Disentangling Spurious Correlations in Vision-Language-Action Models via Predicting Domain-Invariant Latent Lookahead

Junghyun Kim[1,2*], Ngseo Kim[1*], ChungWoo Lee[1], Seoyeon Lee[1], Woo-Jeong Baek[5], Adam Zhou[2], Chip Huyen[2], Jun-Ki Lee[1†], Gi-Cheon Kang[3†], Byoung-Tak Zhang[1,4†]

[1]Seoul National University, [2]OpenMind, [3]Ajou University
[4]Tommoro Robotics, [5]Hyundai Motors

***Abstract*—Vision-Language-Action (VLA) models remain brittle under visual distribution shifts, often relying on spurious correlations tied to domain-specific factors rather than task-relevant structure. We propose *Domain-Invariant Latent Lookahead* (DILL), a representation-learning framework that mitigates shortcut learning in VLA policies. Our key idea is to supervise policies with domain-invariant future latents learned from domain-transformed trajectory data. A Task-Domain Encoder is trained with contrastive objectives and Gaussian disentanglement regularization to separate task-relevant structure from domain-specific visual variation. The learned encoder then provides future latents for VLA policy learning through lookahead prediction and domain disentanglement, encouraging the policy to focus on task-relevant structure rather than incidental visual factors. Under counterfactual task–view compositions, DILL reduces shortcut degree while improving OOD success. On LIBERO-Plus visual perturbations, DILL achieves the highest average success rate of 69.1%, outperforming the strongest matched-input baseline by 11.4 absolute points, while limiting the average performance drop to 12.5 points. Latent-space diagnostics further show that DILL better preserves task-consistent structure while suppressing domain-specific variation.**

***Index Terms*—Vision-Language-Action Models, Shortcut Learning, Generalist Robot Policies, Predictive Future Latents, Domain Generalization**

I. Introduction

A longstanding goal of robot learning is to build robots that generalize across a wide range of tasks and environments. Recent advances in Vision-Language-Action (VLA) models [8, 26, 21, 7] trained on large-scale robot datasets [12, 20] have established a promising path toward generalist robot policies by grounding actions in language and visual observations. Despite this progress, VLA models often remain brittle under distribution shift, particularly under incidental visual variation such as changes in viewpoint, background, lighting, or camera configuration [14, 39, 36], which should be irrelevant to task success. This brittleness undermines reliable out-of-distribution generalization and remains a major obstacle to practical deployment.

Such failures can be naturally understood through the lens of *shortcut learning* [15, 38], where policies rely on task-irrelevant visual cues that correlate with successful actions in the training distribution rather than the task-relevant semantics required for robust control. VLA models are particularly susceptible to shortcut learning for two reasons. First, robot datasets often exhibit limited within-dataset diversity and are fragmented across collection sources, causing the same task or action to repeatedly co-occur with particular visual cues [38]. Second, VLA models often inherit representations from pretrained vision-language backbones [30, 3] that are optimized for visual understanding or image-language alignment rather than control, and may therefore preserve domain-specific visual information irrelevant to action generation. When mapped to actions, these representations can allow spurious visual correlations in training data to be absorbed into the policy's decision rule [13], producing the task-substitution failure illustrated in Fig. 1: under counterfactual task–domain recomposition, the policy may execute the task associated with the observed domain.

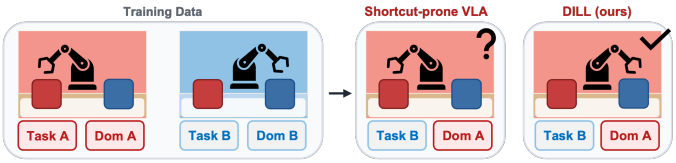


Fig. 1. **Shortcut learning under task–domain confounding.** When tasks and visual domains are spuriously correlated, a VLA may use domain-specific appearance as a shortcut for action selection. DILL reduces this failure by conditioning actions on a domain-invariant latent lookahead.

To mitigate such shortcuts, a VLA policy should learn representations that preserve task-relevant structure while discarding incidental visual variability. A natural learning signal for this purpose is future state prediction [43, 33]. Predicting how a task will evolve can encourage models to capture factors that determine future task states, such as object configuration, task progress, and action-conditioned scene changes. In this sense, future prediction provides a form of supervision that is more closely tied to control than to static visual appearance. However, future prediction alone is not sufficient. If the target is a raw future observation or an entangled visual latent, the model may still preserve domain-specific factors such as viewpoint or background that help predict future appearance but do not determine the correct action. Thus, the key question is not simply whether a VLA policy should predict the future,

but *what representation of the future it should predict*. We argue that the predictive target should be *domain-invariant*: it should retain the future task structure needed for control while suppressing domain-specific factors that can act as shortcuts.

