UNDERSTANDING ROBUSTNESS TO OBSERVATION CORRUPTIONS THROUGH ACTOR–CRITIC INFORMATION SPECIALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Observation corruptions can disrupt reinforcement learning (RL) by altering an agent's inputs and subsequent decisions. However, how Actor–Critic architecture shapes the resulting information and output responses remains poorly understood. We study shared and decoupled PPO trained on clean observations across Procgen, DMC, and Robust Gymnasium. Our framework combines information-theoretic decompositions, representation-to-output sensitivity, and performance-difference relations. Paired diagnostics and closed-loop evaluations address four questions concerning control performance, information specialization, output sensitivity, and responses across corruption types. Decoupled PPO achieves higher corrupted returns in most evaluated tasks, while return-related specialization exhibits greater proportional erosion than action-related specialization. The results distinguish information retention from feature and output stability, supporting a joint assessment of robustness that accounts for corruption type. Our code is included in the supplementary material and will be released on GitHub.

1 INTRODUCTION

Deep reinforcement learning (RL) has achieved strong performance in sequential decision making, but learned policies remain sensitive to observation changes at test time (Mnih et al., 2015; Landers & Doryab, 2023; Hafner et al., 2025). Non-adversarial corruptions such as noise, blur, missing measurements, and compression can substantially degrade neural-network performance (Hendrycks & Dietterich, 2019). In RL, such changes can alter the current action, affect subsequent interactions, and ultimately reduce accumulated return. Robustness to observation corruption is therefore important for control under imperfect sensing conditions (Igl et al., 2019).

Existing methods improve robustness through data augmentation, invariance regularization, and representation learning (Yarats et al., 2021; Fan & Li, 2022). However, final return alone does not reveal what information changes under corruption or how representation shifts affect policy and value outputs. Understanding robustness therefore requires analysis beyond task performance.

Actor–Critic methods provide a natural setting for this analysis (Sutton et al., 1999; Mnih et al., 2016; Schulman et al., 2017). The Actor and Critic serve different roles and need not encode the same information. Prior work has shown that separating them can induce infor-

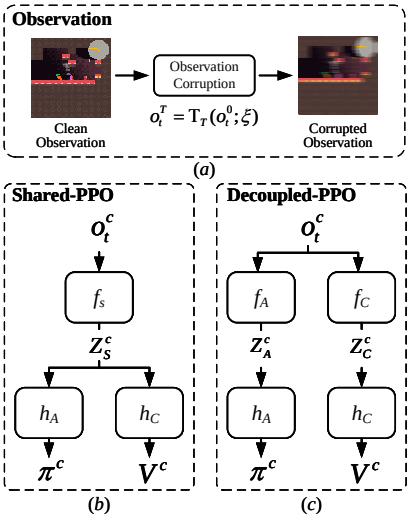

Figure 1: **Observation-corruption analysis in shared and decoupled Actor–Critic architectures.** (a) Corrupted observation transformation. (b) Shared representation Z_S. (c) Decoupled Actor and Critic representations Z_A and Z_C.

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

mation specialization, with Actor representations emphasizing action-related information and Critic representations emphasizing value- or return-related information (Cobbe et al., 2021; Raileanu & Fergus, 2021; Moon et al., 2022; Garcin et al., 2025). Yet how this specialization responds to observation corruption remains unclear, as does its relation to output sensitivity and closed-loop performance.

We organize this investigation around four research questions (Table 1). The first two ask whether decoupling improves control performance under corruption and how corruption changes information retention and Actor–Critic specialization. The other two ask whether representation drift alone explains output sensitivity and how responses across levels vary with corruption type. These questions extend the architecture comparison beyond final return to the information organization and network responses that accompany control under corrupted observations.

We compare shared and decoupled PPO across Procgen, DMC, and Robust Gymnasium, covering visual and state-vector control. As illustrated in Figure 1, our analysis follows observations through representations and the original output heads to closed-loop performance. Information-theoretic decompositions distinguish observation-channel limitations from encoder retention, while mutual-information measurements characterize changes in action-related and return-related specialization. We then analyze how feature shifts interact with fixed output heads and use performance-difference relations to connect local policy responses with return. Comparisons across corruption types integrate these analyses, while additional PPG and DrAC evaluations examine performance under alternative training schemes.

2 Related Work

2.1 Robustness to Observation Perturbations

Observation robustness covers common corruptions (Hendrycks & Dietterich, 2019) and adversarial perturbations (Madry et al., 2018; Goodfellow et al., 2014). RL defenses use policy regularization (Zhang et al., 2020), learned adversaries (Zhang et al., 2021b), detection and denoising (Xiong et al., 2023), or Lipschitz-constrained policies (Nie et al., 2024). Visual RL addresses visual variability through data augmentation (Yarats et al., 2021; Laskin et al., 2020b; Hansen et al., 2021) and representation learning (Laskin et al., 2020a; Hansen & Wang, 2021; Zhang et al., 2021a; Fan & Li, 2022). We examine how sharing versus decoupling Actor–Critic representations shapes information specialization, output sensitivity, and performance under observation corruption.

2.2 Actor–Critic Representation Specialization

Separating policy and value learning can reduce optimization interference and improve generalization (Cobbe et al., 2021; Raileanu & Fergus, 2021). Further studies examine critic regularization (Moon et al., 2022) and the contributions of policy regularization and data diversity to PPG (Wang et al., 2023). Representation analyses characterize learned feature properties (Wang et al., 2024) and probe action- and reward-relevant information (Zhang et al., 2024). Mutual-information analysis further identifies distinct information preferences in decoupled Actors and Critics (Garcin et al., 2025). We examine how observation corruption alters these role differences, distinguishing information retention from the stability of the original output heads and closed-loop performance.

3 Preliminaries

Agents and observations. We consider an MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \rho_0, \gamma)$ with initial-state distribution ρ_0 and discount $\gamma \in [0, 1)$. Agents observe o_t^0 and maximize

$$J(\pi) = \mathbb{E}_{\pi,\rho_0}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]. \tag{1}$$

Shared and decoupled PPO use

$$Z_S = f_S(o), \quad \pi = h_A(Z_S), \quad \hat{V} = h_C(Z_S), \tag{2}$$

$$Z_A = f_A(o), \quad Z_C = f_C(o), \quad \pi = h_A(Z_A), \quad \hat{V} = h_C(Z_C). \tag{3}$$

Table 1: **Research questions and corresponding analysis levels.**

	Research question	Analysis level
RQ1	Does decoupling improve closed-loop performance under observation corruptions?	Performance
RQ2	How does observation corruption affect Actor–Critic information specialization and information retention?	Information
RQ3	Is Actor–Critic output robustness determined by representation drift alone?	Representation → Output
RQ4	How do information, representation, output, and closed-loop responses jointly characterize robustness across observation corruptions?	Cross-level

During frozen evaluation, $o_t^T = \mathcal{T}_T(o_t^0; \xi_t)$ changes observations while preserving dynamics and rewards. Superscripts $0, T$ denote paired clean and corrupted quantities; experiments report undiscounted episode returns separately from the theoretical objective.

Information targets. Mutual information measures statistical dependence:

$$I(X;Y) = \mathbb{E}_{(x,y)\sim p(x,y)}\left[\log \frac{p(x,y)}{p(x)p(y)}\right], \tag{4}$$

where p denotes the corresponding probability mass or density and natural logarithms give units of nats. For reference actions and returns held fixed across each observation pair, define

$$I_A(Z) = I(Z; A_{\mathrm{ref}}), \qquad I_G(Z) = I(Z; G_{\mathrm{ref}}). \tag{5}$$

These quantities concern the specified targets, rather than all task-relevant information.

4 Analytical Framework for Robustness in Actor-Critic Architectures

4.1 Analysis Overview

We examine corrupted performance (RQ1), information specialization (RQ2), representation-to-output sensitivity (RQ3), and their joint responses across corruption types (RQ4); Table 1 gives the detailed questions. Paired diagnostics use matched inputs, whereas closed-loop evaluation allows changed actions to affect subsequent interactions.

4.2 Corruption Effects on Information Specialization

For a fixed corruption type, let $Y \in \{A_{\mathrm{ref}}, G_{\mathrm{ref}}\}$ and $R \in \{S, A, C\}$. Data processing and the MI chain rule give the following statements.

Lemma 1 (Information Limitation under Observation Corruption)**.** *If target-independent corruption and frozen encoding yield $Y \to O^0 \to O^T \to Z_R^T$, with finite MI terms, then*

$$I(Y; Z_R^T) \le I(Y; O^T) \le I(Y; O^0). \tag{6}$$

Corollary 1 (Information Decomposition under Corruption)**.** *Under Lemma 1,*

$$I(Y; O^0) - I(Y; Z_R^T) = I(Y; O^0 \mid O^T) + I(Y; O^T \mid Z_R^T). \tag{7}$$

Corollary 2 (Representation Information Shift under Corruption)**.** *Under Lemma 1, if additionally $Y \to O^0 \to Z_R^0$ and the clean MI terms are finite, then*

$$\begin{aligned} I(Y; Z_R^0) - I(Y; Z_R^T) = {} & I(Y; O^0 \mid O^T) + I(Y; O^T \mid Z_R^T) \\ & - I(Y; O^0 \mid Z_R^0). \end{aligned} \tag{8}$$

Representation-information change therefore combines observation-channel loss and changes in encoder retention; data processing alone does not order clean and corrupted representation MI. Proofs and history-dependent corruptions are addressed in Appendix A.1.

The two conditional terms distinguish where target information is lost. $I(Y; O^0 \mid O^T)$ measures information in the clean observation that is unavailable after corruption, whereas $I(Y; O^T \mid Z_R^T)$ measures corrupted-observation information not retained by the encoder. Subtracting the corresponding clean encoding gap explains why the representation-level change can differ from the observation-level loss.

For decoupled features under $c \in \{0, T\}$, define

$$S_A^c = I_A(Z_A^c) - I_A(Z_C^c), \qquad S_G^c = I_G(Z_C^c) - I_G(Z_A^c). \tag{9}$$

Positive scores indicate relatively more action information in the Actor and return information in the Critic. For sharing, we compare $I_A(Z_S^c)$ and $I_G(Z_S^c)$ directly. Writing $\Delta S_Y^T = S_Y^T - S_Y^0$ and $\Delta I_Y^T(Z_R) = I_Y(Z_R^T) - I_Y(Z_R^0)$ for $Y \in \{A, G\}$ gives

$$\begin{aligned} \Delta S_A^T &= \Delta I_A^T(Z_A) - \Delta I_A^T(Z_C), \\ \Delta S_G^T &= \Delta I_G^T(Z_C) - \Delta I_G^T(Z_A). \end{aligned} \tag{10}$$

A shrinking role gap can reflect either loss in one branch or gain in the other, motivating analysis of all four channels.

4.3 Representation-to-Output Sensitivity

Let U_A, U_C be the features directly entering the frozen output heads and $\Delta U_R^T = U_R^T - U_R^0$. For a discrete Actor and linear Critic,

$$\ell = W_A U_A + b_A, \quad \pi = \mathrm{softmax}(\ell), \quad \hat{V} = w_C^\top U_C + b_C. \tag{11}$$

With $P = I - \mathbf{1}\mathbf{1}^\top / K$ removing common shifts across K logits,

$$P\Delta\ell^T = PW_A \Delta U_A^T, \tag{12}$$

$$\Delta\hat{V}^T = w_C^\top \Delta U_C^T. \tag{13}$$

Null-space displacements leave the corresponding output unchanged; visible displacements are weighted by the head. Thus, neither total drift nor its row-space energy fraction alone determines output sensitivity. These identities apply at the final head inputs, after any intervening nonlinear mapping.

This distinction separates the amount, direction, and output effect of a representation displacement. For the Critic, only its component along w_C changes the predicted value. For the Actor, the centered mapping PW_A removes directions that affect neither relative logits nor action probabilities. Measuring both feature drift and the mapped displacement therefore connects representation changes to the functions actually computed by the trained agent.

Lemma 2 (Policy Sensitivity for a Linear Softmax Head)**.** *For the fixed unit-temperature head in Eq. 11, with natural logarithms,*

$$D_{\mathrm{KL}}(\pi^0 \| \pi^T) \le \tfrac{1}{4} \| PW_A \Delta U_A^T \|_2^2. \tag{14}$$

The proof is in Appendix A.2; greedy-action stability also depends on the clean logit margin (Appendix A.3). For a Gaussian Actor with linear mean, $\Delta\mu^T = W_\mu \Delta U_A^T$; with fixed positive-definite covariance, KL weights this displacement by the inverse covariance (Appendix A.4). We therefore measure policy divergence, executed-action change, and value drift separately; value drift is sensitivity, not prediction error.

4.4 From Local Output Sensitivity to Return Degradation

Let π_0, π_T denote behavior policies under the actual action-selection rule, with $J^c = J(\pi_c)$ and $D_{\mathrm{ret}}^T = J^0 - J^T$. These differ from local network distributions when evaluation uses greedy actions.

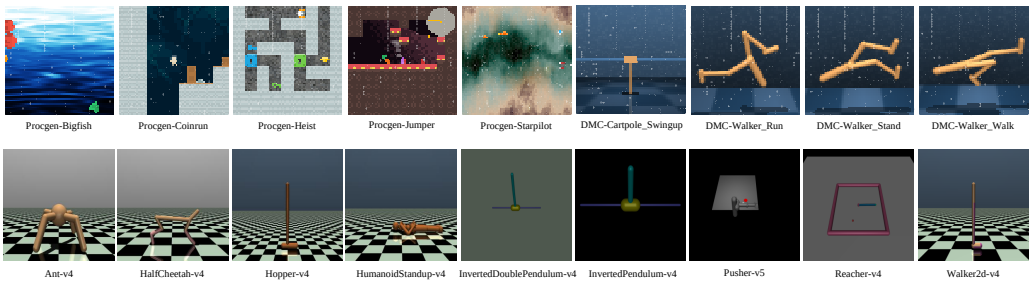

Figure 2: Example views of the five Procgen, four DMC, and nine Robust Gymnasium tasks. Procgen and DMC use image observations; Robust Gymnasium policies receive state vectors.

