg max definition.

Step 1: Exact nuisance invariance. Since $(F,Y) = (h_1(S), h_2(S))$ is a deterministic function of S and $K \perp S$, we have

$$p(k,f,y) = p(k)\,p(f,y) \quad \forall k,f,y,$$

i.e., $K \perp (F,Y)$ jointly. Therefore, we have

$$p(k \mid y) = \frac{\sum_f p(k,f,y)}{p(y)} = \frac{p(k)\sum_f p(f,y)}{p(y)} = p(k), \quad p(k \mid f,y) = \frac{p(k,f,y)}{p(f,y)} = p(k),$$

so $p(k \mid f,y) = p(k \mid y)$ for all k,f,y, which is exactly $K \perp F \mid Y$. Hence $I(F;K \mid Y) = 0$. Setting $Z = F$ gives $I(Z;K \mid Y) = 0$.

Step 2: Approximate task sufficiency. By definition of mutual information and $\eta = H(Y \mid F)$:

$$I(Z;Y) = I(F;Y) = H(Y) - H(Y \mid F) = H(Y) - \eta.$$

Thus $Z = F$ meets the IB task constraint $I(Z;Y) \geq \beta$ with $\beta = H(Y) - \eta$. When $\eta \approx 0$, task sufficiency is approximately attained. □

Remark. The two results are asymmetric in their requirements: nuisance invariance is exact under $K \perp S$ alone, whereas task sufficiency depends on η. We treat the empirical return parity between FG-only and full-background DreamerV3 (Table 6) as a proxy for small η on average, while task-level variance indicates η can be non-negligible for individual tasks. □

C. ACO-MoE Architecture Details

This section documents the complete ACO-MoE architecture, including the dual-stream MoE U-Net and detailed per-expert decoder/router configurations used in our implementation.

C.1. Dual-Stream MoE U-Net

Our ACO-MoE preprocessor is implemented as a dual-stream Mixture-of-Experts U-Net with a shared encoder, a bottleneck router, and per-expert RGB/mask decoders. The encoder uses four downsampling stages (depth=4) with *DoubleConv* blocks: two 3×3 convolutions with BatchNorm and SiLU. The router applies global average pooling to the bottleneck feature, followed by a two-layer MLP that outputs routing logits over N_e latent repair experts. We use N_e only as a capacity hyperparameter rather than as the number of corruption modes. Expert indices are latent: no degradation or perturbation labels are provided to the router, no minibatch contains an expert target, and no loss term supervises expert–corruption assignment. Each expert contains two decoders: (i) an RGB decoder that predicts a 3-channel residual added to the degraded input (followed by tanh and clamping to $[-1, 1]$), and (ii) a mask decoder that outputs 2-channel logits for foreground/background. The foreground probability is obtained by softmax and used to compose an agent-centric observation with a black background (-1).

When `task_conditioned_mask` is enabled, mask decoders share the upsampling trunk and only the final 1×1 head is task-specific; otherwise each expert predicts its own 2-channel mask logits. An optional `special_rgb_expert` can be loaded from a standalone U-Net checkpoint for a specific degradation (e.g., motion blur) and kept frozen during MoE training.

Expert aggregation. Routing logits are converted to probabilities with softmax. In the default training path we mix expert outputs using soft routing weights. At inference we optionally use hard top-1 routing for efficiency. The final agent-centric observation is computed as $\hat{o}_t \odot m_t + (-1)(1 - m_t)$, where m_t is the foreground probability.

C.2. Per-Expert Decoder and Router Specifications

We provide detailed channel configurations for the per-expert decoders and the routing network, complementing the summary in Table 4.

Each expert is a U-Net with a shared encoder and two decoder branches. The encoder has four downsampling blocks with channels $[64, 128, 256, 512]$; each block applies Conv→GroupNorm→SiLU→Conv→GroupNorm→SiLU

Table 4: ACO-MoE U-Net architecture (default base channel C=9). $H \times W$ is the input resolution; depth= 4 requires H, W divisible by 16. DoubleConv denotes 3×3 Conv→BN→SiLU→ 3×3 Conv→BN→SiLU.

Block	Stage	Configuration	Output size
Shared encoder	Input	RGB in $[-1, 1]$	$H \times W \times 3$
	Enc-1	DoubleConv $3 \to C$	$H \times W \times C$
	Enc-2	MaxPool 2×2 + DoubleConv $C \to 2C$	$\frac{H}{2} \times \frac{W}{2} \times 2C$
	Enc-3	MaxPool 2×2 + DoubleConv $2C \to 4C$	$\frac{H}{4} \times \frac{W}{4} \times 4C$
	Enc-4	MaxPool 2×2 + DoubleConv $4C \to 8C$	$\frac{H}{8} \times \frac{W}{8} \times 8C$
	Bottleneck	MaxPool 2×2 + DoubleConv $8C \to 16C$	$\frac{H}{16} \times \frac{W}{16} \times 16C$
Router	GAP	AdaptiveAvgPool to 1×1	$1 \times 1 \times 16C$
	FC-1	MLP $16C \to 256$ with SiLU	256
	FC-2	MLP $256 \to N_e$ (routing logits over latent experts)	N_e
Decoder trunk (per expert)	Up-1	Upsample ×2 + concat (skip Enc-4) + DoubleConv $(16C{+}8C) \to 8C$	$\frac{H}{8} \times \frac{W}{8} \times 8C$
	Up-2	Upsample ×2 + concat (skip Enc-3) + DoubleConv $(8C{+}4C) \to 4C$	$\frac{H}{4} \times \frac{W}{4} \times 4C$
	Up-3	Upsample ×2 + concat (skip Enc-2) + DoubleConv $(4C{+}2C) \to 2C$	$\frac{H}{2} \times \frac{W}{2} \times 2C$
	Up-4	Upsample ×2 + concat (skip Enc-1) + DoubleConv $(2C{+}C) \to C$	$H \times W \times C$
RGB head (per expert)	Out	1×1 Conv $C \to 3$ (residual)	$H \times W \times 3$
	Residual add	$\hat{o}_t = \mathrm{clip}(x_t + \tanh(\Delta_t), [-1, 1])$	$H \times W \times 3$
Mask head (per expert)	Out	1×1 Conv $C \to 2$ (logits) + softmax	$H \times W \times 2$
	Foreground prob.	$m = \mathrm{softmax}(\cdot)[\mathrm{fg}]$	$H \times W \times 1$

followed by MaxPool, taking an $84 \times 84 \times 3$ input normalized to $[-1, 1]$. The RGB decoder uses four upsampling blocks with skip connections and predicts a residual Δ_t, yielding restored output $\hat{o}_t = \mathrm{clip}(x_t + \tanh(\Delta_t), [-1, 1])$, producing an $84 \times 84 \times 3$ output. The mask decoder mirrors the upsampling structure and predicts 2-channel logits, which are converted to a foreground probability via softmax; this yields an $84 \times 84 \times 2$ logits tensor and a corresponding $84 \times 84 \times 1$ foreground map. The router consists of four strided conv layers with channels $[32, 64, 128, 256]$ and stride 2, global average pooling, and a two-layer MLP $256 \to 256 \to N_e$, followed by softmax routing probabilities over latent experts. In our default implementation, we use $N_e = 9$ as a fixed capacity choice; this value is not used as a corruption-mode label space.

D. Offline Dataset Generation

D.1. Setup for Dataset Generation

ACO-MoE is pretrained entirely offline on a paired dataset before any RL training begins. This section describes the dataset construction pipeline, which couples VDCS physical degradations with DMC-GB background distractions. The training tuple contains only (corrupted image, clean image, agent-only image, foreground mask), which supervise the three loss terms in Eq. 7. The corruption-mode index k_t is stored only as HDF5 metadata for auditing and reproducibility; it is never returned in training minibatches, never fed to the router, and never used as an expert or degradation label.

We construct a unified dataset over DMControl tasks by balancing samples across tasks and perturbation-generation settings. For each pair, we render clean frames o_t and collect a parallel set of observations under a uniform background setting. Foreground masks m_t are then obtained by color-threshold segmentation: pixels deviating from the background appearance beyond a fixed tolerance are labeled foreground, covering the agent and task-relevant objects including goal sites. The tolerance is set per-task to ensure robust separation across varied scene appearances.