We propose *Domain-Invariant Latent Lookahead* (DILL), a predictive representation learning framework for robust VLA control. DILL first learns a Task-Domain Encoder from domain-transformed trajectory chunks using contrastive objectives over two pair types: *task-positive pairs* that preserve trajectory chunk across domain changes, and *domain-positive pairs* that share the same domain condition across different chunks. A Gaussian disentanglement loss further separates task and domain latents: task latents remain stable across changes in viewpoint, environment appearance, camera configuration, and visual degradation, while domain latents capture such domain-specific factors. During policy learning, the pretrained Task-Domain Encoder provides future latent supervision: the VLA policy predicts a lookahead latent aligned with the future task latent, while its current representation is regularized to be disentangled from the corresponding domain latent. The predicted lookahead latent and current representation jointly condition the action head, encouraging the policy to base its actions on task-relevant future structure rather than incidental visual factors.

The central effect of DILL is to reshape the information that the policy uses to choose actions: instead of allowing domain-specific visual cues to enter the action head as reliable proxies for the task, DILL conditions actions on a predicted future latent that is stable across domain changes and informative about the commanded behavior. This makes the learned action representation less tied to where or how a scene is observed, and more tied to the information needed to produce the correct action. Empirically, we evaluate this effect through complementary behavioral and representational analyses. Under counterfactual task–view compositions, DILL substantially reduces shortcut reliance and improves OOD task success, showing that policies are less likely to substitute the commanded task with the task spuriously associated with the observed view. On LIBERO-Plus visual perturbations, DILL achieves the highest average success rate among matched-input baselines and exhibits a substantially smaller performance drop under camera, lighting, background, and sensor-noise shifts. Latent-space diagnostics further show that the final action-conditioning representation is organized more by task-consistent trajectory content than by shared visual appearance. Together, these results show that shortcut learning in VLA policies can be mitigated not only by increasing data diversity, but by explicitly shaping the predictive representation used for control.

II. Related Work

A. Spurious Correlations in Robot Learning

Spurious correlations [38] have emerged as an important challenge in robot learning, where policies may rely on incidental visual factors, such as camera viewpoint, background, and lighting, that correlate with successful actions in the training data [14, 39, 44, 36]. Prior work has addressed this issue from two perspectives: *data-centric* and *representation-centric* approaches. Data-centric approaches increase nuisance diversity through novel-view synthesis [35] or synthetic viewpoint augmentation [38]. Within representation-centric approaches, one line of work suppresses task-irrelevant factors by building invariance into visual representations, for example, by reducing sensitivity to viewpoint changes [24, 31, 27]. Another line of work injects task-relevant structure, such as spatial information [40]. Our work bridges these directions by using *predictive future latents* as task-relevant supervision while explicitly suppressing task-irrelevant visual factors through domain-invariant latent disentanglement.

B. World Models and Future Prediction in VLA Models

Recent VLA work has incorporated *future prediction* or *world modeling* signals to complement direct perception-to-action learning by anticipating the consequences of actions [43, 18, 42]. One line performs pixel-level imagination by generating future frames or subgoal observations before acting, as in the *GR* series, Ctrl-World, CoT-VLA, and SuSIE [37, 11, 16, 42, 6]. These approaches provide interpretable visual foresight, but they can be costly and prone to compounding errors. The second line moves future prediction into latent space by predicting or aligning future latent representations, including Video Prediction Policy, FLARE, VLA-JEPA, and FRAPPE [18, 43, 33, 41, 1, 2]. These methods avoid explicit pixel rollouts while encouraging representations that capture future structure useful for control. In contrast, DILL focuses on what the policy is asked to predict: rather than aligning to raw or entangled future latents, it learns future targets whose task-relevant structure is separated from domain-specific visual variation.

III. Method

A. Overview

Domain-Invariant Latent Lookahead (DILL) trains VLA policies with domain-invariant predictive supervision through a two-stage procedure. As shown in Fig. 2, we first learn a Task-Domain Encoder using domain-transformed trajectory data. The task encoder produces task latents that are encouraged to remain invariant across domain changes, while the domain encoder produces domain latents that capture domain-specific visual variation. Both latents are learned using contrastive objectives defined by task-positive and domain-positive pairs, together with a Gaussian disentanglement loss that encourages the task and domain latents to be separated.