On a common Markov state space, augmented with history when needed, the performance-difference identity gives (Schulman et al., 2015; Achiam et al., 2017)

$$J(\pi_T) - J(\pi_0) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_T},\, a\sim\pi_T(\cdot|s)}[A^{\pi_0}(s,a)], \tag{15}$$

where d^{π_T} is normalized discounted visitation. Similar local changes can therefore yield different returns through action consequences and induced visitation.

Lemma 3 (Policy-Divergence Bound)**.** *For stationary behavior policies on the same Markov state space, unchanged dynamics and rewards, $|r(s,a)| \le R_{\max}$, and $0 \le \gamma < 1$, let*

$$\delta_T = \sup_s D_{\mathrm{TV}}(\pi_T(\cdot \mid s), \pi_0(\cdot \mid s))\,. \tag{16}$$

Then

$$\|V^{\pi_T} - V^{\pi_0}\|_\infty \le \frac{2R_{\max}}{(1-\gamma)^2}\delta_T. \tag{17}$$

The same bound holds for $|J^T - J^0|$ under a common initial-state distribution.

The proof is in Appendix A.5. Average KL on paired observations does not establish this uniform behavior-policy condition, motivating direct return evaluation alongside local diagnostics.

5 Experiments

5.1 Experimental Setup

Benchmarks and agents. We compare Shared PPO and Decoupled PPO on 5 Procgen tasks (Cobbe et al., 2020), 4 visual DMC tasks (Tunyasuvunakool et al., 2020), and 9 state-vector tasks from Robust Gymnasium (Tables 3 and 24). Each configuration uses three training seeds and is trained without the evaluation corruptions; subscripts S and D distinguish the configurations. Figure 2 shows example views of all benchmark environments.

Observation corruptions. Visual observations receive 16 corruption types. Vector observations receive scalar Gaussian noise $\epsilon_t \sim \mathcal{N}(0, 0.005^2)$ or constant shifts of $+0.005$ and $+0.4$, applied identically to all coordinates before frozen normalization. Rewards and physical dynamics remain unchanged.Further details and explanations of the perturbation settings are provided in Appendix B.2.

Evaluation and aggregation. Closed-loop evaluation reports mean undiscounted episode returns $\widehat{J}^0, \widehat{J}^T$ and signed loss $\widehat{D}^T_{\mathrm{ret}} = \widehat{J}^0 - \widehat{J}^T$. Paired diagnostics keep $A_{\mathrm{ref}}, G_{\mathrm{ref}}$ fixed and measure information, specialization, policy divergence $\mathcal{K}^T = \mathbb{E}[D_{\mathrm{KL}}(\pi^0 \| \pi^T)]$, greedy-action flips $\mathcal{F}^T = \Pr[\arg\max_a \ell^0_a \neq \arg\max_a \ell^T_a]$, and value drift $\mathcal{D}^T_V = \mathbb{E}[|\Delta\hat{V}^T|]$. Continuous actions use the corresponding action distances. After averaging seeds within tasks and conditions, we weight tasks and conditions equally. For positive clean gaps, specialization erosion is $-\overline{\Delta S^T_Y}/\overline{S^0_Y}$; individual-channel MI losses are within-task, within-condition ratios averaged afterward. Appendix B.3 specifies the remaining conventions and diagnostic scope.

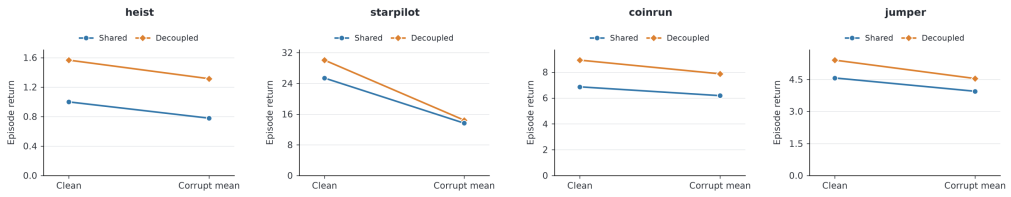

Figure 3: **Clean and corrupted returns on four Procgen tasks.** Shared and Decoupled PPO are compared in each task's original reward units. Corrupted endpoints average evaluated conditions; markers are point estimates and higher is better.

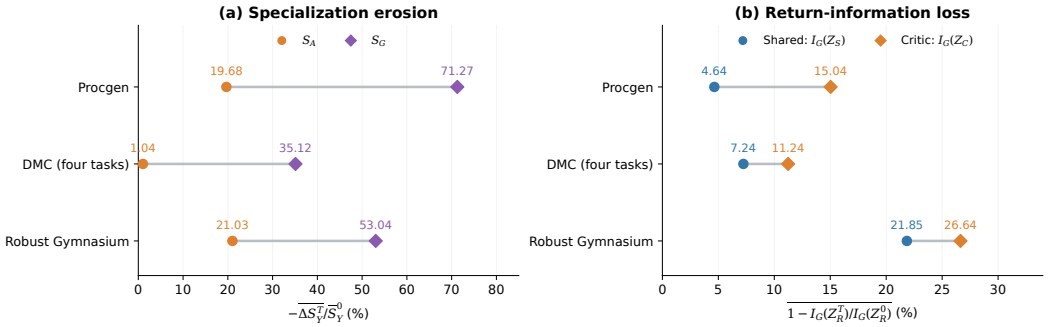

Figure 4: **Specialization erosion and return-information loss.** (a) Decoupled role-gap erosion $-\overline{\Delta S_Y^T / S_Y^0}$, $Y \in \{A, G\}$. (b) Mean relative MI loss $\overline{1 - I_G(Z_R^T)/I_G(Z_R^0)}$ for shared features (S) and the decoupled Critic (C).

For Procgen, MI is estimated using nearest-neighbor estimators: KSG for continuous return targets (Kraskov et al., 2004) and Ross for discrete action targets (Ross, 2014). Branch-specific eight-dimensional PCA projections are fitted on clean features and fixed across clean and corrupted inputs. The observation-proxy analysis uses a separate fixed projection of the input. The paired geometry analysis instead measures the original features entering the output heads, preserving the coordinate system used by the trained networks.

5.2 Performance under Observation Corruption

For RQ1, Figure 3 shows four Procgen tasks; Appendix C.1 provides the remaining tasks. Decoupled PPO achieves higher mean corrupted return in 14 of 18 tasks: all five Procgen tasks, all 4 reported DMC tasks, and 5 of 9 Robust Gymnasium tasks. Relative retention $\widehat{J}^T / \widehat{J}^0$ instead favors sharing on average in Procgen and the positive-clean-return vector tasks, with nearly equal DMC aggregates. Thus, higher attained return and greater proportional preservation of clean performance have different architectural orderings. Appendix C.1.1 further resolves these comparisons by corruption type.

5.3 Asymmetric Erosion of Information Specialization

For RQ2, Figure 4 compares specialization erosion and relative return-information loss; higher values indicate stronger erosion or loss. All three clean benchmark aggregates satisfy $I_A(Z_A^0) > I_A(Z_C^0)$ and $I_G(Z_C^0) > I_G(Z_A^0)$, indicating relative Actor action specialization and Critic return specialization, qualitatively consistent with prior analysis (Garcin et al., 2025). Appendix Tables 28 and 29 report raw MI and all four relative channel losses.

Return specialization is more strongly eroded. Mean S_A and S_G remain positive after corruption, but return-specialization erosion reaches 71.27%, 35.12%, and 53.04% in Procgen, DMC, and Ro-

Table 2: **Representation and output responses in Procgen.** The first three rows use the output evaluation; remaining rows use paired geometry and its matched outputs. Actor/Critic geometry uses the corresponding final-head inputs U_A/U_C. Drift is normalized by clean feature variation; Spearman values are median within-checkpoint correlations (Appendix C.3).

Quantity	Shared PPO	Decoupled PPO
Policy divergence $\mathcal{K}^T$	0.75805	0.36954
Greedy-action flips $\mathcal{F}^T$	50.73%	34.13%
Value drift $\mathcal{D}_V^T$	1.08709	0.51219
Actor feature drift $D_{\mathrm{rel}}(U_A)$	0.69350	0.61670
Critic feature drift $D_{\mathrm{rel}}(U_C)$	0.69350	0.77120
Centered logit shift $\mathbb{E}\|P\Delta\ell^T\|_2$	5.66270	4.51675
Policy-head row-space energy fraction	5.69%	20.51%
Spearman: total feature L2 and $\mathcal{K}^T$	0.903	0.953
Spearman: centered logits and $\mathcal{K}^T$	0.965	0.944

bust Gymnasium, versus 19.68%, 1.04%, and 21.03% for action specialization. In Procgen, $I_G(Z_C)$ decreases while $I_G(Z_A)$ increases slightly; in DMC and Robust Gymnasium, both decrease, with a larger Critic reduction. Each pattern narrows S_G, illustrating why role-gap erosion differs from Critic information loss alone.

The aggregate asymmetry also varies by corruption. (Appendix C.2.2). In DMC, translation reverses the mean action-specialization gap, with 176.21% erosion, whereas Markov switching expands it, corresponding to -208.13% erosion. The small aggregate action erosion therefore reflects offsetting responses across corruption types; it does not imply that action specialization is uniformly preserved.

Sharing retains proportionally more return information. Mean relative loss in $I_G(Z_S)$ is smaller than in the decoupled Critic's $I_G(Z_C)$: 4.64% versus 15.04% in Procgen, 7.24% versus 11.24% in DMC, and 21.85% versus 26.64% in Robust Gymnasium. Appendix C.2.1 extends this comparison to all four action- and return-information channels. This information-retention ordering differs from the attained-return ordering, motivating analysis of the original output heads.

5.4 Representation-to-Output Responses

For RQ3, the paired Procgen analysis shows smaller policy divergence, action-flip rate, and value drift under decoupling (Table 2). Normalized Actor drift is smaller, but Critic drift is larger: greater output stability does not require uniformly smaller feature changes. The Critic result is consistent with $\Delta\hat{V}^T = w_C^\top \Delta U_C^T$, which depends on the head-weighted displacement.

For the Actor, decoupling produces smaller centered logit shifts despite a larger policy-head row-space energy fraction. Total feature L2 and centered logit displacement both correlate strongly with $\mathcal{K}^T$; centered logits correlate more strongly only for Shared PPO. The weighted mapping $PW_A\Delta U_A^T$, rather than row-space fraction alone, therefore helps interpret the output response. These results distinguish feature invariance from stability of the functions computed by the frozen heads.

The row-space fractions make this distinction concrete: Decoupled PPO places 20.51% of displacement energy in the policy-head row space, compared with 5.69% for Shared PPO, yet has a smaller centered logit shift. Direction allocation must therefore be interpreted together with the displacement magnitude and the weighting imposed by each trained head.

5.5 Tracing Corruption Responses

For RQ4, larger local policy changes accompany larger proportional return losses across Procgen corruption types (Figure 5). Spearman correlations for Shared/Decoupled PPO are $0.697/0.809$ for $\mathcal{K}^T$ and $0.715/0.818$ for $\mathcal{F}^T$. Robust Gymnasium also shows positive median output–return

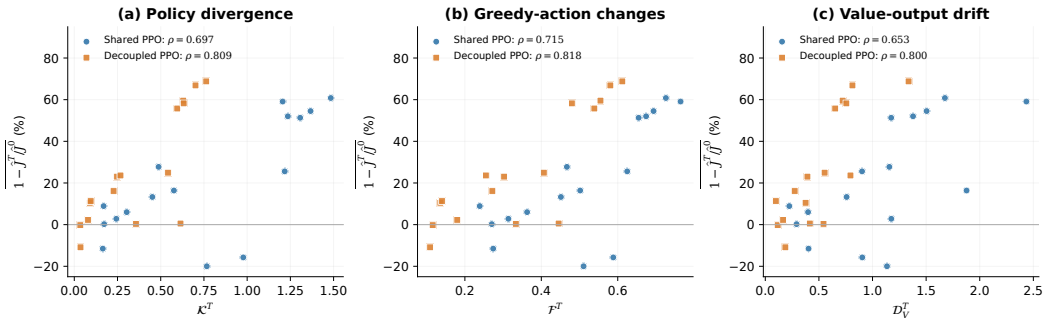

Figure 5: **Local outputs and proportional return loss in Procgen.** Each point averages one corruption category across tasks after seed aggregation. Axes show $\mathcal{K}^T$, $\mathcal{F}^T$, or $\mathcal{D}_V^T$ against mean within-task return loss (%). Legends report descriptive Spearman correlations over 16 categories.

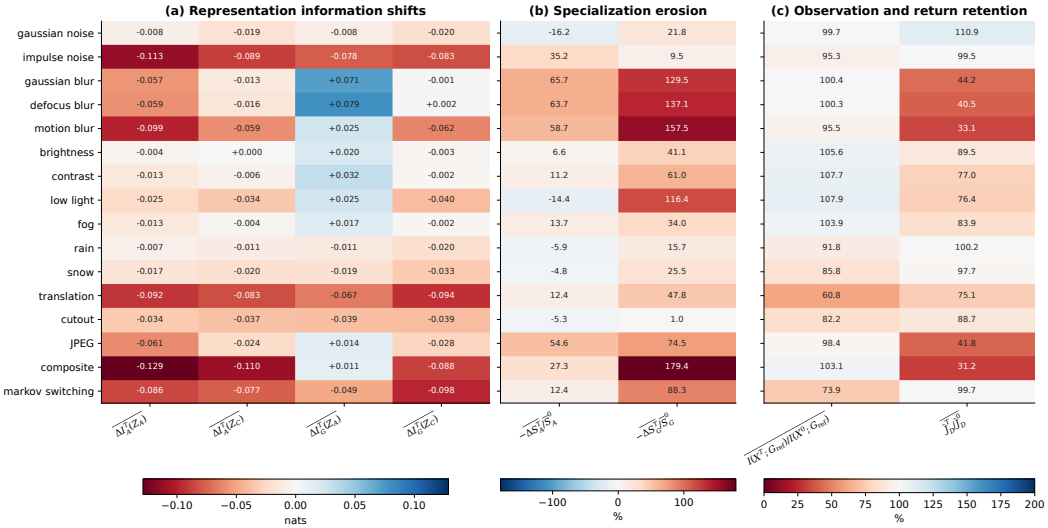

Figure 6: **Corruption-specific responses in Procgen.** (a) Four decoupled MI shifts $\overline{\Delta I_Y^T(Z_R)}$ (nats). (b) Specialization erosion (%): negative values indicate expansion and values above 100% indicate gap reversal. (c) Projected observation–return MI and Decoupled PPO return retention, averaged as within-task ratios (%). Colors center at zero for changes/erosion and 100% for retention; observation MI concerns the fixed projection X.

associations under its separate aggregation (Table 31). These are descriptive associations; the frozen Critic remains a parallel diagnostic.