For VDCS physical degradations, we apply stochastic intensity jitter within a $\pm 10\%$ band around a base

Algorithm 1 Unified DMC Dataset Generation

Input: tasks $\mathcal{T}$, perturbation generators $\mathcal{K}$, samples N, size (H, W), train ratio ρ, background path $\mathcal{B}$, seed s

for each task $\tau \in \mathcal{T}$ **do**
  infer action_repeat from benchmark map or policy logs
  if policy logdir is provided **then**
    load Dreamer policy for τ
  end if
  for each degradation $k \in \mathcal{K}$ **do**
    initialize clean env and/or DMC-GB env for (τ, k)
    for $i = 1$ to N **do**
      render clean image o_t and uniform-background image o_t^{bg}
      build mask m_t via color-threshold segmentation on o_t^{bg}
      $o_t^{\text{agent}} \leftarrow o_t \odot m_t$
      if k is VDCS **then**
        apply degradation $\mathcal{D}_k$ with jittered intensity
      end if
      if k is DMC-GB **then**
        render corrupted frame from DMC-GB env and sync physics
      end if
      write H5 training tuple (degraded, clean, agent_only, mask); store k only as audit metadata and never return it to the training loader
      assign to train/val split with probability ρ
      step env with random action or Dreamer policy
    end for
  end for
end for

severity to encourage robustness to severity variation. For DMC-GB background distractions, we render dynamic backgrounds (e.g., natural video clips) or palette-based color shifts to induce appearance changes that are independent of the agent dynamics. We use a 0.9/0.1 train/validation split. In our main experiments, images are resized to 84×84, and we generate 5,000 samples per task, with the task list and degradation set matching the settings described in the main text. Algorithm 1 summarizes the complete data generation pipeline.

D.2. Toward Real-World Data Generation without Manual Foreground Labels

Figure 7 illustrates a practical path for extending ACO-MoE beyond simulation without manual foreground annotation. In our current setting, simulation provides clean frames and foreground masks automatically, which are used only for offline pretraining. At deployment time, ACO-MoE takes only corrupted RGB as input and does not require clean references, foreground masks, or corruption labels.

For real-world systems, the same supervision can be approximated from two automatic sources. First, pseudo foreground masks can be generated from robot geometry, motion cues, or offline segmentation/tracking models. Second, corrupted–clean pseudo pairs can be constructed from nominal-condition real frames or repeated trajectories, optionally combined with synthetic degradations or temporal alignment. As shown in Fig. 7, these two sources jointly provide supervision for offline ACO-MoE pretraining, while deployment still uses corrupted RGB only. This suggests that the method is not inherently tied to simulator-only labels, although validating such a real-world data-generation pipeline on physical robotic platforms remains future

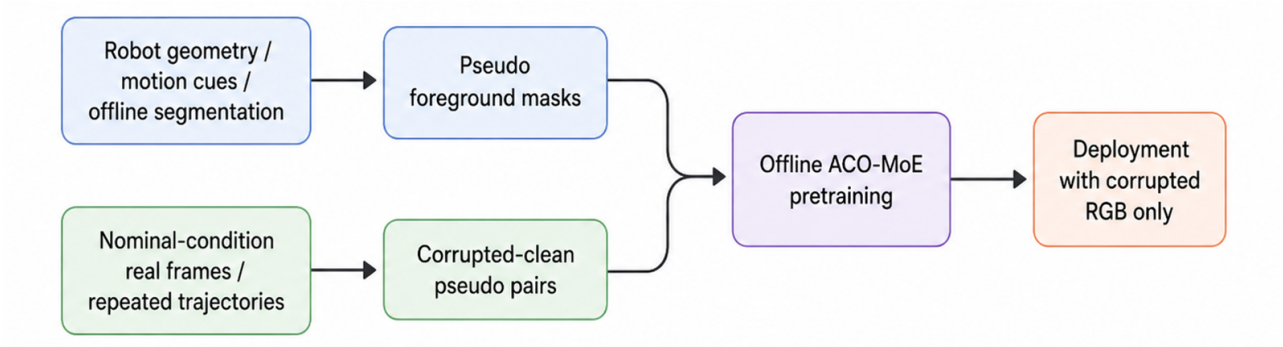


Figure 7: Conceptual pathway for extending ACO-MoE beyond simulation without manual foreground labels. Pseudo foreground masks can be obtained automatically from robot geometry, motion cues, or offline segmentation, while corrupted–clean pseudo pairs can be constructed from nominal-condition real frames or repeated trajectories. Both are used only for offline pretraining; deployment still requires corrupted RGB only.

work.

E. Training Settings and Computational Cost

This section summarizes the MoE pretraining procedure, the frozen inference settings, and the hyperparameter choices and computational costs used in our experiments.

E.1. MoE Pretraining

We pretrain the dual-stream MoE U-Net on the unified offline HDF5 dataset. Each training sample provides only a degraded input x_t, the clean rendered target o_t, the clean-background target o_t^{cb}, and a pseudo foreground mask m_t^{ps}; no degradation identity k_t or perturbation label is provided. Inputs are normalized to $[-1, 1]$ and augmented with synchronized random horizontal flips to preserve mask–image alignment. The router applies global average pooling to the encoder bottleneck and produces routing weights π_t via a two-layer MLP. These weights soft-mix the per-expert RGB residuals and mask logits at every step, and the router and all expert decoders are updated only through the three observation-adaptation losses ($\mathcal{L}_{\text{rgb}}$, $\mathcal{L}_{\text{mask}}$, $\mathcal{L}_{\text{final}}$ of Eq. 7), without any routing-label or expert-assignment supervision. For video-background distraction modes, we down-weight $\mathcal{L}_{\text{rgb}}$ to avoid incentivizing unnecessary background inpainting. We use AdamW with the hyperparameters reported in Section E.3. Algorithm 2 outlines the training loop.

E.2. Frozen Inference Settings

At evaluation time, we freeze the pretrained MoE and use hard top-1 routing for efficiency. For each environment step, the corrupted observation is first processed by the frozen MoE to produce an agent-centric observation, which is then fed into the downstream RL backbone (model-free or model-based). We report mean±std returns across evaluation seeds/episodes under both VDCS (Markov-temporal) and background-distraction settings, and optionally record visualization panels showing the degraded input, restored output, and agent-centric observation. Algorithm 3 details the frozen inference procedure.

Algorithm 2 Dual-Stream MoE Pretraining

Input: paired dataset $\mathcal{D} = \{(x_t, o_t, o_t^{\text{cb}}, m_t^{\text{ps}})\}$ without degradation labels, number of experts N_e, loss weights $\lambda_{\text{rgb}}, \lambda_{\text{mask}}, \lambda_{\text{final}}$

Initialize a dual-stream MoE U-Net with a shared encoder, a router, and per-expert RGB/mask decoders
Initialize AdamW optimizer
for each training step **do**
  Sample a minibatch $(x_t, o_t, o_t^{\text{cb}}, m_t^{\text{ps}}) \sim \mathcal{D}$
  Encode x_t to bottleneck feature u_t and compute router weights $\pi_t = \text{softmax}(\text{MLP}(\text{GAP}(u_t)))$
  Aggregate per-expert outputs: $\Delta_t = \sum_{j=1}^{N_e} \pi_{t,j} \Delta_t^{(j)}, \quad \ell_t^{\text{mask}} = \sum_{j=1}^{N_e} \pi_{t,j} \ell_t^{\text{mask},(j)}$
  Compute restored RGB $\hat{o}_t = \text{clip}_{[-1,1]}(x_t + \tanh(\Delta_t))$, foreground probability $m_t = \text{softmax}(\ell_t^{\text{mask}})_{[\text{fg}]}$, and agent-centric output $\hat{o}_t^{\text{agent}} = \hat{o}_t \odot m_t + b \odot (1 - m_t)$
  $\mathcal{L}_{\text{rgb}} \leftarrow \|\hat{o}_t - o_t\|_1$
  $\mathcal{L}_{\text{mask}} \leftarrow \text{CE}(\ell_t^{\text{mask}}, m_t^{\text{ps}})$
  $\mathcal{L}_{\text{final}} \leftarrow \|\hat{o}_t^{\text{agent}} - o_t^{\text{cb}}\|_1$
  $\mathcal{L} \leftarrow \lambda_{\text{rgb}} \mathcal{L}_{\text{rgb}} + \lambda_{\text{mask}} \mathcal{L}_{\text{mask}} + \lambda_{\text{final}} \mathcal{L}_{\text{final}}$
  Optionally rescale $\mathcal{L}_{\text{rgb}}$ for video-distraction modes
  Update all parameters by backpropagation; periodically validate and save checkpoints
end for