We then use the learned Task-Domain Encoder to supervise VLA policy learning. For each training trajectory, the encoder provides future task and domain latents as supervisory signals. From the current observation and language instruction, the policy predicts a lookahead latent and a current representation: the former is aligned with the future task latent, while the latter is regularized to be disentangled from the domain latent. These two policy latents are fed to the action head for control, encouraging the policy to combine a predicted future task

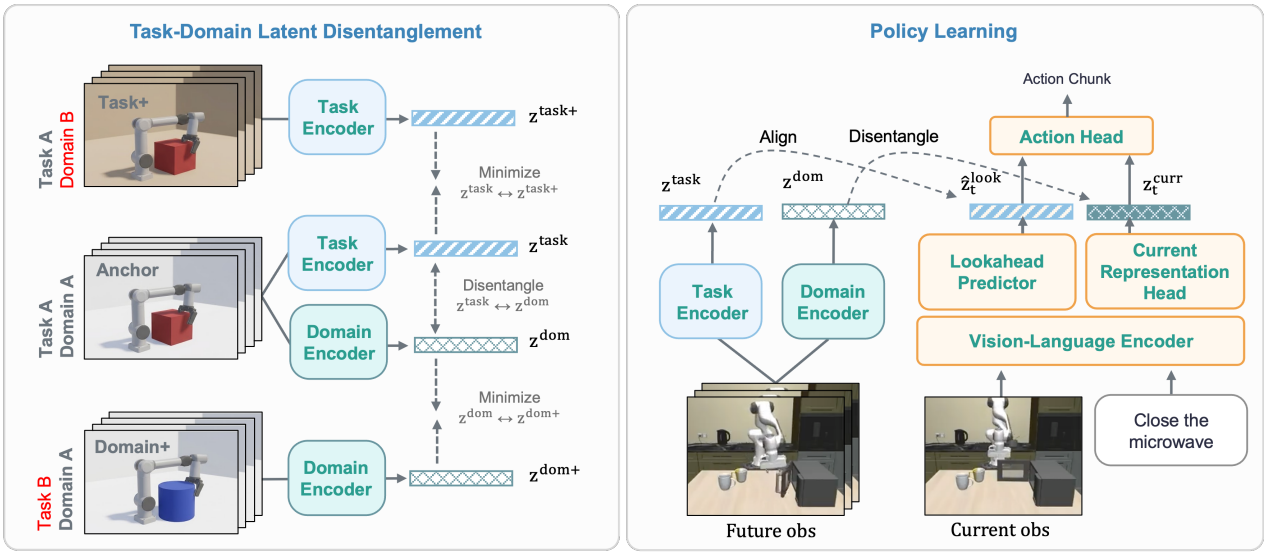


Fig. 2. **Overview of Domain-Invariant Latent Lookahead.** We first learn a Task-Domain Encoder that maps observation chunks into task and domain latents. During policy learning, a Lookahead Predictor predicts a future task latent from the current observation and instruction, while a Current Representation Head produces a domain-disentangled current representation. The two policy latents condition the action head for control.

representation with a current representation discouraged from carrying domain-specific visual information. As a result, the policy is trained to rely on task-relevant future structure rather than shortcut-inducing visual cues. At test time, the policy requires only the current observation and instruction.

B. Task-Domain Latent Disentanglement

The Task-Domain Encoder learns task and domain latents from video observations: task latents are encouraged to preserve task-relevant structure, while domain latents capture domain-specific visual factors. Let $\mathbf{o}_t^i = (o_t^i, \ldots, o_{t+T_v-1}^i)$ denote an observation chunk from trajectory i, and let $\mathcal{T}_\eta$ denote a domain transformation with condition η, where η specifies the full domain condition. We construct task-positive and domain-positive pairs:

$$\underbrace{(\mathcal{T}_{\eta_1}(\mathbf{o}_t^i), \mathcal{T}_{\eta_2}(\mathbf{o}_t^i))}_{\text{task-positive}}, \quad \eta_1 \neq \eta_2, \qquad \underbrace{(\mathcal{T}_\eta(\mathbf{o}_t^i), \mathcal{T}_\eta(\mathbf{o}_{t'}^j))}_{\text{domain-positive}}, \quad i \neq j. \tag{1}$$

Task-positive pairs preserve the same observation chunk while changing the domain condition, whereas domain-positive pairs share the same domain condition across different observation chunks. The transformation families include viewpoint changes, environment appearance variation, camera heterogeneity, and visual degradation. See Appendix A-A for details.

Let $x = \mathcal{T}_\eta(\mathbf{o})$ denote a transformed observation chunk. The Task-Domain Encoder consists of two separate video encoders: a task encoder and a domain encoder. Each encoder maps the input observation chunk to a pooled latent vector:

$$z^{\text{task}} = E_\psi^{\text{task}}(x), \qquad z^{\text{dom}} = E_\xi^{\text{dom}}(x). \tag{2}$$

Implementation details are provided in Appendix A-D.