Figure 6 aligns four-channel MI shifts, specialization erosion, observation-proxy MI retention, and Decoupled PPO return across 16 corruption types. Defocus and Gaussian blur leave the measured observation-proxy association approximately unchanged, yet reverse mean S_G and substantially reduce return. Translation and cutout reduce the proxy association more while preserving more return (Table 32). Under the two blur conditions, Actor return information increases while Critic return information changes much less, so the specialization contraction reflects a changed role contrast rather than uniform information removal.

Quantitatively, defocus and Gaussian blur retain 40.47% and 44.24% of Decoupled PPO return while eroding S_G by 137.12% and 129.49%, respectively. Translation and cutout retain 75.07% and 88.66% of return. The observation-proxy MI concerns a fixed low-dimensional projection, so these patterns describe complementary empirical responses rather than a bound from observation MI to achieved return.

Corruptions therefore produce distinct combinations of information change, specialization erosion, output sensitivity, and control degradation. The measurements play complementary roles: information concerns reference-target associations, local diagnostics characterize frozen-network responses, and return measures the resulting control outcome.

5.6 Return Performance beyond PPO

To examine whether the observed advantage of decoupling is specific to standard PPO, we further evaluate PPG and DrAC on Procgen. PPG provides a stronger separation between policy and value optimization, whereas DrAC introduces augmentation-based policy and value regularization to encourage invariance to observation transformations. These comparisons test whether architectural separation and explicit robustness regularization provide similar or complementary benefits.

Table 3: **Clean and corrupted Procgen returns beyond standard PPO.**

Task	Shared PPO	Decoupled PPO	PPG	Shared PPO + DrAC	Decoupled PPO + DrAC
Clean					
BigFish	2.31	20.57	1.50	8.87	26.97
CoinRun	6.87	8.93	8.33	8.67	9.33
Heist	1.00	1.57	1.00	2.67	4.33
Jumper	4.57	5.40	3.67	6.00	6.33
StarPilot	25.40	30.07	8.00	36.30	36.10
Corrupted					
BigFish	2.04	13.89	1.23	5.34	14.98
CoinRun	6.19	7.87	8.33	7.27	9.13
Heist	0.78	1.31	1.50	2.04	2.92
Jumper	3.94	4.54	4.25	3.42	4.90
StarPilot	13.65	14.43	5.44	17.37	16.46

PPG exceeds Decoupled PPO only on CoinRun and Heist, suggesting that stronger separation of policy and value optimization alone does not consistently improve corruption robustness.

DrAC provides a clearer benefit. It improves corrupted return over the corresponding PPO baseline on all five tasks for Decoupled PPO and on four tasks for Shared PPO. With DrAC, the decoupled architecture remains better on four of five tasks. These results suggest that architectural decoupling and augmentation-based invariance provide complementary robustness benefits. Additional results are provided in Appendix C.5.

6 Conclusion

We studied observation-corruption robustness through control performance, information specialization, output sensitivity, and responses across corruption types. Decoupled PPO achieves higher corrupted returns in most evaluated tasks, while return specialization S_G exhibits greater proportional erosion than action specialization S_A. Shared representations retain proportionally more return-related information than decoupled Critics, showing that information retention and control performance need not favor the same architecture. The paired analysis further shows that output sensitivity depends on how representation shifts interact with the trained Actor and Critic heads, rather than on representation drift alone.

Different corruption types produce distinct combinations of information change, specialization erosion, output sensitivity, and return degradation. These findings support a joint analysis of information organization, network responses, and closed-loop behavior, with attention to the corruption type rather than reliance on a single intermediate statistic.

Building on this framework, future work will extend the study to more algorithms and environments and systematically examine corruption severity beyond the fixed settings considered here. These directions aim to further disentangle the mechanisms connecting observation corruption, network responses, and control performance.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. Pmlr, 2017.

Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pp. 2048–2056. PMLR, 2020.

Karl W Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *International Conference on Machine Learning*, pp. 2020–2027. PMLR, 2021.

Jiameng Fan and Wenchao Li. Dribo: Robust deep reinforcement learning via multi-view information bottleneck. In *International conference on machine learning*, pp. 6074–6102. PMLR, 2022.

Samuel Garcin, Trevor McInroe, Pablo Samuel Castro, Christopher Lucas, David Abel, Prakash Panangaden, and Stefano V Albrecht. Studying the interplay between the actor and critic representations in reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 35590–35616, 2025.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Shangding Gu, Laixi Shi, Muning Wen, Ming Jin, Eric Mazumdar, Yuejie Chi, Adam Wierman, and Costas Spanos. Robust gymnasium: A unified modular benchmark for robust reinforcement learning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 102014–102041, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/fcc22e5b7d5d2155d994da22d045f0a6-Paper-Conference.pdf.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025. doi: 10.1038/s41586-025-08744-2.

Nicklas Hansen and Xiaolong Wang. Generalization in reinforcement learning by soft data augmentation. In *2021 IEEE International Conference on Robotics and Automation*, pp. 13611–13617. IEEE, 2021. doi: 10.1109/ICRA48506.2021.9561103.

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

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm.

Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/e2ccf95a7f2e1878fcafc8376649b6e8-Paper.pdf.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 69(6):066138, 2004.

Matthew Landers and Afsaneh Doryab. Deep reinforcement learning verification: A survey. *ACM Comput. Surv.*, 55(14s), 2023. doi: 10.1145/3596444. URL https://doi.org/10.1145/3596444.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5639–5650. PMLR, 13–18 Jul 2020a. URL https://proceedings.mlr.press/v119/laskin20a.html.

Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 19884–19895. Curran Associates, Inc., 2020b. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/e615c82aba461681ade82da2da38004a-Paper.pdf.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PmLR, 2016.

Seungyong Moon, JunYeong Lee, and Hyun Oh Song. Rethinking value function learning for generalization in reinforcement learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=JkEz1fqN3hX.

Buqing Nie, Jingtian Ji, Yangqing Fu, and Yue Gao. Improve robustness of reinforcement learning against observation perturbations via ℓ_∞-lipschitz policy networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 14457–14465, 2024.

Roberta Raileanu and Rob Fergus. Decoupling value and policy for generalization in reinforcement learning. In *International Conference on Machine Learning*, pp. 8787–8798. PMLR, 2021.

Brian C Ross. Mutual information between discrete and continuous data sets. *PloS one*, 9(2): e87357, 2014.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. Pmlr, 2015.

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

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638.

Han Wang, Erfan Miahi, Martha White, Marlos C Machado, Zaheer Abbas, Raksha Kumaraswamy, Vincent Liu, and Adam White. Investigating the properties of neural network representations in reinforcement learning. *Artificial Intelligence*, 330:104100, 2024.

Kaixin Wang, Daquan Zhou, Jiashi Feng, and Shie Mannor. PPG reloaded: An empirical study on what matters in phasic policy gradient. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 36694–36713. PMLR, 23–29 Jul 2023.

Zikang Xiong, Joe Eappen, He Zhu, and Suresh Jagannathan. Defending observation attacks in deep reinforcement learning via detection and denoising. In Massih-Reza Amini, Stéphane Canu, Asja Fischer, Tias Guns, Petra Kralj Novak, and Grigorios Tsoumakas (eds.), *Machine Learning and Knowledge Discovery in Databases*, volume 13715 of *Lecture Notes in Computer Science*, pp. 235–250. Springer, 2023. doi: 10.1007/978-3-031-26409-2_15. ECML PKDD 2022.

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=GY6-6sTvGaf.

Amy Zhang, Rowan Thomas McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=-2FCwDKRREu.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21024–21037. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/f0eb6568ea114ba6e293f903c34d7488-Paper.pdf.

Huan Zhang, Hongge Chen, Duane S. Boning, and Cho-Jui Hsieh. Robust reinforcement learning on state observations with learned optimal adversary. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=sCZbhBvqQaU.

Wancong Zhang, Anthony GX-Chen, Vlad Sobal, Yann LeCun, and Nicolas Carion. Light-weight probing of unsupervised representations for reinforcement learning. *Reinforcement Learning Journal*, 4:1924–1949, 2024.

AI USE STATEMENT

In this work, we used generative AI tools for provide critical ingredients for proving mathematical claims, assist with translation.

We have not used generative AI tools for help develop theoretical models or conceptual frameworks, formulate mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. And generating synthetic data sets is not applicable to this work.

Additionally, we used generative AI tools for create or modify scientific figures or images, create or edit software code and identify relevant literature.

We have reviewed all AI-assisted work. An author carefully verified and manually derived the key steps in the mathematical derivations provided by the artificial intelligence tool. For content that required the use of AI for translation, such as technical terms, the author reviewed relevant literature and determined the appropriate translation methods. The scientific charts modified by AI were also carefully checked by an author to ensure data accuracy. After the project was completed, an author used the AI tool to assist in reorganizing the project code, aiming to make the code concise, save memory, and improve readability, and the code repository was also inspected. We also used the AI tool to assist in verifying the references to avoid errors, and this part of the work was completed by an author.

We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Experimental hyperparameters and network configurations are provided in Appendix B.1. Observation-corruption protocols are detailed in Appendix B.2, while representation-analysis procedures, diagnostic definitions, and aggregation conventions are described in Appendices B.3 and C.3.

We submit the project code as supplementary material for Procgen, DMC, and Robust Gymnasium. The code covers virtual-environment setup and dependencies, agent training and checkpoint generation, clean and corrupted evaluation, corruption configuration, and representation analysis.

APPENDIX

A THEORETICAL DERIVATIONS

The derivations follow the order of Section 4: information changes, local output sensitivity, and the policy-divergence return bound.

A.1 DERIVATIONS FOR THE INFORMATION-THEORETIC ANALYSIS

We prove Lemma 1 and Corollaries 1 and 2. All information statements concern a fixed corruption type and a fixed paired evaluation distribution. The observation model is

$$o_t^T = \mathcal{T}_T(o_t^0; \xi_t), \tag{18}$$

MI is defined in Eq. 4; the identities below assume the relevant terms are finite. Let $Y \in \{A_{\text{ref}}, G_{\text{ref}}\}$ and $R \in \{S, A, C\}$. Assume $p(o^T \mid o^0, y) = p(o^T \mid o^0)$ and $p(z_R^T \mid o^T, o^0, y) = p(z_R^T \mid o^T)$. For the clean representation, also assume $p(z_R^0 \mid o^0, y) = p(z_R^0 \mid o^0)$. These are conditional-independence requirements on the analyzed variables, not consequences of the name of a corruption operator. For history-dependent channels, the input must include sufficient history, or the statements apply only when these requirements hold for the chosen observation variables.

The first conditional MI in Eq. 7 measures information lost through the observation channel; the second measures information present in the corrupted observation but absent from its representation. The clean-encoder term is subtracted in the representation-shift identity. Consequently, the data processing inequality does not imply

$$I(Y; Z_R^T) \le I(Y; Z_R^0). \tag{19}$$

An increase in representation MI is therefore compatible with the stated data processing inequality. The observation-channel term is common across architectures only when their analyzed joint distribution of (Y, O^0, O^T) is the same.

Information decomposition under observation corruption. Under the conditional-independence assumption

$$p(o^T \mid o^0, y) = p(o^T \mid o^0), \tag{20}$$

we have

$$I(Y; O^T \mid O^0) = 0. \tag{21}$$

By the chain rule,

$$\begin{aligned} I(Y; O^0, O^T) &= I(Y; O^0) + I(Y; O^T \mid O^0) \\ &= I(Y; O^0). \end{aligned} \tag{22}$$

Applying the chain rule in the reverse order gives

$$I(Y; O^0, O^T) = I(Y; O^T) + I(Y; O^0 \mid O^T). \tag{23}$$

Therefore,

$$I(Y; O^0) = I(Y; O^T) + I(Y; O^0 \mid O^T). \tag{24}$$

Similarly, because the representation is generated only from the corrupted observation,

$$p(z_R^T \mid o^T, o^0, y) = p(z_R^T \mid o^T), \tag{25}$$

which implies

$$I(Y; Z_R^T \mid O^T) = 0. \tag{26}$$

Hence,

$$\begin{aligned} I(Y; O^T, Z_R^T) &= I(Y; O^T) + I(Y; Z_R^T \mid O^T) \\ &= I(Y; O^T). \end{aligned} \tag{27}$$

Reversing the chain-rule order gives

$$I(Y; O^T, Z_R^T) = I(Y; Z_R^T) + I(Y; O^T \mid Z_R^T), \tag{28}$$

and therefore

$$I(Y; O^T) = I(Y; Z_R^T) + I(Y; O^T \mid Z_R^T). \tag{29}$$

Nonnegativity of the conditional MI terms in Eqs. 24 and 29 also yields Lemma 1.

Substituting Eq. 29 into Eq. 24 yields

$$I(Y; O^0) - I(Y; Z_R^T) = I(Y; O^0 \mid O^T) + I(Y; O^T \mid Z_R^T), \tag{30}$$

which proves Corollary 1.