Algorithm 3 Evaluation with Frozen ACO-MoE

Input: env $\mathcal{E}$ with VDCS/DMC-GB, pretrained MoE G, RL agent π

for each episode **do**
  reset env, initialize agent state
  for each step t **do**
    observe corrupted frame x_t
    compute $(\hat{o}_t, m_t) = G(x_t)$ with top-1 routing
    compose agent-centric frame $o_t^{\text{agent}} = \hat{o}_t \odot m_t + (-1)(1 - m_t)$
    action $a_t \leftarrow \pi(o_t^{\text{agent}})$
    step env and accumulate return
  end for
end for

E.3. Hyperparameter Configuration

We report all hyperparameters for MoE pretraining, router training, and downstream RL training to support reproducibility.

MoE pretraining uses AdamW with learning rate 10^{-4}, weight decay 10^{-4}, and batch size 1024, for 15,000 iterations over the unified paired dataset. The total loss combines RGB restoration (ℓ_1), mask segmentation (2-class cross-entropy against the pseudo masks m_t^{ps}), and agent-centric reconstruction (ℓ_1 against o_t^{cb}), with weights λ_{rgb}:λ_{mask}:λ_{final} = 1:1:1. The shared encoder, router MLP, and all N_e per-expert decoders are optimized jointly under this single objective, with no auxiliary degradation-classification loss or expert-label supervision. For DreamerV3, the RSSM uses 1024 deterministic units with 32×32 categorical stochastic units, imagination horizon $T = 15$, discount $\gamma = 0.997$, batch size 16 sequences of 64 timesteps, and 1M total environment steps. For frozen-policy RL evaluation, DMControl (DreamerV3) uses `steps=1e6`, `action_repeat=2`, input size 84×84, and episode length 1000. RoboSuite (TD-MPC2) uses `image_size=168`, `frame_stack=3`, `action_repeat=2`, and episode length 200. Both RL backbones

Table 5: Latency decomposition (ms/step). RTX 5090, 84×84, batch size 1 (mean±std). Preprocess modules: ACO-MoE restorer (4.90M params) vs. SAM2-tiny (38.95M params). For FTR, preprocessing is amortized with segmentation invoked once every T_{sel}=20 steps and tracking otherwise: $\frac{1}{20}t_{\text{segment}} + \frac{19}{20}t_{\text{track}}$.

System	RL policy	Preprocess	Total
ACO-MoE (ours)	2.36±0.003	20.54±0.93	**22.90±0.93**
FTR [56]	2.76±0.34	73.48±1.21	**76.24±1.25**

require approximately 24 hours for each task/seed on a single NVIDIA GeForce RTX 5090.

E.4. Computational Overhead

All latency measurements are conducted on a single NVIDIA GeForce RTX 5090 with batch size 1 and 84×84 RGB inputs. We report a decomposition into *RL policy* (one policy forward per environment step), *Preprocess* (image preprocessing), and their *Total* runtime. Results are averaged over multiple trials after warm-up.

Table 5 shows that the main runtime gap comes from preprocessing. For ACO-MoE, the restorer contributes 20.54 ± 0.93 ms/step, and total latency is 22.90 ± 0.93 ms/step with one policy forward. For FTR, the low-level RL policy is DrQ-v2-based and relatively lightweight (2.76 ± 0.34 ms/step), while SAM2-tiny preprocessing dominates: $t_{\text{segment}} = 85.77 \pm 3.64$ ms, $t_{\text{track}} = 72.84 \pm 1.17$ ms, yielding an amortized 73.48 ± 1.21 ms/step at $T_{\text{sel}} = 20$. Hence, FTR's end-to-end latency is 76.24 ± 1.25 ms/step, substantially higher than ACO-MoE.

E.5. Existing Assets and Licenses

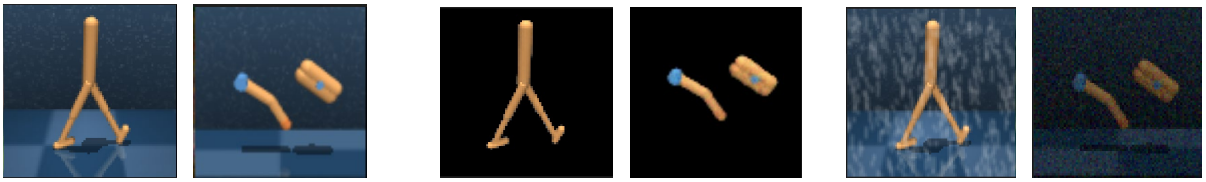
We use existing public assets only for non-commercial academic research and evaluation. The simulation benchmarks, RL backbones, and image-restoration baselines are credited through citations to their original papers and repositories. DMControl is released under Apache-2.0. DreamerV3, TD-MPC2, Restormer, AdaIR, and NAFNet are released under open-source licenses stated in their repositories. PromptIR is used under its academic/non-commercial research terms. We do not redistribute third-party datasets, models, or code beyond their stated terms of use.

F. Additional Experimental Results and Ablations

This section reports complete per-task results omitted from the main paper due to space constraints. Section F.1 analyzes ACO-MoE recovery vs. DreamerV3 clean baselines. Section F.2 covers DMC-GB generalization. Section F.3 provides training curves. Section F.4 validates backbone-agnostic transferability via DrQ-v2. Section F.5 presents full ablation results.

F.1. Clean Baseline Recovery Analysis

To contextualize ACO-MoE's performance under VDCS Markov-temporal perturbations, Table 6 reports DreamerV3 trained on full-background observations, DreamerV3 trained on foreground-only observations, ACO-MoE under VDCS corruption, and the resulting recovery rate per task. To empirically validate the assumption $H(Y_t \mid F_t) \approx 0$, we train DreamerV3 on full-background clean observations and obtain an average score of 767, compared to 769 for foreground-only training—a difference of 0.3%—confirming that background cues carry negligible policy-relevant information in these benchmarks (Figure 8(a) vs. (b)). ACO-MoE recovers 95.3% of clean foreground-only performance on average, with simple locomotion tasks (`cartpole_swingup`, `walker_walk`) reaching >98% and the hardest case (`finger_spin`) at 90.6%.



(a) Full BG (b) Clean (FG-Only) (c) VDCS

Figure 8: Three observation settings: (a) **Full BG**: standard DMC rendering; (b) **Clean (FG-only)**: black-masked background; (c) **VDCS**: FG-only with Markov-temporal perturbations.

Table 6: Clean baselines vs. ACO-MoE (VDCS). Recovery = ACO-MoE / FG-only $\times 100$. Full BG vs. FG-only differ by 0.3%, confirming $H(Y_t \mid F_t) \approx 0$.

Task	DreamerV3 (Full BG)	DreamerV3 (FG-only)	ACO-MoE (VDCS)	Recovery (%)
`cartpole, swingup`	852	878	864	98.5
`finger, spin`	558	543	492	90.6
`finger, turn_hard`	793	929	882	95.0
`hopper, stand`	952	948	910	96.0
`hopper, hop`	388	359	339	94.5
`cheetah, run`	848	749	700	93.5
`walker, walk`	963	974	963	98.8
`walker, run`	782	772	738	95.5
Average	**767**	**769**	**736**	**95.3**

F.2. DMC-GB Generalization: Per-Task Results

Tables 7 and 8 report full per-task performance on DMC-GB `video_hard` and `color_hard` settings. ACO-MoE achieves state-of-the-art averages across both settings, with particularly strong gains on `cheetah_run` (+71% video-hard, +167% color-hard) and `walker_walk` (+4% and +30%), demonstrating that foreground extraction generalizes robustly across background-distraction regimes. The `finger_spin` underperformance is consistent with the backbone limitation identified in Section 6.2.