Both encoders are trained with the same InfoNCE form but with different positive pair constructions. For a minibatch of positive pairs $\{(x_i, x_i^+)\}_{i=1}^B$, let z_i and z_i^+ denote the pooled latent vectors from the corresponding encoder:

$$\mathcal{L}_{\text{NCE}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp(\text{sim}(z_i, z_i^+)/\tau)}{\sum_{k=1}^{B} \exp(\text{sim}(z_i, z_k^+)/\tau)} \tag{3}$$

where $\text{sim}(\cdot,\cdot)$ denotes cosine similarity and τ is a temperature. Applying Eq. 3 to the task encoder with task-positive pairs and to the domain encoder with domain-positive pairs gives $\mathcal{L}_{\text{task}}$ and $\mathcal{L}_{\text{dom}}$, respectively.

To further disentangle the task and domain latents, we introduce a Gaussian disentanglement loss by applying SIGReg [4] to their concatenation. For each minibatch, we form a joint task-domain latent:

$$u_i^{\text{td}} = [z_i^{\text{task}}; z_i^{\text{dom}}], \qquad \mathcal{L}_{\text{dis}}^{\text{td}} = \mathcal{R}_{\text{dis}}\left(\{u_i^{\text{td}}\}_{i=1}^B\right), \tag{4}$$

where $[\cdot;\cdot]$ denotes concatenation. Unlike applying distributional regularization to each latent separately, this joint regularizer acts on the concatenated task-domain latent. Under a Gaussian approximation, matching the joint latent to an isotropic Gaussian discourages cross-covariance between the task and domain latents, thereby encouraging approximate disentanglement. Appendix A-B provides the corresponding argument.

The task-domain disentanglement objective is

$$\mathcal{L}_{\text{TDD}} = \lambda_{\text{task}}\mathcal{L}_{\text{task}} + \lambda_{\text{dom}}\mathcal{L}_{\text{dom}} + \lambda_{\text{dis}}^{\text{td}}\mathcal{L}_{\text{dis}}^{\text{td}}. \tag{5}$$

Task-domain encoder pretraining. We pretrain the task and domain encoders on large-scale domain-transformed trajectory chunks from ManiSkill [34], MimicGen [25], and a subset of OXE [12]. We refer to these data as the source trajectory collection. The pretrained encoders define the task and domain latent spaces used later to provide future supervision for VLA policy learning.

C. VLA Policy Learning with Domain-Invariant Latent Lookahead

During VLA policy training, the pretrained Task-Domain Encoder provides future latent supervision. Given the current observation o_t and language instruction ℓ, the vision-language backbone [3] produces image-language tokens:

$$H_t^{\text{vlm}} = M_\theta(o_t, \ell). \tag{6}$$

Two policy heads map these tokens to a predicted lookahead latent and a current representation:

$$\hat{z}_t^{\text{look}} = P_\theta^{\text{look}}(H_t^{\text{vlm}}), \qquad z_t^{\text{curr}} = P_\theta^{\text{curr}}(H_t^{\text{vlm}}). \tag{7}$$

Let $\mathbf{o}_t^+ = (o_{t+1}, \ldots, o_{t+T_v})$ denote the future observation chunk. Using the pretrained Task-Domain Encoder, we extract a future task target and its domain latent:

$$z_t^{\text{task},\star} = E_{\bar{\psi}}^{\text{task}}\left(\mathbf{o}_t^+\right), \qquad z_t^{\text{dom},\star} = E_{\bar{\xi}}^{\text{dom}}\left(\mathbf{o}_t^+\right). \tag{8}$$

The lookahead latent is aligned with the future task target using the InfoNCE objective in Eq. 3. For each minibatch, we instantiate Eq. 3 by using the predicted lookahead latents as anchors, $z_i = \hat{z}_i^{\text{look}}$, and the stop-gradient future task latents as positives, $z_i^+ = \text{sg}(z_i^{\text{task},\star})$. We denote this loss by $\mathcal{L}_{\text{look}}$.

We also disentangle the current representation from the domain latent extracted by the domain encoder:

$$u_i^{\text{curr}} = \left[z_i^{\text{curr}}; \text{sg}\left(z_i^{\text{dom},\star}\right)\right], \qquad \mathcal{L}_{\text{dis}}^{\text{curr}} = \mathcal{R}_{\text{dis}}\left(\{u_i^{\text{curr}}\}_{i=1}^B\right). \tag{9}$$

Since the current and future chunks come from the same trajectory and domain condition, $z_t^{\text{dom},\star}$ provides the corresponding domain latent. The Gaussian disentanglement loss encourages z_t^{curr} to be separated from the domain latent while preserving information useful for action prediction.