Clean-to-corrupted representation information shift. For the clean representation, the assumption

$$p(z_R^0 \mid o^0, y) = p(z_R^0 \mid o^0) \tag{31}$$

implies

$$I(Y; Z_R^0 \mid O^0) = 0. \tag{32}$$

Following the same chain-rule argument,

$$I(Y; O^0) = I(Y; Z_R^0) + I(Y; O^0 \mid Z_R^0). \tag{33}$$

Equivalently,

$$I(Y; O^0) - I(Y; Z_R^0) = I(Y; O^0 \mid Z_R^0). \tag{34}$$

From Corollary 1,

$$I(Y; O^0) - I(Y; Z_R^T) = I(Y; O^0 \mid O^T) + I(Y; O^T \mid Z_R^T). \tag{35}$$

Subtracting Eq. 34 from the above relation gives

$$\begin{aligned} I(Y; Z_R^0) - I(Y; Z_R^T) = I(Y; O^0 \mid O^T) + I(Y; O^T \mid Z_R^T) \\ - I(Y; O^0 \mid Z_R^0), \end{aligned} \tag{36}$$

which proves Corollary 2.

Specialization shifts. The explicit coordinate-wise definitions used in the main text are

$$\Delta S_A^T = S_A^T - S_A^0, \qquad \Delta S_G^T = S_G^T - S_G^0. \tag{37}$$

$$\begin{aligned} \Delta I_A^T(Z_R) &= I_A(Z_R^T) - I_A(Z_R^0), \\ \Delta I_G^T(Z_R) &= I_G(Z_R^T) - I_G(Z_R^0). \end{aligned} \tag{38}$$

Subtracting the clean specialization scores from their corrupted counterparts gives Eq. 10. Because each score is a difference between branches, its decrease can be caused by information loss in the preferred branch, information gain in the other branch, or both.

A.2 Proof of Policy Sensitivity for a Linear Softmax Head

For fixed network parameters, the feature and logit changes are

$$\Delta U_A^T = U_A^T - U_A^0, \qquad \Delta U_C^T = U_C^T - U_C^0. \tag{39}$$

$$\Delta \ell^T = \ell^T - \ell^0 = W_A \Delta U_A^T. \tag{40}$$

With the centering projection

$$P = I - \frac{1}{K}\mathbf{1}\mathbf{1}^\top. \tag{41}$$

softmax invariance to common logit shifts gives

$$P W_A \Delta U_A^T = 0 \quad \Longrightarrow \quad \pi^T = \pi^0. \tag{42}$$

We prove Lemma 2. Let

$$d = \ell^T - \ell^0 = W_A \Delta U_A^T. \tag{43}$$

Define the log-sum-exp function

$$F(\ell) = \log \sum_{a=1}^{K} \exp(\ell_a). \tag{44}$$

Its gradient is the softmax,

$$\nabla F(\ell) = \pi, \tag{45}$$

and its Hessian is

$$\nabla^2 F(\ell) = \mathrm{diag}(\pi) - \pi\pi^\top. \tag{46}$$

Using $\log \pi_a = \ell_a - F(\ell)$, the KL divergence between the clean and corrupted softmax distributions can be written as

$$D_{\mathrm{KL}}\left(\pi^0 \| \pi^T\right) = F(\ell^0 + d) - F(\ell^0) - \nabla F(\ell^0)^\top d. \tag{47}$$

This is the Bregman divergence generated by the log-sum-exp function.

For the centering projection

$$P = I - \frac{1}{K}\mathbf{1}\mathbf{1}^\top, \tag{48}$$

we can decompose

$$d = Pd + \bar{d}\,\mathbf{1}, \qquad \bar{d} = \frac{1}{K}\mathbf{1}^\top d. \tag{49}$$

Softmax is invariant to a common translation of all logits, so

$$\mathrm{softmax}(\ell^0 + d) = \mathrm{softmax}(\ell^0 + Pd). \tag{50}$$

Hence the KL divergence depends only on Pd.

To bound the Hessian, for any vector x,

$$x^\top \nabla^2 F(\ell) x = \sum_i \pi_i x_i^2 - \left(\sum_i \pi_i x_i\right)^2, \tag{51}$$

which is the variance of the values $\{x_i\}$ under the categorical distribution π. By the standard variance bound,

$$x^\top \nabla^2 F(\ell) x \le \frac{1}{4}\left(\max_i x_i - \min_i x_i\right)^2. \tag{52}$$

Moreover,

$$\max_i x_i - \min_i x_i \le \sqrt{2}\|x\|_2, \tag{53}$$

and therefore

$$x^\top \nabla^2 F(\ell) x \le \frac{1}{2}\|x\|_2^2. \tag{54}$$

Thus,

$$\left\|\nabla^2 F(\ell)\right\|_2 \le \frac{1}{2}. \tag{55}$$

Applying the integral form of the second-order Taylor remainder to Eq. 47 with the effective displacement Pd gives

$$\begin{aligned} D_{\mathrm{KL}}\left(\pi^0 \| \pi^T\right) &= \int_0^1 (1-\tau)(Pd)^\top \nabla^2 F\left(\ell^0 + \tau Pd\right)(Pd)\, d\tau \\ &\le \int_0^1 (1-\tau)\frac{1}{2}\|Pd\|_2^2\, d\tau \\ &= \frac{1}{4}\|Pd\|_2^2. \end{aligned} \tag{56}$$

Finally, substituting

$$Pd = PW_A \Delta U_A^T \tag{57}$$

yields

$$D_{\mathrm{KL}}\left(\pi^0 \| \pi^T\right) \le \frac{1}{4}\left\|PW_A \Delta U_A^T\right\|_2^2, \tag{58}$$

which proves Lemma 2.

A.3 Derivation of the Greedy-Action Stability Condition

Let the clean greedy action be unique:

$$a^* = \arg\max_a \ell_a^0 \tag{59}$$

Let $a^T \in \arg\max_a \ell_a^T$ denote a corrupted greedy action. The clean logit margin is

$$m^0 = \ell_{a^*}^0 - \max_{a \neq a^*} \ell_a^0. \tag{60}$$

A sufficient stability condition is

$$2 \left\| P \Delta \ell^T \right\|_\infty < m^0 \quad \Longrightarrow \quad a^T = a^*. \tag{61}$$

For any action $a \neq a^*$,

$$\ell_{a^*}^T - \ell_a^T = \ell_{a^*}^0 - \ell_a^0 + \Delta \ell_{a^*}^T - \Delta \ell_a^T. \tag{62}$$

Adding or subtracting a common logit shift does not affect pairwise differences, so

$$\Delta \ell_{a^*}^T - \Delta \ell_a^T = (P \Delta \ell^T)_{a^*} - (P \Delta \ell^T)_a. \tag{63}$$

Hence,

$$\left| \Delta \ell_{a^*}^T - \Delta \ell_a^T \right| \leq 2 \left\| P \Delta \ell^T \right\|_\infty. \tag{64}$$

If

$$2 \left\| P \Delta \ell^T \right\|_\infty < m^0, \tag{65}$$

then for every $a \neq a^*$,

$$\begin{aligned} \ell_{a^*}^T - \ell_a^T &> \ell_{a^*}^0 - \ell_a^0 - m^0 \\ &\geq 0. \end{aligned} \tag{66}$$

Therefore a^* remains the unique greedy action under corruption,

$$a^T = a^*. \tag{67}$$

A.4 Continuous-Action and Value-Head Relations

For a diagonal Gaussian Actor,

$$\pi(a \mid U_A) = \mathcal{N}\left(\mu(U_A), \mathrm{diag}(\sigma^2(U_A))\right). \tag{68}$$

a linear mean head gives

$$\mu = W_\mu U_A + b_\mu, \tag{69}$$

$$\Delta \mu^T = W_\mu \Delta U_A^T. \tag{70}$$

If the paired covariance is unchanged and positive definite, $\Sigma^0 = \Sigma^T = \Sigma \succ 0$, the Gaussian KL formula reduces to

$$D_{\mathrm{KL}}\left(\pi^0 \| \pi^T\right) = \frac{1}{2} (\Delta \mu^T)^\top \Sigma^{-1} \Delta \mu^T, \tag{71}$$

Substitution yields

$$D_{\mathrm{KL}}\left(\pi^0 \| \pi^T\right) = \frac{1}{2} (\Delta U_A^T)^\top W_\mu^\top \Sigma^{-1} W_\mu \Delta U_A^T. \tag{72}$$

This is a property of the Gaussian distributions before any action clipping. If the covariance changes, both mean and covariance terms contribute to KL; mean displacement alone is insufficient.

For a linear Critic,

$$\hat{V} = w_C^\top U_C + b_C. \tag{73}$$

Eq. 13 follows directly by subtracting the clean output from the corrupted output, and therefore

$$w_C^\top \Delta U_C^T = 0 \quad \Longrightarrow \quad \hat{V}^T = \hat{V}^0. \tag{74}$$

The value-head projection measures output change and does not identify value-prediction error without a matched target.

A.5 Proof of the Policy-Divergence Value Bound

Behavior policies and assumptions. The clean and corrupted policies must be defined on the same Markov state space and use the same environment transition and reward functions. Let $q(\cdot \mid o)$ denote the actual action-selection kernel used during evaluation. For a memoryless observation map $o(s)$ and exogenous corruption randomness, the induced behavior policies are

$$\pi_0(\cdot \mid s) = q(\cdot \mid o(s)), \qquad \pi_T(\cdot \mid s) = \mathbb{E}_\xi[q(\cdot \mid \mathcal{T}_T(o(s); \xi))]\,. \tag{75}$$

For greedy evaluation, q is the point mass on the selected action; it is not the softmax distribution used to diagnose the network. For temporally coupled corruptions or history-dependent policies, the state is augmented with sufficient corruption memory and observation history, with a common initial distribution for the comparison. The following statements then apply to stationary behavior policies on that common augmented space. The signed return difference is

$$D_{\mathrm{ret}}^T = J^0 - J^T. \tag{76}$$

We use the uniform behavior-policy discrepancy δ_T defined in Eq. 16. For probability measures p and q on the action space, define $D_{\mathrm{TV}}(p, q) = \sup_{E \in \mathcal{B}(\mathcal{A})} |p(E) - q(E)|$. When masses or densities exist with respect to a common dominating measure, this equals one half of their L^1 distance. For every bounded measurable g, $|\int g\, d(p - q)| \le 2\|g\|_\infty D_{\mathrm{TV}}(p, q)$.

We prove Lemma 3. Let T_π denote the Bellman operator associated with policy π, and let P_π denote the corresponding state-transition operator. Using the Bellman equations,

$$V^{\pi_T} = T_{\pi_T} V^{\pi_T}, \qquad V^{\pi_0} = T_{\pi_0} V^{\pi_0}. \tag{77}$$

We have

$$\begin{aligned} V^{\pi_T} - V^{\pi_0} &= T_{\pi_T} V^{\pi_T} - V^{\pi_0} \\ &= \gamma P_{\pi_T} \left(V^{\pi_T} - V^{\pi_0}\right) + T_{\pi_T} V^{\pi_0} - V^{\pi_0}. \end{aligned} \tag{78}$$

Therefore,

$$V^{\pi_T} - V^{\pi_0} = \left(I - \gamma P_{\pi_T}\right)^{-1} \left(T_{\pi_T} V^{\pi_0} - V^{\pi_0}\right). \tag{79}$$

For each state s,

$$T_{\pi_T} V^{\pi_0}(s) - V^{\pi_0}(s) = \int_{\mathcal{A}} Q^{\pi_0}(s, a)\, [\pi_T(da \mid s) - \pi_0(da \mid s)]\,. \tag{80}$$

This measure notation covers discrete, continuous, and deterministic behavior policies.

Because

$$|r(s, a)| \le R_{\max}, \tag{81}$$

the action-value function satisfies

$$\|Q^{\pi_0}\|_\infty \le \sum_{t=0}^{\infty} \gamma^t R_{\max} = \frac{R_{\max}}{1-\gamma}. \tag{82}$$

Hence,

$$\begin{aligned} |T_{\pi_T} V^{\pi_0}(s) - V^{\pi_0}(s)| &\le 2\|Q^{\pi_0}\|_\infty D_{\mathrm{TV}}(\pi_T(\cdot \mid s), \pi_0(\cdot \mid s)) \\ &\le \frac{2R_{\max}}{1-\gamma} D_{\mathrm{TV}}(\pi_T(\cdot \mid s), \pi_0(\cdot \mid s)) \,. \end{aligned} \tag{83}$$

Taking the supremum over states gives

$$\|T_{\pi_T} V^{\pi_0} - V^{\pi_0}\|_\infty \le \frac{2R_{\max}}{1-\gamma} \delta_T. \tag{84}$$

The inverse operator in Eq. 79 can be written using the Neumann series,

$$(I - \gamma P_{\pi_T})^{-1} = \sum_{k=0}^{\infty} \gamma^k P_{\pi_T}^k. \tag{85}$$

Because a Markov transition operator is non-expansive in the sup norm,

$$\left\| P_{\pi_T}^k \right\|_\infty \le 1, \tag{86}$$

and therefore

$$\left\| (I - \gamma P_{\pi_T})^{-1} \right\|_\infty \le \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}. \tag{87}$$

Combining Eqs. 79, 84, and 87 gives

$$\begin{aligned} \|V^{\pi_T} - V^{\pi_0}\|_\infty &\le \frac{1}{1-\gamma} \cdot \frac{2R_{\max}}{1-\gamma} \delta_T \\ &= \frac{2R_{\max}}{(1-\gamma)^2} \delta_T \\ &= \frac{2R_{\max}}{(1-\gamma)^2} \sup_s D_{\mathrm{TV}} \left(\pi_T(\cdot \mid s), \pi_0(\cdot \mid s)\right). \end{aligned} \tag{88}$$

This proves Lemma 3. Under the common initial distribution ρ_0,

$$|J^T - J^0| = \left| \int (V^{\pi_T}(s) - V^{\pi_0}(s)) \, \rho_0(ds) \right| \le \|V^{\pi_T} - V^{\pi_0}\|_\infty,$$

which gives the stated return bound.