Table 7: Performance comparison on Video Hard perturbations. Results show mean ± std over evaluation episodes. Red: best performance; Blue: second best.

Task	PAD	Q^2	SimGRL	FTR	DrQ	SODA	SVEA	SRM	SGQN	SMG	ACO-MoE (Ours)	△
cartpole, swingup	79 ±9	807 ±84	579 ±306	646 ±169	168 ±35	346 ±59	510 ±177	254 ±69	599 ±112	764 ±32	860 ±7	+53 7%
finger, spin	1 ±1	696 ±122	352 ±274	904 ±71	54 ±44	310 ±72	353 ±71	131 ±89	710 ±159	910 ±61	497 ±19	−413 −45%
walker, stand	802 ±41	476 ±45	932 ±17	912 ±51	278 ±79	406 ±68	814 ±57	558 ±139	870 ±78	955 ±9	990 ±11	+35 4%
walker, walk	26 ±11	325 ±62	734 ±42	900 ±45	110 ±33	175 ±31	348 ±80	165 ±99	634 ±136	814 ±51	936 ±52	+36 4%
cheetah, run	7 ±7	222 ±50	221 ±111	446 ±65	38 ±26	118 ±40	105 ±13	87 ±24	135 ±44	303 ±46	763 ±43	+317 71%
Average	183.0 ±347.4	505.2 ±245.5	563.6 ±286.0	761.6 ±209.2	129.6 ±97.5	271.0 ±120.4	426.0 ±260.8	239.0 ±188.6	589.6 ±274.7	749.2 ±260.6	809.2 ±194.3	+47.6 6%

Table 8: Performance comparison on Color Hard perturbations. Results show mean ± std over evaluation episodes. Red: best performance; Blue: second best.

Task	SAC	DrQ	SODA	SVEA	SRM	SGQN	SMG	**ACO-MoE (Ours)**	△
cartpole, swingup	184 ±26	717 ±133	585 ±66	752 ±86	752 ±103	636 ±110	726 ±62	845 ±33	+93 12%
finger, spin	271 ±23	655 ±214	663 ±106	868 ±74	834 ±90	700 ±219	841 ±113	497 ±11	−371 −43%
walker, stand	526 ±259	769 ±182	719 ±138	799 ±118	807 ±128	788 ±114	878 ±70	990 ±11	+112 13%
walker, walk	379 ±37	456 ±192	396 ±78	571 ±134	483 ±123	632 ±176	739 ±31	958 ±37	+219 30%
cheetah, run	208 ±54	147 ±80	199 ±38	238 ±69	203 ±30	210 ±18	299 ±22	798 ±25	+499 167%
Average	313.6 ±140.7	548.8 ±254.0	512.4 ±213.5	645.6 ±253.0	615.8 ±269.6	593.2 ±223.3	696.6 ±231.6	817.6 ±195.8	+121.0 17%

F.3. Training Curves and Qualitative Visualizations

Figure 9 shows training curves for all eight DMControl tasks under VDCS Markov-switching over 1M steps. Figure 10 reports RoboSuite results alongside qualitative restoration visualizations. Figures 11 and 12 provide comprehensive visual examples of each degradation type across tasks and environments.

F.4. Backbone Generalization Analysis

To verify that ACO-MoE's gains are not specific to the DreamerV3 backbone, we replace DreamerV3 with DrQ-v2 [64] and evaluate the same frozen ACO-MoE preprocessing module without any modification. Figure 13 reports results on `finger_spin` and the 8-task average under VDCS Markov-temporal perturbations.

On `finger_spin`, DrQ-v2 achieves a VDCS score of 912, far exceeding DreamerV3's ceiling of 543—confirming that the `finger_spin` gap reported in Section 6.2 stems from DreamerV3's known limitation on fine-grained contact tasks rather than from preprocessing quality. With ACO-MoE, DrQ-v2 scores 872 (95.6% recovery), matching the recovery rates observed with DreamerV3 (95.3% average). Averaged over all 8 tasks, ACO-MoE recovers 91.8% of DrQ-v2's performance, demonstrating that the frozen preprocessing module transfers across RL algorithms without modification.

F.5. Additional Ablation Studies

This section reports additional ablations for ACO-MoE. Figure 14 compares frozen ACO-MoE with no-preprocessing and freeze-then-adapt variants on VDCS Markov-temporal perturbations. Tables 9 and 10 evaluate component ablations under DMC-GB video-hard and color-hard settings, while Table 11 reports VDCS static single-degradation results. Figure 15 visualizes dual-stream and single-stream outputs.

F.6. Comparison with Generic Image Restoration Methods

We compare ACO-MoE with four all-in-one image restoration models used as frozen preprocessors before DreamerV3: PromptIR [43], Restormer [68], AdaIR [7], and NAFNet [4]. Each method maps the corrupted RGB frame to a restored frame, with no foreground composition, task-aware tuning, or RL retraining.

Table 12 shows that ACO-MoE achieves an average return of 766.9, outperforming the best generic baseline, AdaIR, by +220.3 points (+40.3%). The gains are largest on tasks requiring accurate foreground localization

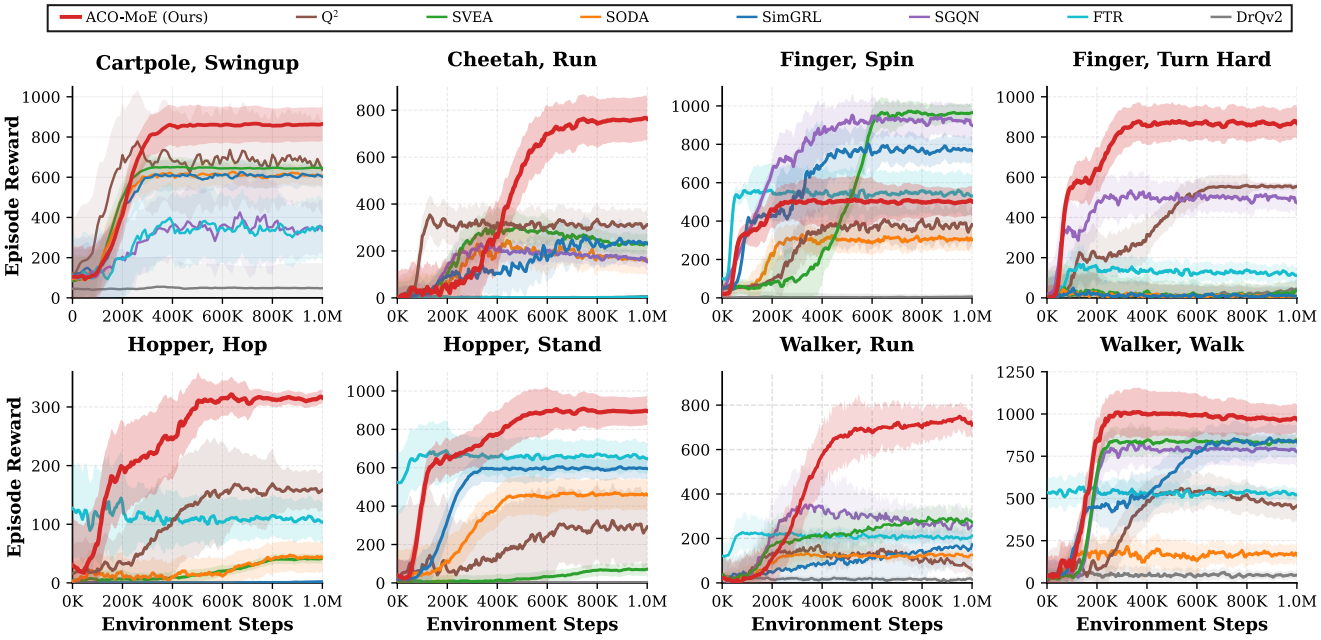


Figure 9: Training curves on DeepMind Control Suite with VDCS Markov-switching perturbations over 1M environment steps. Each curve shows the mean episode reward across 5 evaluation seeds (10 episodes each), with shaded regions indicating ± 1 standard deviation. Among all methods, FTR is a test-time adaptation approach, while the others are trained end-to-end. ACO-MoE consistently achieves higher asymptotic performance and faster convergence on most tasks.