The lookahead latent provides predictive task information about the future observation chunk, while the current representation provides action-relevant information from the present input after domain disentanglement:

$$\hat{a}_{t:t+K_a-1} = A_\phi\left([\hat{z}_t^{\text{look}}; z_t^{\text{curr}}]\right). \tag{10}$$

We supervise the action chunk with behavior cloning:

$$\mathcal{L}_{\text{act}} = \frac{1}{K_a}\sum_{k=0}^{K_a-1} \|\hat{a}_{t+k} - a_{t+k}\|_1. \tag{11}$$

The final policy objective is

$$\mathcal{L}_{\text{policy}} = \lambda_{\text{act}}\mathcal{L}_{\text{act}} + \lambda_{\text{look}}\mathcal{L}_{\text{look}} + \lambda_{\text{dis}}^{\text{curr}}\mathcal{L}_{\text{dis}}^{\text{curr}}. \tag{12}$$

Policy pretraining and adaptation. We pretrain the policy-side modules on action-labeled trajectories from the source trajectory collection. For downstream settings such as LIBERO [23] or real-world robot data, we fine-tune the policy with the same lookahead, disentanglement, and action losses. At deployment, the policy receives only the current observation and instruction.

IV. Experiments

We evaluate the central claim of this paper: domain-invariant latent lookahead mitigates shortcut learning in vision-language-action policies. The main experiments focus on controlled simulation settings, where task–domain correlations and visual distribution shifts can be systematically manipulated. Specifically, we ask:

1. Does our approach reduce shortcut reliance under counterfactual task–view compositions?

2. Does our approach improve robustness under visual distribution shifts?

3. Do the learned latents successfully separate task structure from task-irrelevant domain factors?

A. LIBERO Shortcut Diagnostic Under Counterfactual Task–View Compositions

We first test shortcut mitigation in a controlled LIBERO diagnostic [38]. This diagnostic intentionally creates a spurious correlation between task identity and camera viewpoint during training, then breaks this correlation at test time to measure whether a policy follows the commanded task or the task spuriously associated with the observed view.

Protocol. The training split consists of two confounded task–view islands. A left-view task group, denoted as Task-L, is observed only from a left-view range, while a right-view task group, denoted as Task-R, is observed only from a right-view range. During evaluation, we counterfactually swap these compositions: Task-R is evaluated under the left-view boundary, and Task-L is evaluated under the right-view boundary. Additional details are provided in Appendix B-A.

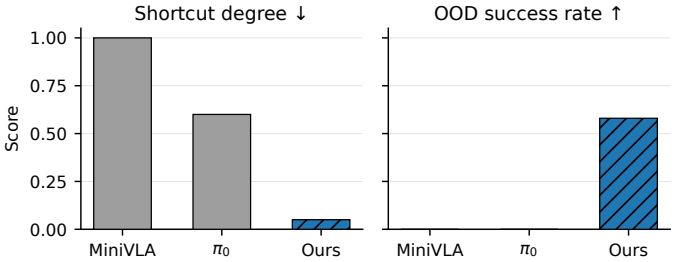


Fig. 3. **LIBERO shortcut diagnostic.** Lower shortcut degree and higher OOD success indicate less domain-induced task substitution.

Metrics. We report OOD success and shortcut degree. OOD success measures whether the commanded task is completed under the counterfactual view. Shortcut degree measures whether the policy instead executes the task group spuriously associated with the observed view during training. Lower shortcut degree indicates less shortcut reliance.

Results. Figure 3 compares our method against MiniVLA [5] and π_0 [7], using the baseline results reported in the low-diversity setting of [38]. MiniVLA and π_0 exhibit shortcut-prone behavior, with shortcut degrees of 1.00 and 0.60, respectively, and both obtain zero OOD success. In contrast, our method reduces shortcut degree to 0.05 while improving OOD success to 0.58. Importantly, shortcut degree

TABLE I

Zero-shot robustness evaluation on visual perturbations in LIBERO-Plus [14]. For each model, the first row shows success rate (%) on four visual perturbation categories in LIBERO-Plus. The second row shows the absolute drop relative to the original score.