B Experimental Settings and Measurement Conventions

This appendix expands Section 5.1, with training parameters, corruption operators, and the aggregation used by the supplementary results.

B.1 Hyperparameters

Table 4: Experimental parameters for Procgen, DMC, and Robust Gymnasium.

Parameter	**Procgen**	**DMC**	**Robust Gymnasium**
Training and rollout			
Total training steps	25,000,000	5,000,000	3,000,000
Number of environments	64	8	8
Rollout length	256	512	256
Minibatches per epoch	8	8	32
Minibatch size	2,048	512	64
Discount factor γ	0.999	0.99	0.99
GAE parameter λ_{GAE}	0.95	0.95	0.95
PPO clipping coefficient ϵ_{clip}	0.2	0.2	0.2
Optimization			
Learning rate	5×10^{-4}	5×10^{-4}	3×10^{-4}
Adam ϵ	10^{-5}	10^{-8}	10^{-5}
Maximum gradient norm	0.5	0.5	0.5
Value-loss coefficient	0.5	0.5	0.5
Entropy coefficient	0.01	0.01	0
Encoder			
Architecture	IMPALA CNN	$4 \times \mathrm{Conv}(32, 3, 2)$	MLP $(64, 64)$
Encoder details	Channels $(16, 32, 32)$; FC 256, ReLU	FC 256, ReLU	Tanh activations

Architecture details. The Procgen encoder comprises three stages with channel widths $(16, 32, 32)$. Each stage contains a 3×3 convolution, 3×3 max pooling with stride 2, and two residual blocks, each containing two 3×3 convolutions with pre-activation ReLU. The final stage is followed by ReLU, flattening, and a 256-unit fully connected layer with ReLU. For DMC, $\mathrm{Conv}(C, k, s)$ denotes a convolution with C output channels, kernel size $k \times k$, and stride s. All four convolutions use valid padding and ReLU, followed by flattening and a 256-unit fully connected layer with ReLU. The Robust Gymnasium encoder has two 64-unit fully connected layers, each followed by Tanh. Shared models use one encoder for the policy and value heads; decoupled models use separate encoders of the same architecture.

B.2 Observation Perturbation Protocol

We evaluate vector-observation perturbations using Robust Gymnasium (Gu et al., 2025). Gaussian noise and fixed shift use its corresponding state-observation branches. The uniform-configuration condition is implemented in our environment adapter, as detailed below. For Procgen and DeepMind Control (DMC), every image-corruption condition uses one fixed severity setting. The operators act on RGB observations. DMC frame stacks are split into their constituent RGB frames before corruption and reassembled afterward. Image operations use the 0–255 floating-point pixel domain, with conversion to 8-bit values where required by JPEG encoding. In the algorithms below, O^0 and O^T denote the clean and corrupted image arrays in the pixel domain; B is a local intermediate array. The map $\mathcal{R}_{O^0}(B)$ denotes clipping B to $[0, 255]$ and restoring the range and data type of O^0; integer-valued output is rounded before conversion. Figure 7 shows the 16 image-corruption

conditions using a clean Procgen Jumper observation as the reference frame. The procedure for each perturbation is specified by the pseudocode in its corresponding table below.

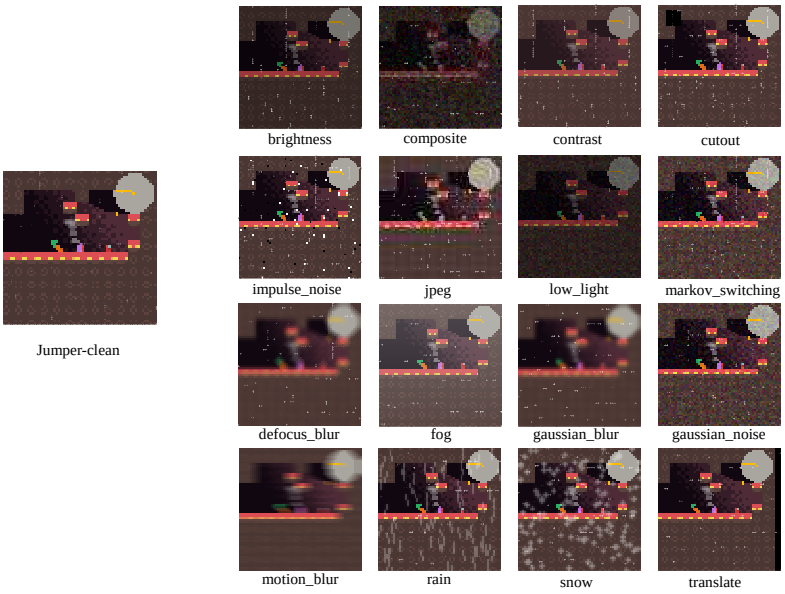

Figure 7: Clean Procgen Jumper observation (left) and its 16 corrupted views (right). The same image-corruption operators are used in the Procgen and DMC experiments.

B.2.1 Pixelwise Gaussian Noise

Pixelwise Gaussian noise affects every color channel at every spatial location. Independent, zero-mean samples with standard deviation $\sigma = 12$ are added on the 0–255 pixel scale. The operator models unstructured noise in image acquisition.

Table 5: Pixelwise Gaussian noise.

Input: RGB image O^0, random seed ξ.
Output: Perturbed image O^T.

1. Initialize the random generator with ξ.
2. Independently draw $\varepsilon_{h,w,k} \sim \mathcal{N}(0, 12^2)$ for every pixel (h, w) and color channel k.
3. Set $B_{h,w,k} = O^0_{h,w,k} + \varepsilon_{h,w,k}$.
4. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.2 Impulse Noise

Impulse noise replaces selected spatial pixels with black or white values. Each pixel is selected with probability 0.02; its three color channels share both the selection event and the replacement value. The resulting sparse extrema represent isolated observation errors.

Table 6: Impulse noise.

Input: RGB image O^0, random seed ξ.
Output: Perturbed image O^T.

1. Initialize the random generator with ξ and set $B = O^0$.
2. For each spatial position (h, w), draw $m_{h,w} \sim \text{Bernoulli}(0.02)$.
3. If $m_{h,w} = 1$, select $v_{h,w}$ uniformly from $\{0, 255\}$ and set $B_{h,w,:} = v_{h,w}$.
4. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.3 GAUSSIAN BLUR

Gaussian blur acts across the complete image. It suppresses fine texture and sharp boundaries by convolving each color channel with a normalized Gaussian kernel of standard deviation 1.0 pixel. The implementation performs separable horizontal and vertical convolutions with reflective boundaries.

Table 7: Gaussian blur.

Input: RGB image O^0.
Output: Blurred image O^T.

1. Construct a normalized one-dimensional Gaussian kernel g with standard deviation 1.0 pixel.
2. Convolve each channel of O^0 horizontally and vertically with g, using reflective boundaries, to obtain B.
3. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.4 DEFOCUS BLUR

Defocus blur acts on the full frame through convolution with a uniform disk kernel of radius 2 pixels. The disk-shaped point-spread function reduces edge sharpness and fine texture while differing from the Gaussian kernel used for Gaussian blur.

Table 8: Defocus blur.

Input: RGB image O^0.
Output: Defocused image O^T.

1. Construct a kernel with equal nonzero weights within a disk of radius 2 pixels.
2. Normalize the kernel weights to sum to one and convolve each channel of O^0 to obtain B.
3. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.5 MOTION BLUR

Motion blur affects the entire image through a directional line kernel of width 7 pixels. One of the horizontal, vertical, and two diagonal orientations is selected for the processed frame. Convolution with this kernel approximates directional smear during image acquisition.

Table 9: Motion blur.

Input: RGB image O^0, random seed ξ.
Output: Blurred image O^T.

1. Initialize the random generator with ξ and sample one of four line orientations uniformly.
2. Construct a 7×7 kernel with equal nonzero weights along the selected line; normalize the weights to sum to one.
3. Convolve O^0 with the kernel to obtain B.
4. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.6 BRIGHTNESS REDUCTION

Brightness reduction scales every channel and spatial location by 0.70. This deterministic transformation represents a uniform reduction in image intensity. The evaluated condition uses the darkening direction.

Table 10: Brightness reduction.

Input: RGB image O^0.
Output: Darkened image O^T.

1. Set $B_{h,w,k} = 0.70 O^0_{h,w,k}$ for every pixel and color channel.
2. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.7 CONTRAST REDUCTION

Contrast reduction affects the full image while preserving the spatial mean of each color channel. Pixel deviations from the corresponding mean are multiplied by 0.65, reducing intensity differences within the current frame. The means are computed from that frame, not from the evaluation dataset.

Table 11: Contrast reduction.

Input: RGB image O^0.
Output: Contrast-reduced image O^T.

1. For each channel k, compute $\bar{O}^0_k = (HW)^{-1} \sum_{h,w} O^0_{h,w,k}$.
2. Set $B_{h,w,k} = \bar{O}^0_k + 0.65(O^0_{h,w,k} - \bar{O}^0_k)$.
3. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.8 LOW-LIGHT CORRUPTION

Low-light corruption combines global intensity attenuation with independent pixelwise noise. Every pixel value is multiplied by 0.55, after which zero-mean Gaussian noise with standard deviation 8 is added on the 0–255 scale. The combined operation represents reduced illumination accompanied by measurement noise.

Table 12: Low-light corruption.

Input: RGB image O^0, random seed ξ.
Output: Low-light image O^T.

1. Initialize the random generator with ξ.
2. Independently draw $\varepsilon_{h,w,k} \sim \mathcal{N}(0, 8^2)$ for every pixel and channel.
3. Set $B_{h,w,k} = 0.55 O^0_{h,w,k} + \varepsilon_{h,w,k}$.
4. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.9 FOG

Fog blends the entire RGB frame with a gray-white reference color $\mathbf{f} = (200, 200, 200)$. The mixture is strongest at the top of the image, where its coefficient is 0.34, and decreases linearly toward the bottom. This spatially varying veil reduces scene contrast and visibility.

Table 13: Fog corruption.

Input: RGB image O^0.
Output: Fogged image O^T.

1. Define $v(h)$ by linear interpolation from 1 at the top row to 0.35 at the bottom row.
2. Set $B_{h,w,:} = [1 - 0.34v(h)]O^0_{h,w,:} + 0.34v(h)\mathbf{f}$, where $\mathbf{f} = (200, 200, 200)$.
3. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.10 RAIN

Rain places 80 short streaks, each of nominal length 5 pixels, at random locations. Their orientations are near vertical. A lightly smoothed streak mask is blended with white using coefficient 0.28, producing sparse local obstruction rather than a uniform change across the frame.

Table 14: Rain corruption.

Input: RGB image O^0, random seed ξ.
Output: Rain-corrupted image O^T.

1. Initialize a zero-valued spatial mask M and seed the random generator with ξ.
2. Draw 80 near-vertical line segments of nominal length 5 pixels at random locations in M.
3. Smooth M with a Gaussian kernel of standard deviation 0.45, then clip it to $[0, 1]$.
4. Set $B_{h,w,:} = [1 - 0.28M_{h,w}]O^0_{h,w,:} + 0.28M_{h,w}(255, 255, 255)$.
5. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.11 SNOW

Snow places 130 bright, approximately circular flakes at random locations. Each flake has radius 1 pixel in this condition. The resulting mask is lightly smoothed and blended with white using coefficient 0.45, creating spatially localized loss of visibility.

Table 15: Snow corruption.

Input: RGB image O^0, random seed ξ.
Output: Snow-corrupted image O^T.

1. Initialize a zero-valued spatial mask M and seed the random generator with ξ.
2. Draw 130 bright disks of radius 1 pixel at random locations in M.
3. Smooth M with a Gaussian kernel of standard deviation 0.55, then clip it to $[0, 1]$.
4. Set $B_{h,w,:} = [1 - 0.45M_{h,w}]O^0_{h,w,:} + 0.45M_{h,w}(255, 255, 255)$.
5. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.12 IMAGE TRANSLATION

Image translation shifts the complete frame by 3 pixels horizontally, vertically, or along both axes. One of eight signed displacements is sampled per processed image. Newly exposed locations are filled with black, changing the apparent position of visual content without changing the environment's physical configuration.

Table 16: Image translation.

Input: RGB image O^0, random seed ξ.
Output: Translated image O^T.

1. Initialize the random generator with ξ.
2. Sample $(\Delta x, \Delta y)$ uniformly from $\{-3, 0, 3\}^2 \setminus \{(0, 0)\}$.
3. Shift O^0 by $(\Delta x, \Delta y)$, assigning zero to newly exposed pixels, to obtain B.
4. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.13 SQUARE OCCLUSION

Square occlusion removes visual information from one randomly positioned region. The selected square has side length 8 pixels, limited by the image dimensions if necessary. All three color channels inside the square are set to zero; locations outside it remain unchanged.

Table 17: Square occlusion.

Input: RGB image O^0, random seed ξ.
Output: Occluded image O^T.

1. Set $q = \min(8, H, W)$.
2. Sample a top-left corner uniformly among positions at which a $q \times q$ square fits inside the image.
3. Copy O^0 to B and set all three channels inside the selected square to zero.
4. Return $O^T = \mathcal{R}_{O^0}(B)$.

B.2.14 JPEG COMPRESSION

JPEG compression affects the complete frame through lossy encoding and decoding at quality factor 40. The resulting quantization and block-related artifacts degrade local image structure.

Table 18: JPEG compression.

Input: RGB image O^0.
Output: Compressed image O^T.
1. Convert O^0 to valid 8-bit RGB values.
2. Encode and decode the image at JPEG quality 40.
3. Return the reconstructed image as O^T.

B.2.15 SEQUENTIAL COMPOSITE CORRUPTION

The composite condition applies low-light corruption, JPEG compression, and pixelwise Gaussian noise in this fixed order. Each operator uses the parameters specified above and takes the preceding operator's output as its input.

Table 19: Sequential composite corruption.