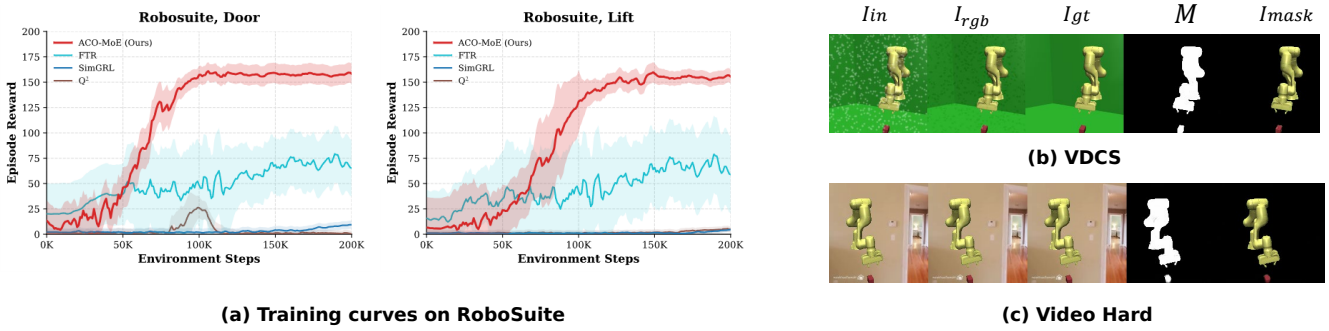
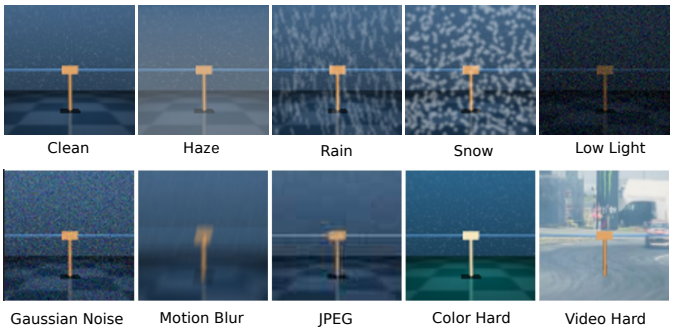
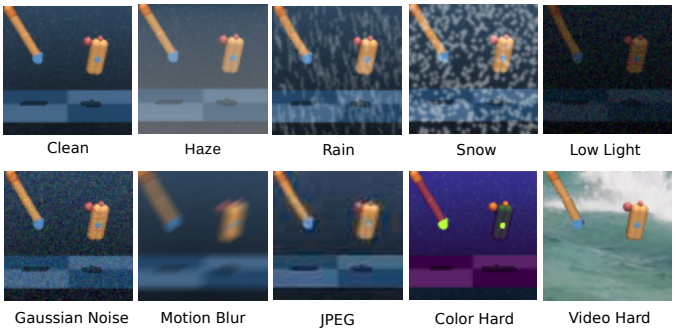


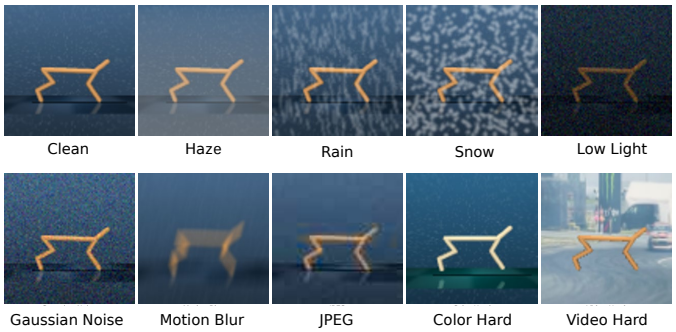
Figure 10: **Quantitative performance and qualitative restoration on RoboSuite manipulation tasks.** **(a)** Training curves for *Door* and *Lift* over 200K environment steps. Each curve reports the mean episode reward across 5 evaluation seeds, with shaded regions indicating ± 1 standard deviation. ACO-MoE achieves significantly faster convergence and higher asymptotic performance than all baselines. **(b) & (c)** Qualitative visualization of the restoration process under VDCS Markov-switching corruptions and Video Hard background distractions, respectively. From left to right: corrupted input x_t, restored RGB $\hat{o}_t$, ground truth o_t, predicted mask m_t, and agent-centric observation $\tilde{x}_t$.