Model	Original	Visual perturbations				Average
		Camera	Light	BG	Noise	
OpenVLA [21]	76.5	0.8 ↓75.7	8.1 ↓68.4	34.8 ↓41.7	15.2 ↓61.3	14.7 ↓61.8
WorldVLA [10]	79.1	0.1 ↓79.0	43.7 ↓35.4	17.1 ↓62.0	10.9 ↓68.2	18.0 ↓61.2
UniVLA [9]	95.5	1.8 ↓93.7	69.0 ↓26.5	81.0 ↓14.5	21.2 ↓74.3	43.3 ↓52.3
NORA [19]	87.9	2.2 ↓85.7	45.7 ↓42.2	58.6 ↓29.3	12.8 ↓75.1	29.8 ↓58.1
OpenVLA-OFT [22]	95.3	10.4 ↓84.9	76.8 ↓18.5	93.6 ↓1.7	49.9 ↓45.4	57.7 ↓37.6
DILL (Ours)	81.6	68.4 ↓13.2	69.4 ↓12.2	70.0 ↓11.6	68.7 ↓12.9	69.1 ↓12.5

measures view-induced task substitution rather than generic task failure. Thus, this result shows that our method does not merely improve execution under distribution shift; it specifically reduces the tendency to replace the commanded task with the task associated with the observed view. This directly supports our central hypothesis that domain-invariant latent lookahead discourages reliance on spurious task–view correlations.

B. LIBERO-Plus Zero-Shot Robustness Evaluation Under Visual Distribution Shifts

We next evaluate whether shortcut mitigation translates into stronger robustness under visual distribution shifts. We use the visual perturbation categories in LIBERO-Plus [14] as a controlled zero-shot evaluation: no LIBERO-Plus perturbed images are used for training. All models are adapted on the original LIBERO training split.

Compared methods. To ensure a controlled comparison, the main table includes only methods evaluated with third-person RGB observations, excluding wrist-camera images. Under this matched-input protocol, we compare against OpenVLA [21], OpenVLA-OFT [22], WorldVLA [10], UniVLA [9], and NORA [19].

Results. Table I reports the LIBERO-Plus results on visual perturbations. Our method achieves the highest average success across the four visual shift categories, improving over the strongest external baseline, OpenVLA-OFT, by 11.4 points. This gain is not explained by stronger in-distribution performance: OpenVLA-OFT and UniVLA obtain higher original LIBERO scores than our method, but suffer substantially larger drops under visual perturbations. In contrast, our method has an average drop of 12.5 points, compared to 37.6 for OpenVLA-OFT and 52.3 for UniVLA.

The improvement is especially pronounced for camera and sensor-noise perturbations, where policies must avoid relying on brittle view- or image-quality-specific cues. Our method achieves 68.4% success under camera perturbations and 68.7% under sensor noise, while all external baselines remain below 11% and 50% on these two categories, respectively. Although OpenVLA-OFT remains stronger on background and lighting perturbations, our method is substantially more balanced across visual shifts, suggesting that domain-invariant latent lookahead reduces sensitivity to spurious visual factors tied to the training distribution. We report the full LIBERO-Plus breakdown in Appendix B-C.

C. Pairwise Diagnostics of Learned Latents

Finally, we analyze whether the learned latent spaces separate task-relevant structure from domain-specific visual variation. We evaluate three representations: the task encoder output, the domain encoder output, and the final VLA latent provided to the action head. For the final VLA latent, we use the concatenated action-conditioning representation, *i.e.*, the current policy representation together with the predicted lookahead latent.

Protocol. We compute cosine similarity between normalized latents for two types of unseen task and domain pairs. *Task pairs* share the same underlying trajectory content but differ in visual domain or augmentation. *Domain pairs* share the same visual domain or augmentation pattern but differ in trajectory content. These pair combinations are not observed during training, so the diagnostic tests whether the learned representations generalize beyond memorized pairings. A task-centric representation should assign higher similarity to task pairs than to domain pairs, while a domain-centric representation should show the opposite behavior. Additional details are provided in Appendix C.

Results. Figure 4 shows that the learned factorization behaves as intended. The task encoder assigns high similarity to task pairs and substantially lower similarity to domain pairs, with mean similarities of 0.88 and 0.35, respectively. Conversely, the domain encoder assigns high similarity to domain pairs and lower similarity to task pairs, with mean similarities of 0.94 and 0.34, indicating that the task and domain encoders capture complementary factors rather than collapsing to the same representation. The final VLA latent also remains strongly task-centric: it assigns high similarity to task pairs ($\mu = 0.94$) while keeping domain pairs noticeably lower ($\mu = 0.55$). Since this is the representation directly