Input: RGB image O^0, random seed ξ.
Output: Composite-corrupted image O^T.
1. Initialize the random generator with ξ and reuse it for the stochastic components.
2. Apply low-light corruption to O^0, obtaining B_1.
3. Apply JPEG compression at quality 40 to B_1, obtaining B_2.
4. Add independent pixelwise Gaussian noise with standard deviation 12 to B_2, obtaining B_3.
5. Return $O^T = \mathcal{R}_{O^0}(B_3)$.

B.2.16 MARKOV-SWITCHING CORRUPTION

This condition applies one active operator per frame from Gaussian noise, low-light corruption, JPEG compression, fog, and defocus blur. In episode-based evaluation, an update is attempted with probability 0.08 per observation or underlying frame. Procgen closed-loop evaluation selects a different operator on an update, whereas DMC redraws from all five candidates and can retain the current operator. Procgen's offline paired observations select an operator independently for each sample and do not form a temporal Markov chain. The active operator determines which image regions and values are affected.

Table 20: Markov-switching corruption in episode-based evaluation.

Input: RGB frame O_t^0, previous active operator κ_{t-1} when $t > 0$, random seed ξ_t.
Output: Perturbed frame O_t^T, active operator κ_t.
1. At the first frame, sample κ_0 uniformly from the five candidate operators. At subsequent frames, update the active operator with probability 0.08 according to the experiment's transition rule; otherwise set $\kappa_t = \kappa_{t-1}$.
2. Apply only κ_t to O_t^0, using the parameters of its corresponding subsection, to obtain B_t.
3. Return $O_t^T = \mathcal{R}_{O_t^0}(B_t)$ and κ_t.

B.2.17 GAUSSIAN VECTOR-OBSERVATION NOISE

The Robust Gymnasium Gaussian condition adds one zero-mean Gaussian scalar to the complete observation vector returned after each environment transition. Its standard deviation is 0.005 in the original observation units. The same scalar is broadcast to all dimensions, so the perturbations of different components are fully correlated within a step. No additional clipping is applied by this perturbation.

Table 21: Gaussian vector-observation noise.

Input: Post-transition observation $o^0_{t+1} \in \mathbb{R}^d$, step-specific random seed ξ_t.
Output: Perturbed observation o^T_{t+1}.
1. Initialize the random generator with ξ_t.
2. Draw one scalar $\varepsilon_t \sim \mathcal{N}(0, 0.005^2)$.
3. Return $o^T_{t+1} = o^0_{t+1} + \varepsilon_t \mathbf{1}_d$.

B.2.18 Fixed Vector-Observation Shift

The Robust Gymnasium shift condition adds a deterministic offset of $+0.005$ to every component of the observation vector returned after each transition. This operation represents a persistent positive measurement bias. Because observation components can have different numerical scales, the same absolute offset need not have the same relative effect in every dimension.

Table 22: Fixed vector-observation shift.

Input: Post-transition observation $o^0_{t+1} \in \mathbb{R}^d$.
Output: Shifted observation o^T_{t+1}.
1. Set the offset to $\delta = 0.005$.
2. Return $o^T_{t+1} = o^0_{t+1} + \delta \mathbf{1}_d$.

B.2.19 Uniform-Configuration Vector-Observation Shift

The configured bounds `uniform_low` and `uniform_high` are both set to 0.4, yielding a deterministic $+0.4$ shift in the original observation units. The adapter disables the environment's native state perturbation for this condition and adds the offset to the returned observation.

Table 23: Uniform-configuration vector-observation shift.

Input: Post-transition observation $o^0_{t+1} \in \mathbb{R}^d$.
Output: Shifted observation o^T_{t+1}.
1. Set the offset to $u_t = 0.4$.
2. Return $o^T_{t+1} = o^0_{t+1} + u_t \mathbf{1}_d$.

B.3 Evaluation and Aggregation

Scope and aggregation. The return comparisons use the task set in Tables 3 and 24: 5 Procgen tasks, 4 reported DMC tasks, and 9 Robust Gymnasium tasks. The return plots average the evaluated corruption conditions within each task and keep its original reward scale; no cross-task mean in raw reward units is plotted. All plotted markers are point estimates, without significance annotations.

For the information plots, bars denote aggregation over tasks and corruption conditions after seed averaging. Specialization erosion is a ratio of macro averages, $-\overline{\Delta S^T_Y}/\overline{S^0_Y}$. In contrast, individual-channel MI loss is the average of within-task and within-condition ratios, $\overline{1 - I_Y(Z^T_R)/I_Y(Z^0_R)}$. Here $Y \in \{A, G\}$ specifies the information channel. The same averaging-of-ratios convention is used for the return and observation-proxy retention entries. These statistics cannot be interchanged by dividing the rounded macro means in Table 28. Erosion of a positive clean gap can exceed 100% when the aggregate corrupted gap becomes negative, without implying negative MI. Negative erosion indicates an increase in the gap.

Observation proxy and associations. Procgen representation MI is estimated in branch-specific eight-dimensional PCA coordinates fitted on clean data and then fixed across clean and corrupted inputs. These estimates characterize the processed features and are not exact MI values of the full neural representations; the geometry diagnostics instead use the original head-input features. For the Procgen observation diagnostic, $X^0 = P_{\text{clean}}(O^0)$ and $X^T = P_{\text{clean}}(O^T)$ use the same eight-dimensional PCA projection fitted to clean observations. The reported estimator uses $k = 5$. In

Figure 5, each architecture contributes 16 corruption-category points; output statistics and within-task proportional return losses are averaged across tasks for each category. Table 31 separately reports the vector-task associations.

C Supplementary Experimental Results

The results follow Section 5: control performance, information specialization, representation-to-output diagnostics, cross-level associations, and additional training methods.

C.1 Control Performance under Observation Corruption (RQ1)

Figure 3 in the main text shows Heist, StarPilot, CoinRun, and Jumper. Figures 8–10 report the remaining 14 tasks: BigFish, the four reported DMC tasks, and all 9 Robust Gymnasium tasks. Together, these figures cover the 18 tasks in Tables 3 and 24.

Table 24: **Clean and corrupted episode returns in DMC and Robust Gymnasium.** S and D denote Shared PPO and Decoupled PPO. Corrupted returns average the evaluated corruption conditions within each task. Higher values are better, including for negative-return tasks. Original reward units are retained, and comparisons are made within tasks. DMC refers to the reported four-task subset. The five Procgen tasks are reported in Table 3.

	Clean		Corrupted	
Task	$\widehat{J}_S^0$	$\widehat{J}_D^0$	$\widehat{J}_S^T$	$\widehat{J}_D^T$
DMC				
cartpole_swingup	568.46	813.12	309.64	406.52
walker_stand	501.40	930.68	426.77	887.94
walker_walk	610.94	751.06	555.95	659.40
walker_run	288.35	381.92	254.65	324.53
Robust Gymnasium				
Ant-v4	2237.70	4094.26	1709.72	2950.18
HalfCheetah-v4	2751.81	3009.37	1707.28	1947.67
Hopper-v4	2935.70	3370.02	1841.22	1650.67
HumanoidStandup-v4	79937.53	116256.75	73281.57	59238.55
InvertedDoublePendulum-v4	2366.04	8407.34	105.60	67.71
InvertedPendulum-v4	965.20	1000.00	546.39	568.76
Pusher-v5	−34.41	−23.09	−52.28	−52.30
Reacher-v4	−5.39	−3.80	−13.29	−9.87
Walker2d-v4	2838.26	4205.87	1675.02	2870.76

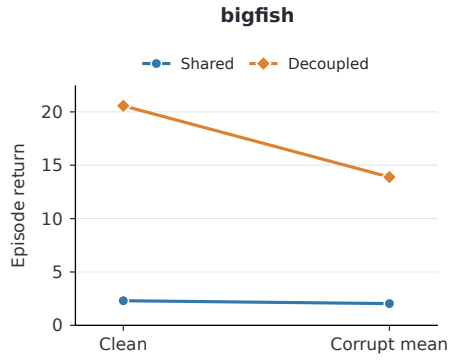

Figure 8: **Clean and corrupted episode returns in BigFish, the remaining Procgen task.** Each panel compares Shared PPO and Decoupled PPO in the task's original reward units. Corrupted returns average the evaluated corruption conditions. Markers show point estimates; higher values are better, including for negative returns.

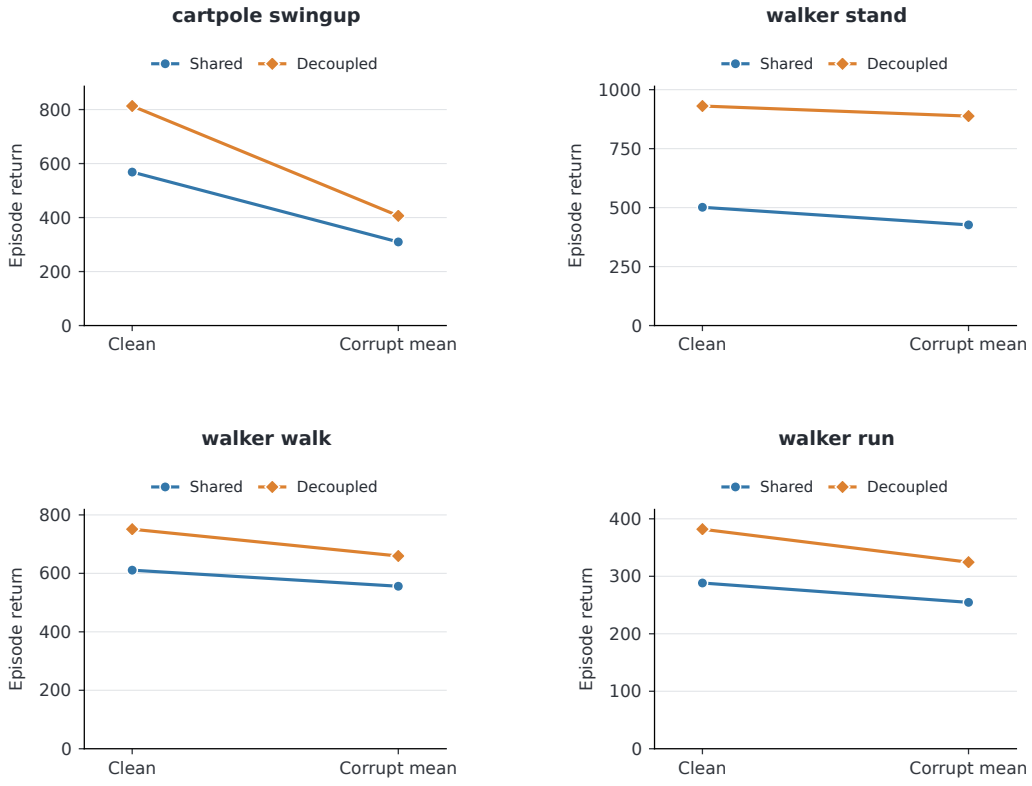

Figure 9: **Clean and corrupted episode returns in the four reported DMC tasks.** Each panel compares Shared PPO and Decoupled PPO in the task's original reward units. Corrupted returns average the evaluated corruption conditions. Markers show point estimates; higher values are better, including for negative returns.

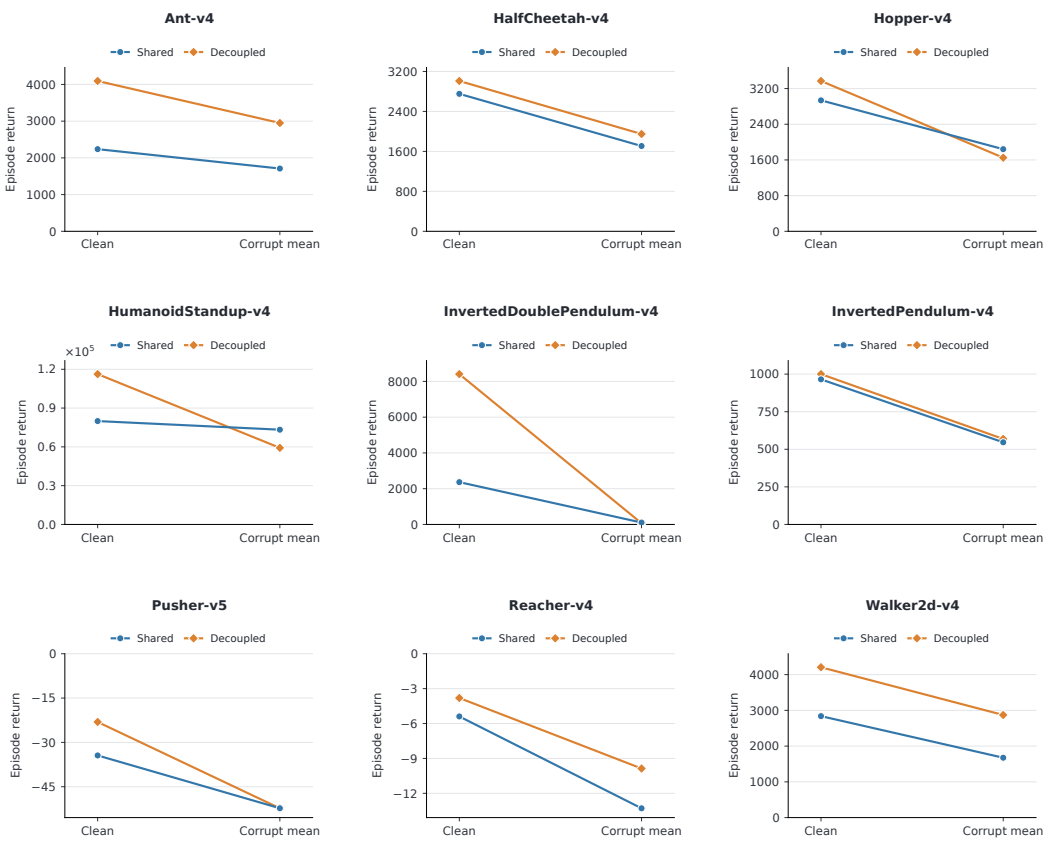

Figure 10: **Clean and corrupted episode returns in the nine reported Robust Gymnasium tasks.** Each panel compares Shared PPO and Decoupled PPO in the task's original reward units. Corrupted returns average the evaluated corruption conditions. Markers show point estimates; higher values are better, including for negative returns.