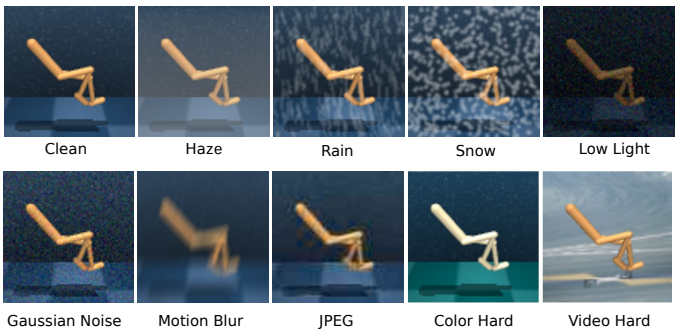
(a) Cartpole Swingup



(b) Finger Turn Hard

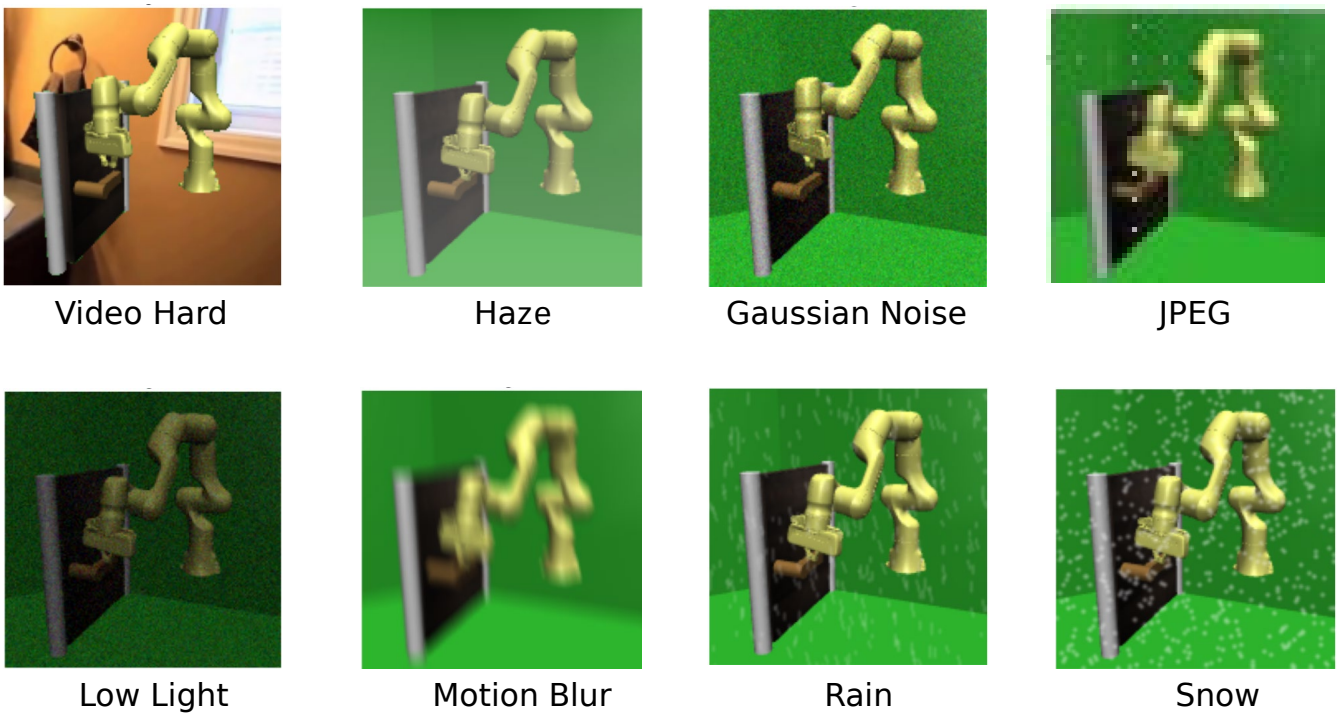


(c) Cheetah Run

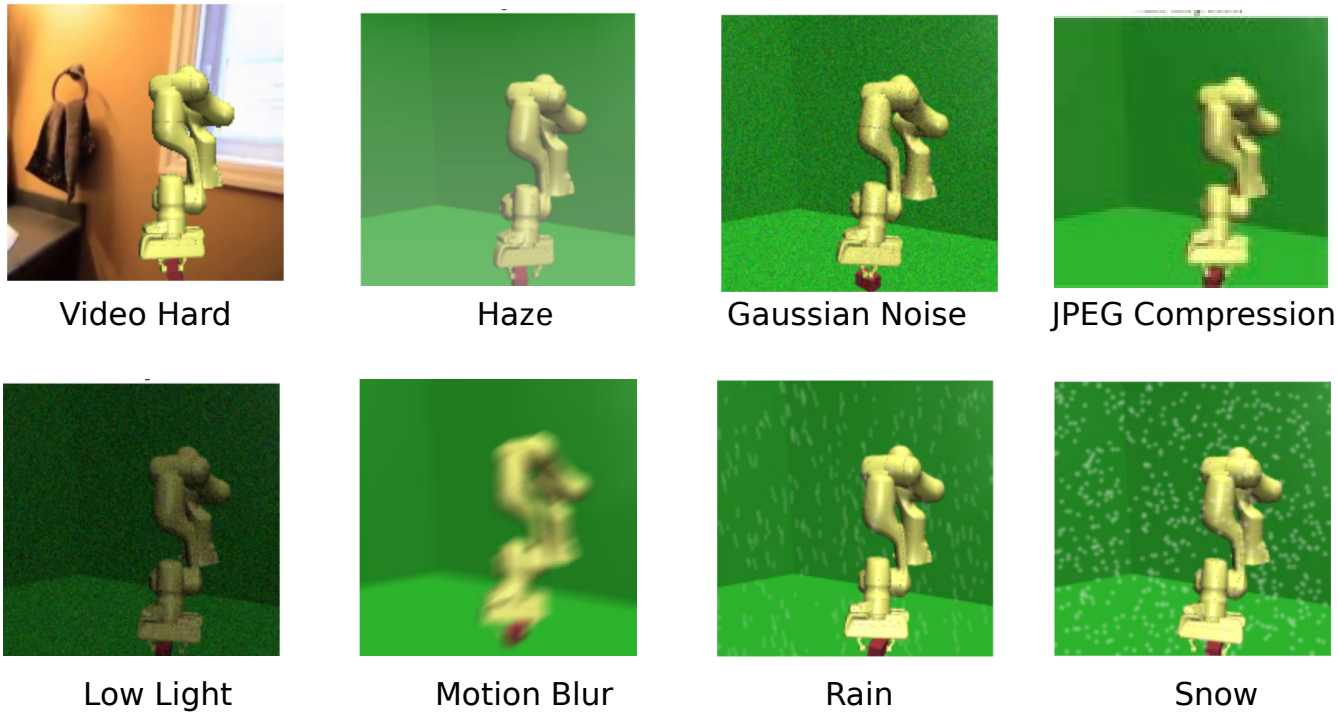


(d) Walker Run

Figure 11: **Comprehensive degradation visualization across VDCS and DMC-GB benchmarks.**

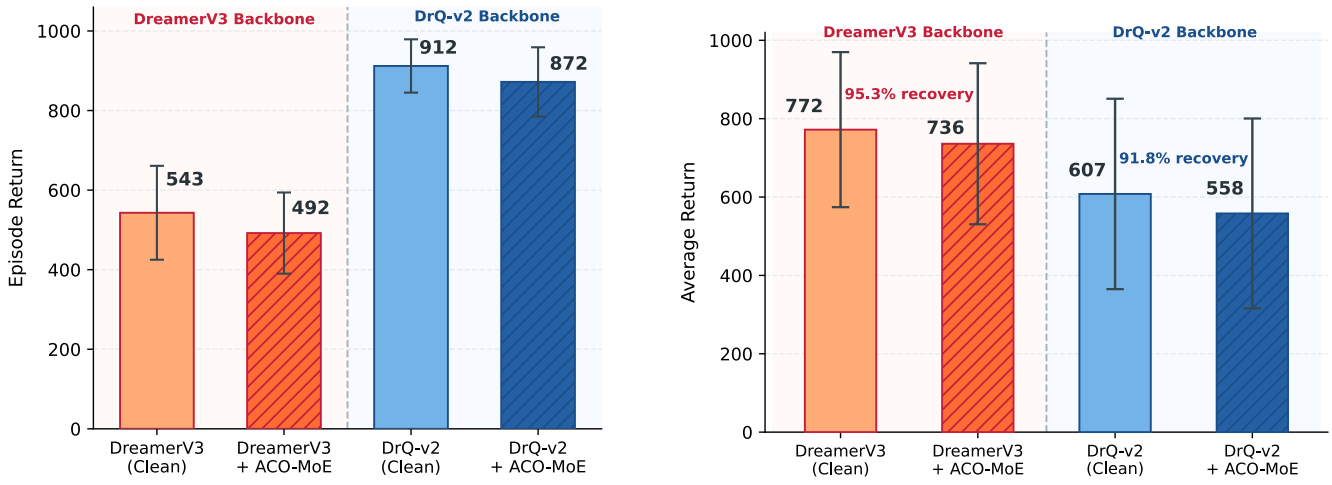


(a) **RoboSuite Door.**



(b) **RoboSuite Lift.**

Figure 12: Visualization of RoboSuite manipulation observations under **VDCS** Markov-switching corruptions and **Video-Hard** background replacement. We apply 7 frame-wise visual corruptions (haze, Gaussian noise, JPEG compression, low light, motion blur, rain, snow) while replacing the original simulator background with dynamic video content (*Video Hard*). Corruption may switch each timestep according to the Markov transition matrix, creating dynamic distracted observations without changing the underlying task dynamics.



(a) finger_spin: Backbone Comparison

(b) Average Return: Backbone × Preprocessing

Figure 13: Backbone generalization of ACO-MoE on VDCS Markov-temporal perturbations. **(a)** DreamerV3's clean ceiling of 543 on `finger_spin` explains its post-ACO-MoE score of 492, while DrQ-v2 reaches 912 and 872 with ACO-MoE (95.6% recovery), confirming the gap is a backbone limitation rather than a preprocessing failure. **(b)** Averaged over 8 tasks, ACO-MoE recovers 95.3% (DreamerV3) and 91.8% (DrQ-v2) of each backbone's clean performance, demonstrating backbone-agnostic robustness.