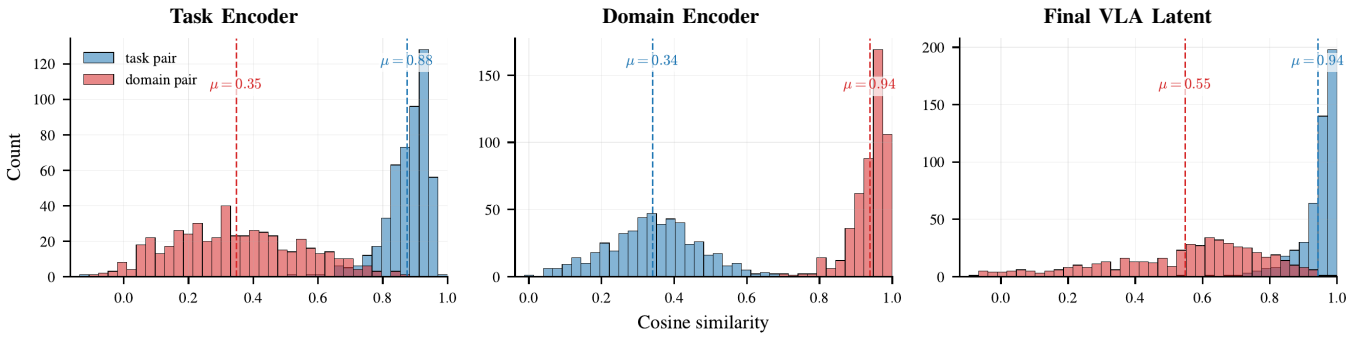


Fig. 4. **Pairwise similarity diagnostics for learned latents.** We compare cosine-similarity distributions for task pairs and domain pairs constructed from unseen task–domain combinations. From left to right, the panels show the task encoder, domain encoder, and final VLA latent. The task encoder should group task pairs, the domain encoder should group domain pairs, and the final VLA latent should preserve task-consistent structure while suppressing domain-specific variation.

provided to the action head, the result suggests that the policy is conditioned on latent features that are stable across domain changes but still discriminative across different trajectory content. This supports the mechanism behind the robustness gains in Table I: domain-invariant latent lookahead encourages the policy to act on task-consistent structure rather than domain-specific visual cues.

TABLE II
Final action-conditioning latent comparison. The gap is task-pair similarity minus domain-pair similarity.

Representation	Task ↑	Domain ↓	Gap ↑
Base VLA final latent	0.52	0.70	-0.18
DILL final latent	**0.94**	**0.55**	**0.39**

Comparison to Base VLA. Table II further compares DILL against an architecture-matched Base VLA. Base VLA uses the same VLA architecture, observation and language inputs, and action head as our method, but removes domain-invariant latent lookahead and its associated representation objectives; it predicts actions directly from the current observation-conditioned representation using behavior cloning. We therefore compare the actual representation routed into the action head: the standard Base VLA latent versus DILL's concatenation of the current representation and predicted lookahead latent. In this comparison, task pairs share trajectory content under different visual domains, while domain pairs share visual domain but differ in trajectory content. Base VLA exhibits the undesirable ordering for shortcut-robust control: its domain-pair similarity (0.70) is higher than its task-pair similarity (0.52), yielding a negative task-centric gap of -0.18. This suggests that the standard VLA latent is more strongly organized by shared visual domain than by shared trajectory content. DILL reverses this ordering, increasing the gap to 0.39, which indicates that the action-conditioning latent becomes stable across domain changes while remaining discriminative across different trajectory content.

V. Limitations

DILL is designed to reduce shortcut reliance induced by visual-domain factors. In our experiments, these factors include changes in viewpoint, scene appearance, camera configuration, and visual degradation. While this captures a common source of brittleness in VLA policies, it does not cover all possible spurious correlations. For example, biases in initial states, object layouts, language templates, task frequencies, or demonstrator styles may also influence action prediction without being part of the intended task semantics. Addressing such non-visual shortcuts would require defining additional nuisance factors, obtaining corresponding paired interventions, or developing objectives that can discover them more automatically.

Another important consideration is the trade-off between robustness and in-distribution performance. In the training distribution, domain-dependent cues can be highly predictive of demonstrated actions, even when they are not causally tied to the intended task semantics. Moreover, the same visual factor may play different roles across contexts: a background pattern may be a spurious cue in one setting, but part of the task-relevant scene configuration in another. Overly strong invariance may therefore suppress information that is useful for control in some situations, reducing peak in-distribution performance even as it improves robustness when those cues become misleading. This suggests that robustness should not be pursued by uniformly removing domain-specific information in all cases. Instead, future work should explore adaptive objectives that preserve action-relevant factors and suppress action-irrelevant shortcuts in a context-dependent manner.

VI. Conclusion

A central obstacle to robust VLA control is that policies can treat domain-specific appearance cues as action-relevant when they are only reliable within the training distribution. Domain-Invariant Latent Lookahead addresses this shortcut learning problem by supervising VLA representations with future latents in a space that separates domain-specific visual variation from task-relevant structure. This encourages the policy to rely less on incidental appearance factors such as background, viewpoint, or lighting, and more on action-relevant scene information stable across domains. Our results suggest that VLA robustness depends not only on scale, but also on shaping what the policy learns.