C.1.1 Performance by Corruption Type

Tables 25–27 report the full task-by-corruption comparisons in original reward units. The DMC scope is restricted to cartpole_swingup, walker_stand, walker_walk, and walker_run. Figures 11 and 12 provide a complementary comparison of proportional return retention by corruption type. The ratios are calculated within each task after seed averaging and then averaged with equal task weights; they do not pool returns across different reward scales.

Table 25: **Procgen episode returns by corruption type.** Each task has Shared (S) and Decoupled (D) columns in its original reward units. Clean and corrupted entries are task-level point estimates after averaging training seeds; higher values are better. Corruption types are not pooled in these rows.

	BigFish		CoinRun		Heist		Jumper		StarPilot	
Condition	S	D	S	D	S	D	S	D	S	D
Clean	2.31	20.57	6.87	8.93	1.00	1.57	4.57	5.40	25.40	30.07
Gaussian noise	2.32	20.36	7.83	8.97	1.30	2.30	5.60	5.77	22.98	30.47
Impulse noise	1.99	6.06	7.73	8.10	1.93	3.37	6.50	6.30	11.17	13.72
Gaussian blur	1.99	12.91	5.50	6.43	0.17	0.40	2.37	3.17	1.24	0.67
Defocus blur	1.87	11.51	5.40	6.17	0.23	0.33	1.90	2.97	0.68	0.35
Motion blur	1.59	6.23	4.57	6.10	0.20	0.30	1.67	2.47	1.08	0.69
Brightness	2.09	17.18	6.83	8.93	1.07	1.50	4.57	5.23	22.69	21.50
Contrast	2.02	13.01	4.60	8.47	0.37	1.07	4.43	5.30	18.59	18.21
Low light	2.20	15.45	6.97	8.80	0.57	1.30	4.63	5.43	16.07	7.43
Fog	2.33	16.39	5.03	8.17	0.87	1.30	4.37	5.17	19.52	20.90
Rain	2.37	20.36	6.80	8.87	0.90	1.70	4.90	5.47	25.29	28.00
Snow	1.93	22.24	6.33	8.13	0.90	1.50	4.87	5.30	24.80	28.76
Translation	2.09	15.54	4.47	7.37	0.73	1.03	3.50	4.57	16.88	20.08
Cutout	1.92	14.58	6.57	8.67	0.80	1.27	4.70	5.37	23.75	28.62
JPEG	1.77	10.83	6.37	8.03	0.27	0.30	1.50	2.03	3.65	2.84
Composite	1.43	3.30	6.53	7.17	0.17	0.33	1.33	2.00	0.40	0.43
Markov switching	2.69	16.32	7.43	7.60	2.00	3.03	6.27	6.10	9.65	8.20

Table 26: **DMC episode returns by corruption type.** Each task has Shared (S) and Decoupled (D) columns in its original reward units. Clean and corrupted entries are task-level point estimates after averaging training seeds; higher values are better. Corruption types are not pooled in these rows. Only the four specified DMC tasks are included.

	Cartpole swingup		Walker stand		Walker walk		Walker run	
Condition	S	D	S	D	S	D	S	D
Clean	568.46	813.12	501.40	930.68	610.94	751.06	288.35	381.92
Gaussian noise	558.40	702.41	499.32	937.67	620.84	751.59	294.14	377.02
Impulse noise	439.20	417.80	466.28	937.51	618.39	751.28	290.53	373.22
Gaussian blur	245.92	524.51	463.58	929.42	589.24	745.45	283.23	374.52
Defocus blur	249.81	367.17	452.38	922.29	575.39	742.88	283.39	371.85
Motion blur	216.28	353.28	441.88	903.83	577.51	600.42	236.42	334.17
Brightness	453.61	698.77	458.02	912.37	583.29	720.17	286.72	356.40
Contrast	237.49	410.21	382.02	891.63	571.73	668.00	230.29	341.11
Low light	359.55	388.35	396.37	847.26	576.20	596.48	281.26	253.02
Fog	403.73	574.52	431.76	894.78	448.25	749.16	217.59	375.41
Rain	340.89	342.52	476.77	935.42	617.34	743.17	280.00	373.93
Snow	186.26	214.49	448.09	942.78	587.66	734.58	262.30	366.06
Translation	155.12	177.73	306.94	624.36	151.94	112.19	81.24	72.33
Cutout	294.92	374.10	514.52	940.17	618.98	738.61	292.78	369.76
JPEG	292.59	460.24	432.63	927.58	632.78	714.48	283.12	363.75
Composite	240.64	91.38	217.49	743.28	551.30	477.40	210.60	153.30
Markov switching	279.87	406.80	440.26	916.66	574.32	704.52	260.80	336.57

Table 27: **Robust Gymnasium episode returns under each evaluated corruption.** Each condition has Shared (S) and Decoupled (D) columns. Returns retain their original signed reward units; higher values are better. All nine reported tasks are included, including Pusher and Reacher. The two shifts are constant offsets; the configuration originally named uniform is the fixed $+0.4$ shift.

	Clean		Gaussian		Shift $+0.005$		Shift $+0.4$	
Task	S	D	S	D	S	D	S	D
Ant-v4	2237.70	4094.26	2265.29	4199.75	2299.28	4191.75	564.58	459.03
HalfCheetah-v4	2751.81	3009.37	2730.66	2927.21	2733.71	2967.11	-342.53	-51.31
Hopper-v4	2935.70	3370.02	2345.70	2845.77	3169.95	2082.85	8.01	23.39
HumanoidStandup-v4	79937.53	116256.75	55029.84	68089.12	93580.70	50256.40	71234.18	59370.13
InvertedDoublePendulum-v4	2366.04	8407.34	151.34	81.31	111.32	65.40	54.15	56.43
InvertedPendulum-v4	965.20	1000.00	865.58	699.93	763.14	1000.00	10.43	6.34
Pusher-v5	-34.41	-23.09	-46.89	-43.43	-52.55	-53.94	-57.41	-59.53
Reacher-v4	-5.39	-3.80	-13.30	-8.29	-12.78	-10.34	-13.78	-10.96
Walker2d-v4	2838.26	4205.87	2402.09	4003.90	2470.59	4333.63	152.38	274.76

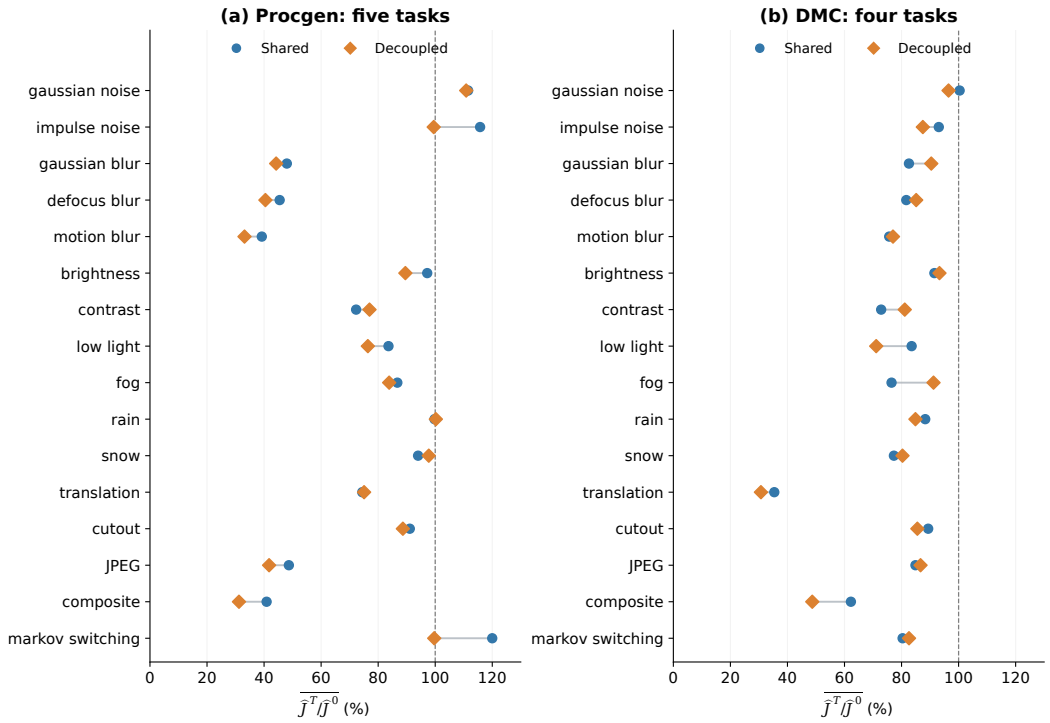

Figure 11: **Shared and decoupled return retention by visual corruption type.** Each pair compares $100\,\overline{\widehat{J}^T/\widehat{J}^0}$ for Shared and Decoupled PPO. Higher values indicate greater proportional retention of clean performance; 100% denotes unchanged return. Procgen uses five tasks, and DMC uses four tasks. Markers are point estimates, and connecting segments link architectures under the same corruption.

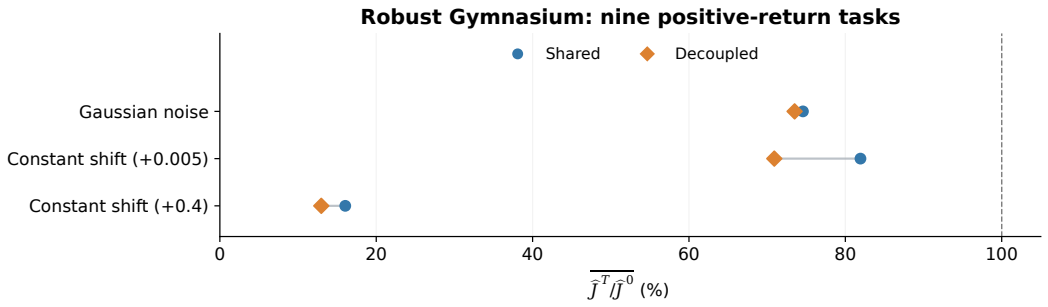

Figure 12: **Shared and decoupled return retention under vector-observation corruptions.** The figure averages within-task retention ratios over tasks with positive clean return. Higher values indicate greater retention; markers are descriptive point estimates.

C.2 Information Specialization and Erosion (RQ2)

Table 28 supplies the clean and corrupted information estimates underlying Figure 4. The relative-loss and corruption-specific analyses below retain their distinct aggregation rules.

Table 28: **Clean and corrupted representation information and specialization.** MI estimates and specialization scores are in nats, shown as clean → corrupted macro averages. Raw MI magnitudes are compared within benchmarks. The corresponding erosion and relative-loss comparisons appear in Figure 4.

Quantity	Procgen	DMC	Robust Gymnasium
$I_A(Z_S)$	$0.36856 \to 0.32374$	$0.78633 \to 0.69439$	$1.99500 \to 1.47638$
$I_G(Z_S)$	$0.21447 \to 0.20360$	$0.18916 \to 0.16521$	$0.79510 \to 0.59503$
$I_A(Z_A)$	$0.38766 \to 0.33669$	$0.75441 \to 0.67267$	$2.01909 \to 1.49422$
$I_A(Z_C)$	$0.32031 \to 0.28260$	$0.71614 \to 0.63479$	$1.88627 \to 1.38933$
$I_G(Z_A)$	$0.20428 \to 0.20559$	$0.14996 \to 0.13428$	$0.75665 \to 0.58761$
$I_G(Z_C)$	$0.25989 \to 0.22157$	$0.19092 \to 0.16085$	$1.08172 \to 0.74025$
$\overline{S}_A^0 \to \overline{S}_A^T$	$0.06735 \to 0.05410$	$0.03827 \to 0.03788$	$0.13282 \to 0.10489$
$\overline{S}_G^0 \to \overline{S}_G^T$	$0.05561 \to 0.01598$	$0.04096 \to 0.02657$	$0.32507 \to 0.15264$

C.2.1 Relative Losses across All Information Channels

Table 29 complements the return-information comparison in Figure 4 with the remaining Actor–Critic channels. It reports relative channel losses, distinct from both the absolute MI shifts in Figure 6 and erosion of the specialization gaps. In Procgen, the Actor–return channel has a negative mean relative loss, while the Critic–return channel loses information on average, complementing the return-specialization contraction reported in the main text.

Table 29: **Mean relative MI loss for all four Actor–Critic information channels.** Each entry is $100\,\overline{1 - I_Y(Z_R^T)/I_Y(Z_R^0)}$ after seed aggregation within each task and corruption condition, followed by equal weighting of tasks and conditions. Shared Actor and Critic features coincide, so each shared information channel is shown once; the decoupled architecture has four distinct channels. Lower values indicate greater proportional retention; negative values indicate an average relative increase in estimated MI. DMC includes only cartpole_swingup, walker_stand, walker_walk, and walker_run.

Architecture	Channel	Procgen (%)	DMC (%)	Robust Gym. (%)
Shared	$I_A(Z_S)$	21.03	15.09	36.17
	$I_G(Z_S)$	4.64	7.24	21.85
Decoupled	$I_A(Z_A)$	19.27	14.24	37.01
	$I_A(Z_C)$	14.46	15.06	38.01
	$I_G(Z_A)$	-11.35	1.45	21.48
	$I_G(Z_C)$	15.04	11.24	26.64

C.2.2 Corruption-Specific Specialization

Figures 13 and 14 expand the benchmark averages in Figure 4 into condition-specific responses. Table 30 reports the corresponding specialization scores and erosion percentages. The nearly preserved aggregate action gap in DMC combines expansions and contractions across corruption types. For example, translation erodes S_A by 176.21%, whereas Markov switching expands it, giving an erosion of -208.13%. The DMC ordering between action and return erosion therefore varies across corruptions. In Robust Gymnasium, return-specialization erosion exceeds action-specialization erosion under each of the three evaluated configurations.