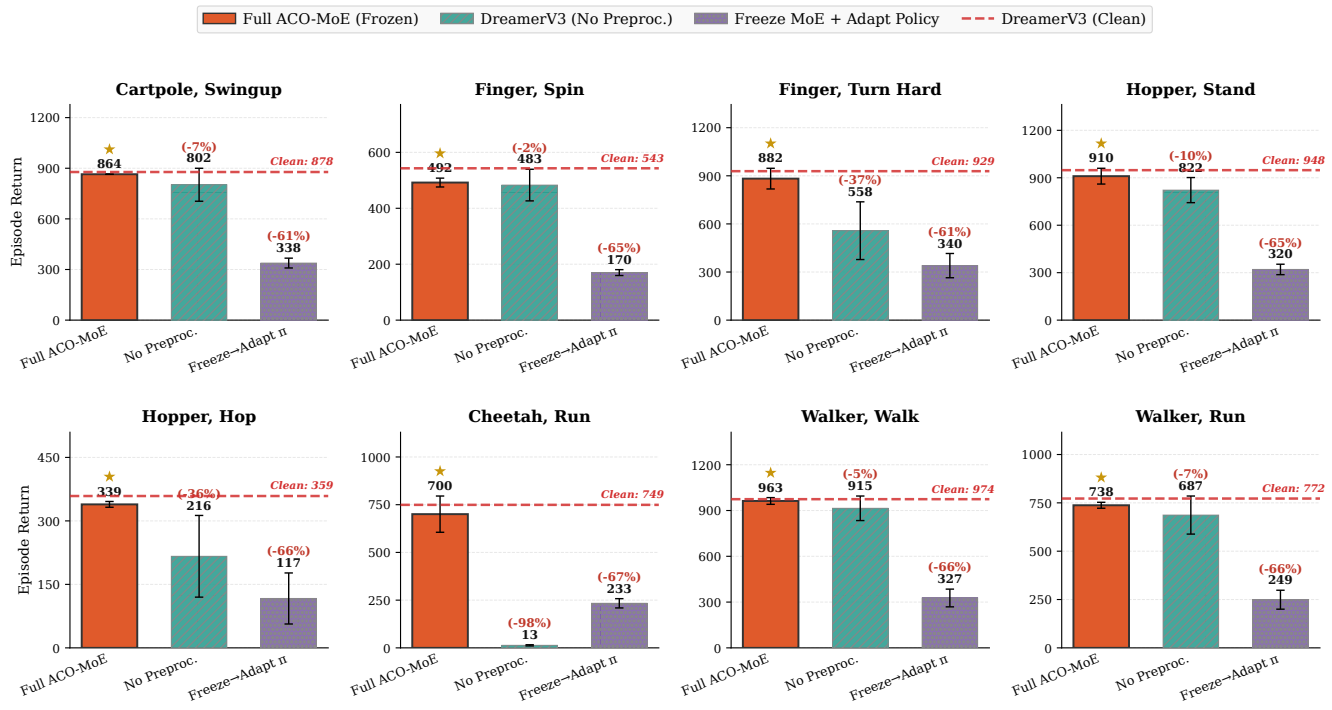


Figure 14: Ablation on VDCS Markov-temporal perturbations with DreamerV3. We compare frozen ACO-MoE, DreamerV3 without preprocessing, and a freeze-then-adapt variant. Percentages denote performance relative to full ACO-MoE. Results are mean ± std over 5 evaluation seeds.

Table 9: Ablation study in DMControl (*video-hard*). Red indicates performance drop relative to the full ACO-MoE model; uncolored positive values indicate marginal gains within one standard deviation.

DMControl (video hard)	**ACO-MoE (Ours)**	Single Expert	w/o frozen policy	mask_only	rgb_only	w/o restoration
cartpole, swingup	706 ± 78	654 ± 105 -52 (7%)	380 ± 96 -327 (46%)	505 ± 116 -202 (29%)	135 ± 95 -571 (81%)	158 ± 68 -548 (78%)
finger, spin	487 ± 24	433 ± 18 -54 (11%)	279 ± 68 -208 (43%)	489 ± 30 +3 (+1%)	277 ± 124 -209 (43%)	160 ± 178 -326 (67%)
walker, stand	985 ± 21	914 ± 21 -71 (7%)	508 ± 33 -477 (48%)	988 ± 13 +3 (+0%)	668 ± 356 -317 (32%)	656 ± 392 -328 (33%)
walker, walk	946 ± 67	904 ± 22 -43 (5%)	479 ± 89 -468 (49%)	975 ± 14 +28 (+3%)	412 ± 263 -534 (56%)	395 ± 285 -552 (58%)
cheetah, run	787 ± 26	580 ± 126 -206 (26%)	441 ± 77 -346 (44%)	539 ± 156 -248 (32%)	188 ± 62 -599 (76%)	86 ± 43 -701 (89%)
Average		-10.9%	-46.7%	-10.6%	-57.1%	-62.8%

Table 10: Ablation study in DMControl (*color-hard*). Red indicates performance drop relative to the full ACO-MoE model; uncolored positive values indicate marginal gains within one standard deviation.

DMControl (color hard)	**ACO-MoE (Ours)**	Single Expert	w/o frozen policy	mask_only	rgb_only	w/o restoration
cartpole, swingup	861 ± 3	843 ± 1 -18 (2%)	463 ± 96 -398 (46%)	737 ± 173 -124 (14%)	218 ± 171 -643 (75%)	625 ± 295 -237 (27%)
finger, spin	505 ± 113	498 ± 11 -8 (2%)	290 ± 68 -216 (43%)	407 ± 110 -98 (19%)	398 ± 161 -107 (21%)	429 ± 78 -77 (15%)
walker, stand	970 ± 47	950 ± 19 -20 (2%)	500 ± 33 -470 (48%)	915 ± 25 -55 (6%)	646 ± 283 -324 (33%)	988 ± 13 +17 (+2%)
walker, walk	954 ± 29	959 ± 26 +5 (+1%)	483 ± 89 -472 (49%)	923 ± 57 -31 (3%)	320 ± 310 -634 (66%)	729 ± 162 -225 (24%)
cheetah, run	615 ± 121	599 ± 170 -16 (3%)	345 ± 77 -271 (44%)	595 ± 152 -20 (3%)	272 ± 162 -344 (56%)	483 ± 147 -132 (21%)
Average		-1.4%	-46.7%	-9.0%	-52.5%	-16.7%

Table 11: Performance on VDCS **static single-degradation** benchmark ($\Pi{=}I$, one corruption fixed per episode). Values are mean $\pm$ std averaged over 8 DMControl tasks. Red: best; Blue: second best. Recovery (%) = score / DreamerV3 clean baseline (769). FTR—the strongest foreground-aware baseline—collapses under motion blur (65) and low-light (225) because SAM-based masking cannot restore physically corrupted foreground pixels. ACO-MoE recovers $\geq$96% for six of seven types; motion blur (75%) remains the hardest due to irreversible foreground pixel destruction. The shaded *Markov-Temporal* row (from Table 1) further shows No-Preproc degrades sharply under switching (594$\rightarrow$552), while ACO-MoE is stable (726$\rightarrow$736).

Degradation	DreamerV3 (No Preproc.)	SimGRL	FTR	ACO-MoE (Ours)	Recovery (%)
Rain	613 ± 355	459 ± 385	516 ± 342	759 ± 221	98.7
Haze	610 ± 356	458 ± 383	451 ± 395	751 ± 222	97.7
Snow	567 ± 376	458 ± 388	463 ± 334	759 ± 217	98.7
Motion Blur	557 ± 411	295 ± 291	65 ± 42	577 ± 262	*75.1*
Gaussian Noise	592 ± 376	462 ± 390	572 ± 363	752 ± 216	97.8
Low-Light	578 ± 375	459 ± 386	225 ± 118	747 ± 218	97.1
JPEG Compr.	639 ± 346	369 ± 332	429 ± 354	739 ± 226	96.1
Avg. (Static)	594 ± 367	423 ± 357	376 ± 276	726 ± 217	**94.5**
Avg. (Markov)	552 ± 122	412 ± 315	389 ± 179	736 ± 206	**95.3**