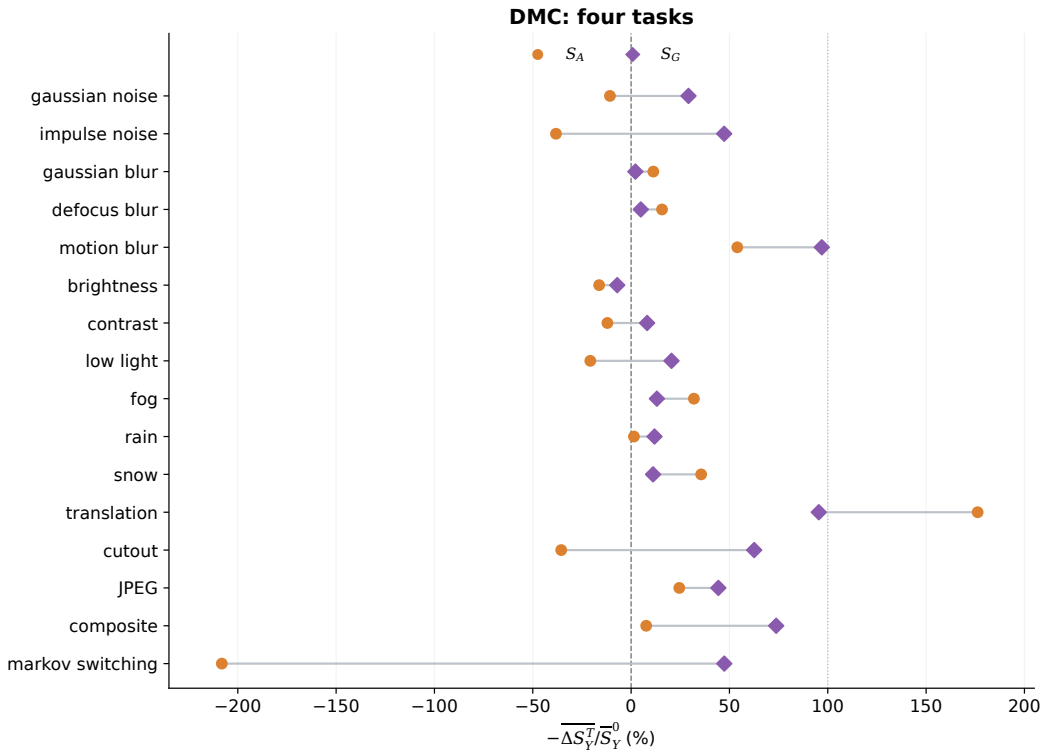

Figure 13: **Action and return specialization erosion by corruption type in DMC.** Each point is $-100\,\overline{\Delta S_Y^T}/\overline{S}_Y^0$, with task means restricted to cartpole_swingup, walker_stand, walker_walk, and walker_run. Larger values indicate stronger erosion; negative values indicate expansion. The zero line marks an unchanged gap, and 100% marks elimination of the aggregate clean gap.

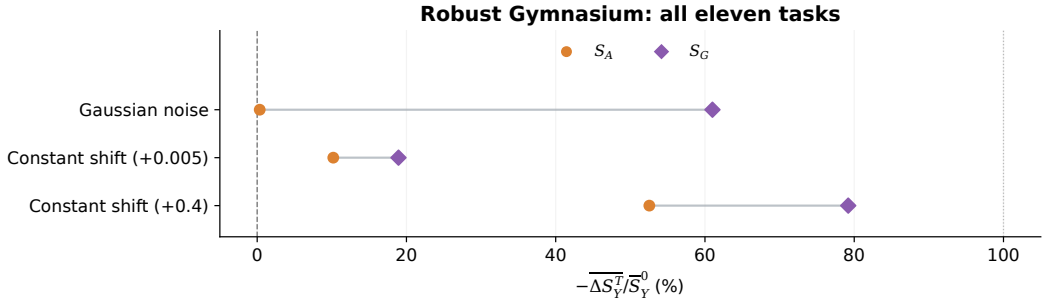

Figure 14: **Action and return specialization erosion under vector-observation corruptions.** Erosion is computed from the macro-average specialization gaps. All three configurations show greater proportional erosion of S_G than S_A.

Table 30: **Corruption-specific specialization in Decoupled PPO.** Specialization scores are task macro averages in nats; erosion is $100(\overline{S}_Y^0 - \overline{S}_Y^T)/\overline{S}_Y^0$. Negative erosion indicates expansion, and erosion above 100% indicates reversal of the aggregate clean gap. DMC uses the specified four tasks; Robust Gymnasium uses all nine tasks. Clean rows give the positive macro-average denominators.

Condition	$\overline{S}_A$	$\overline{S}_G$	S_A erosion (%)	S_G erosion (%)
DMC (four tasks)				
Clean	0.03827	0.04096	—	—
Gaussian noise	0.04239	0.02901	-10.76	29.18
Impulse noise	0.05290	0.02158	-38.20	47.31
Gaussian blur	0.03395	0.04006	11.30	2.18
Defocus blur	0.03226	0.03894	15.71	4.92
Motion blur	0.01761	0.00124	53.99	96.96
Brightness	0.04450	0.04385	-16.25	-7.06
Contrast	0.04289	0.03762	-12.06	8.14
Low light	0.04622	0.03255	-20.75	20.54
Fog	0.02606	0.03559	31.92	13.11
Rain	0.03773	0.03608	1.43	11.90
Snow	0.02464	0.03638	35.62	11.17
Translation	-0.02917	0.00187	176.21	95.44
Cutout	0.05188	0.01533	-35.54	62.57
JPEG	0.02890	0.02278	24.49	44.37
Composite	0.03534	0.01075	7.67	73.76
Markov switching	0.11793	0.02156	-208.13	47.37
Robust Gymnasium (nine tasks)				
Clean	0.13282	0.32507	—	—
Gaussian noise	0.13237	0.12677	0.34	61.00
Shift $+0.005$	0.11928	0.26352	10.20	18.94
Shift $+0.4$	0.06303	0.06763	52.55	79.20

C.3 Representation-to-Output Measurements (RQ3)

Paired geometry measurements. For the original final-head inputs U_R, $R \in \{A, C\}$, the Procgen feature analysis computes

$$D_{\mathrm{rel}}(U_R) = \left(\frac{\mathbb{E}_{\mathrm{pair}} \|U_R^T - U_R^0\|_2^2}{\mathbb{E}_{\text{clean-train}} \|U_R^0 - \mathbb{E}_{\text{clean-train}} U_R^0\|_2^2} \right)^{1/2}. \tag{89}$$

The denominator is the total variance of the corresponding clean training features. This statistic measures each network's paired displacement relative to its own scale; it does not align representation coordinates between architectures. For $B = PW_A$, let Π_B be the orthogonal projector onto its row space. The policy-head row-space energy fraction is the mean of $\|\Pi_B \Delta U_A^T\|_2^2 / \|\Delta U_A^T\|_2^2$ over nonzero displacements. It measures direction allocation without the singular-value weighting in $\|B\Delta U_A^T\|_2$. The geometry correlations in Table 2 use the matched outputs from that paired analysis. They are computed across corruption conditions within each checkpoint and summarized by their median, rather than by pooling individual transitions.

C.4 Output–Return Associations and Corruption Traces (RQ4)

Table 31 reports association coefficients, complementing the response magnitudes in Table 2. Table 32 gives numerical values for four illustrative conditions in Figure 6.

Table 31: **Local output changes are associated with return degradation.** Entries are Spearman correlations. Procgen uses 16 corruption-category summaries and proportional return drop $1 - \widehat{J}^T/\widehat{J}^0$. Robust Gymnasium computes correlations with $\widehat{D}^T_{\mathrm{ret}}$ over the nine seed–condition pairs within each task and reports the median across tasks. The aggregation procedures differ, so the coefficients are interpreted within their respective analyses. These are descriptive associations rather than causal effects.

Benchmark	Local measurement	Shared PPO	Decoupled PPO
Procgen	Policy divergence $\mathcal{K}^T$	0.697	0.809
	Greedy-action flips $\mathcal{F}^T$	0.715	0.818
	Value drift $\mathcal{D}^T_V$	0.653	0.800
Robust Gymnasium	Policy divergence $\mathcal{K}^T$	0.517	0.483
	Mean shift $\mathbb{E}\|\Delta\mu^T\|_2$	0.567	0.467
	Executed-action distance $\mathbb{E}\|a^T - a^0\|_2$	0.567	0.533
	Value drift $\mathcal{D}^T_V$	0.559	0.467

Table 32: **Different corruptions produce distinct responses across analysis levels.** All entries are percentages. Observation and return retention average within-task ratios. Erosion of S_G uses $-\overline{\Delta S^T_G}/\overline{S}^0_G$. The return column concerns Decoupled PPO. Observation MI refers to the projected input rather than the complete observation. Erosion above 100% indicates reversal of the macro-average specialization gap.

Corruption	Retention of $I(X; G_{\mathrm{ref}})$	Erosion of S_G	Return retention $\widehat{J}^T_D/\widehat{J}^0_D$
Defocus blur	100.31	137.12	40.47
Gaussian blur	100.36	129.49	44.24
Translation	60.76	47.80	75.07
Cutout	82.19	1.05	88.66

C.5 Return Performance beyond PPO

Table 33 supplements the Procgen comparison in Table 3 with the four DMC tasks. Figure 15 shows the corresponding clean training curves. For completeness, Figure 16 shows the clean PPO training curves in the nine reported vector tasks.

Table 33: **Clean and corrupted episode returns on four visual DMC tasks.** PPG separates policy and auxiliary value learning, but scores below Decoupled PPO on every task under both conditions. DrAC regularizes policy and value predictions across image augmentations. It improves mean corrupted return on three tasks with Shared PPO and two with Decoupled PPO, while reducing it on walker_walk for both. With DrAC, Decoupled PPO outperforms Shared PPO on all four tasks under both conditions. Corrupted returns average 16 corruption types per task. PPG uses a decoupled architecture. Higher is better. The Shared and Decoupled PPO baselines are in Table 24.

Task	PPG	Shared PPO + DrAC	Decoupled PPO + DrAC
Clean			
cartpole_swingup	650.63	635.86	856.09
walker_stand	840.82	782.92	920.74
walker_walk	730.57	473.77	657.16
walker_run	363.34	325.47	341.65
Corrupted			
cartpole_swingup	255.04	333.64	498.62
walker_stand	813.74	605.07	911.20
walker_walk	564.76	382.74	623.56
walker_run	300.74	284.89	324.50

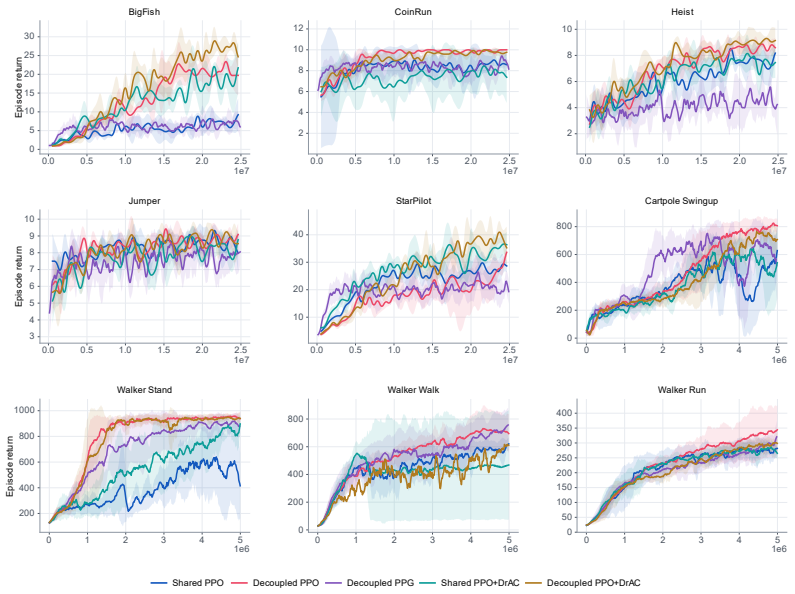

Figure 15: Clean returns on five Procgen and four DMC environments. The curves compare shared PPO, decoupled PPO, decoupled PPG, shared PPO+DrAC, and decoupled PPO+DrAC. Solid lines show the multi-seed mean; shaded bands show one sample standard deviation across available seeds. Curves end at the last available training step.

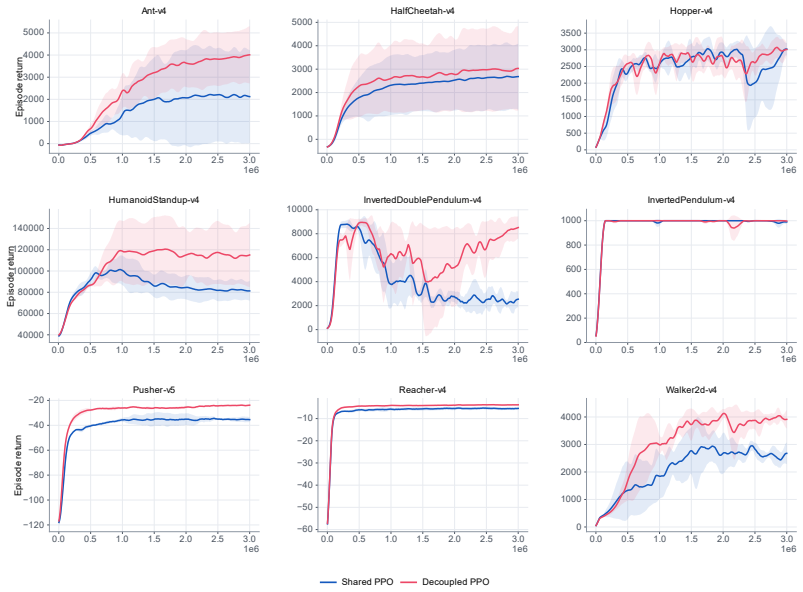

Figure 16: Clean returns on nine Robust Gymnasium environments. Blue and red curves show shared PPO and decoupled PPO, respectively. Shaded bands show one sample standard deviation across seeds; Return scales are specific to each environment.