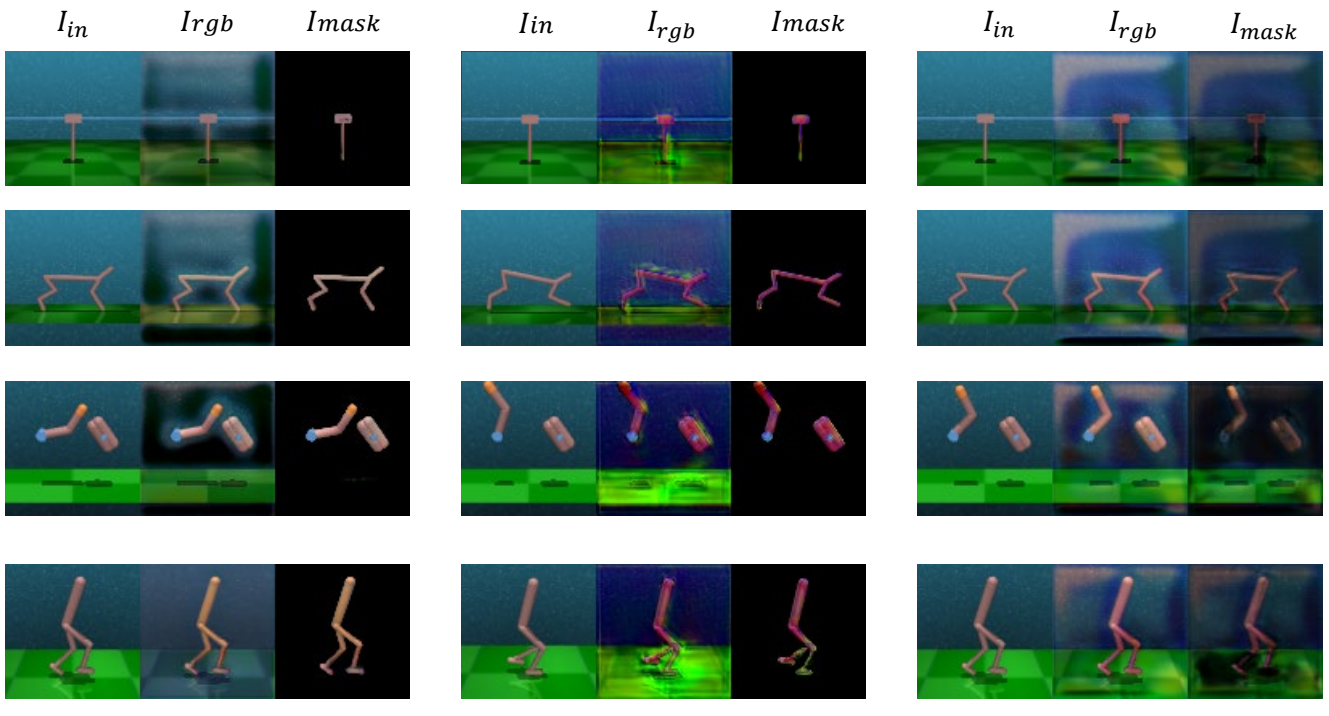


Figure 15: Ablation study on the importance of dual-stream agent-centric observations. We compare the performance of using only RGB restoration, only foreground mask, and our combined approach across various dynamic perturbations.

under physical degradation, including `hopper_stand`, `hopper_hop`, and `cheetah_run`. The only near-tie is `walker_stand`, where the task is almost saturated. These results suggest that generic pixel restoration alone is insufficient; the main benefit comes from ACO-MoE's agent-centric foreground composition.

Table 12: Comparison with generic image restoration methods as frozen preprocessors on VDCS Markov-temporal perturbations. All methods use the same frozen DreamerV3 policy; only the preprocessing module differs. Results are mean ± std over evaluation episodes. △ denotes ACO-MoE's gain over the best generic baseline. Red: best; Blue: second best.

Task	PromptIR [43]	Restormer [68]	AdaIR [7]	NAFNet [4]	**ACO-MoE (Ours)**	△
cartpole, swingup	457 ±185	393 ±142	421 ±216	226 ±90	864 ±75	+407 89%
finger, spin	441 ±38	386 ±108	432 ±23	346 ±39	492 ±102	+51 12%
finger, turn hard	779 ±102	791 ±114	834 ±69	522 ±313	882 ±65	+48 6%
hopper, stand	361 ±150	345 ±88	444 ±126	240 ±57	910 ±50	+466 105%
hopper, hop	130 ±15	95 ±29	111 ±12	54 ±24	339 ±46	+209 161%
cheetah, run	388 ±75	386 ±102	386 ±55	270 ±111	700 ±95	+312 80%
walker, walk	857 ±73	826 ±68	760 ±53	616 ±54	963 ±87	+106 12%
walker, stand†	949 ±62	982 ±15	985 ±16	770 ±101	985 ±18	−0.3 0%
Average	545.2 ±265.7	525.5 ±283.6	546.6 ±268.2	380.5 ±221.0	766.9 ±221.5	+220.3 40.3%

F.7. Zero-Shot Generalization to Unseen OOD Visual Perturbations

We further evaluate whether frozen ACO-MoE generalizes to unseen out-of-distribution visual perturbations. Both ACO-MoE and the downstream controller are kept frozen, with no retraining or test-time adaptation. We use the same eight DMControl tasks as in the main VDCS experiments. Each unseen corruption is applied in single-corruption mode, with one fixed corruption family per episode and a ±10% intensity jitter. Results are averaged over 5 seeds with one evaluation episode per seed.

The six unseen perturbations are shown in Fig. 16: `defocus_blur`, `frost`, `occlusion_patch`, `saturation`, `shadow`, and `shot_noise`. These families are excluded from ACO-MoE pretraining and test different open-set shifts, including blur, weather-like veiling, partial occlusion, color shift, illumination shift, and signal-dependent sensor noise.

Table 13 reports the full results. ACO-MoE transfers well to several unseen perturbations, with average returns around 745–752 for `frost`, `saturation`, `shadow`, and `shot_noise`. The hardest cases are `defocus_blur` and `occlusion_patch`, with average returns of 542.1 and 649.8, respectively. This indicates that ACO-MoE generalizes beyond the training corruption families, but remains limited when unseen perturbations strongly destroy spatial structure or occlude task-relevant foreground regions.

Table 13: Zero-shot evaluation on unseen OOD visual perturbations. Both ACO-MoE and the downstream controller are frozen, with no retraining or test-time adaptation. Results are mean ± std over 5 seeds, with 1 evaluation episode per seed. The last row reports the mean over the 8 tasks for each unseen corruption family.

Task	Defocus Blur	Frost	Occlusion Patch	Saturation	Shadow	Shot Noise
cartpole_swingup	859.65 ± 2.65	865.04 ± 0.42	854.98 ± 4.06	865.40 ± 0.45	864.85 ± 0.60	863.85 ± 0.92
finger_spin	491.80 ± 10.76	505.80 ± 10.19	467.60 ± 15.08	509.40 ± 8.24	499.40 ± 25.06	497.00 ± 14.31
finger_turn_hard	374.80 ± 444.15	867.00 ± 131.93	611.80 ± 357.54	837.40 ± 147.54	957.20 ± 23.60	927.60 ± 38.19
hopper_stand	329.37 ± 61.02	939.58 ± 17.52	890.74 ± 75.71	935.79 ± 21.30	929.05 ± 23.80	925.02 ± 21.31
hopper_hop	98.60 ± 54.01	355.79 ± 18.09	285.81 ± 10.30	361.30 ± 19.24	329.14 ± 37.60	364.52 ± 20.33
cheetah_run	541.97 ± 111.11	753.71 ± 67.93	567.78 ± 88.08	763.86 ± 111.11	743.88 ± 76.45	733.90 ± 59.70
walker_walk	936.41 ± 37.17	962.68 ± 21.93	911.50 ± 37.06	962.73 ± 16.38	950.86 ± 45.54	946.81 ± 38.25
walker_run	704.00 ± 39.96	745.47 ± 13.56	608.49 ± 43.38	722.29 ± 32.53	739.28 ± 22.69	748.05 ± 13.57
Average over tasks	542.1	749.4	649.8	744.8	751.7	750.8



Figure 16: Examples of the six unseen OOD visual perturbations used for zero-shot evaluation: `defocus_blur`, `frost`, `occlusion_patch`, `saturation`, `shadow`, and `shot_noise`. These perturbation families are excluded from ACO-MoE pretraining and are evaluated only at test time with the frozen observation adapter and frozen downstream controller.