References

[1] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15619–15629, 2023.

[2] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

[3] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

[4] Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics, 2025. URL https://arxiv.org/abs/2511.08544.

[5] Suneel Belkhale and Dorsa Sadigh. Minivla: A better vla with a smaller footprint, 2024. URL https://github.com/Stanford-ILIAD/openvla-mini.

[6] Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pre-trained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.

[7] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[8] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[9] Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. UniVLA: Learning to act anywhere with task-centric latent actions, 2025. URL https://arxiv.org/abs/2505.06111.

[10] Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. WorldVLA: Towards autoregressive action world model, 2025. URL https://arxiv.org/abs/2506.21539.

[11] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

[12] Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, et al. Open X-Embodiment: Robotic learning datasets and RT-X models. https://arxiv.org/abs/2310.08864, 2023.

[13] Pim De Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. *Advances in neural information processing systems*, 32, 2019.

[14] Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

[15] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2 (11):665–673, 2020.

[16] Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. *arXiv preprint arXiv:2510.10125*, 2025.

[17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

[18] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

[19] Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U-Xuan Tan, Navonil Majumder, and Soujanya Poria. NORA: A small open-sourced generalist vision language action model for embodied tasks, 2025. URL https://arxiv.org/abs/2504.19854.

[20] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[21] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[22] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

[23] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking

knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

[24] Fanfan Liu, Feng Yan, Liming Zheng, Chengjian Feng, Yiyang Huang, and Lin Ma. Robouniview: Visual-language model with unified view representation for robotic manipulation. *arXiv preprint arXiv:2406.18977*, 2024.

[25] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. In *7th Annual Conference on Robot Learning*, 2023.

[26] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.

[27] Jing-Cheng Pang, Nan Tang, Kaiyuan Li, Yuting Tang, Xin-Qiang Cai, Zhen-Yu Zhang, Gang Niu, Masashi Sugiyama, and Yang Yu. Learning view-invariant world models for visual robotic manipulation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[28] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=di52zR8xgf.

[29] Poly Haven. Poly Haven: A free 3D asset library. https://polyhaven.com/, 2024. Textures licensed under CC0 1.0 Universal Public Domain Dedication.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[31] Younggyo Seo, Junsu Kim, Stephen James, Kimin Lee, Jinwoo Shin, and Pieter Abbeel. Multi-view masked world models for visual robotic manipulation. In *International Conference on Machine Learning*, pages 30613–30632. PMLR, 2023.

[32] Stability AI. Stable Diffusion XL Base 1.0. https://huggingface.co/stabilityai/stable-diffusion-xl-base-1.0, 2023. Hugging Face model card.

[33] Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. Vla-jepa: Enhancing vision-language-action model with latent world model. *arXiv preprint arXiv:2602.10098*, 2026.

[34] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *Robotics: Science and Systems*, 2025.

[35] Stephen Tian, Blake Wulfe, Kyle Sargent, Katherine Liu, Sergey Zakharov, Vitor Guizilini, and Jiajun Wu. View-invariant policy learning via zero-shot novel view synthesis. *arXiv preprint arXiv:2409.03685*, 2024.

[36] Guodong Wang, Chenkai Zhang, Qingjie Liu, Jinjin Zhang, Jiancheng Cai, Junjie Liu, and Xinmin Liu. Libero-x: Robustness litmus for vision-language-action models. *arXiv preprint arXiv:2602.06556*, 2026.

[37] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023.

[38] Youguang Xing, Xu Luo, Junlin Xie, Lianli Gao, Hengtao Shen, and Jingkuan Song. Shortcut learning in generalist robot policies: The role of dataset diversity and fragmentation. *arXiv preprint arXiv:2508.06426*, 2025.

[39] Borong Zhang, Jiahao Li, Jiachen Shen, Yishuai Cai, Yuhao Zhang, Yuanpei Chen, Juntao Dai, Jiaming Ji, and Yaodong Yang. Vla-arena: An open-source framework for benchmarking vision-language-action models. *arXiv preprint arXiv:2512.22539*, 2025.

[40] Ji Zhang, Shihan Wu, Xu Luo, Hao Wu, Lianli Gao, Heng Tao Shen, and Jingkuan Song. Inspire: Vision-language-action models with intrinsic spatial reasoning. *arXiv preprint arXiv:2505.13888*, 2025.

[41] Han Zhao, Jingbo Wang, Wenxuan Song, Shuai Chen, Yang Liu, Yan Wang, Haoang Li, and Donglin Wang. Frappe: Infusing world modeling into generalist policies via multiple future representation alignment. *arXiv preprint arXiv:2602.17259*, 2026.

[42] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1702–1713, 2025.

[43] Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loic Magne, et al. Flare: Robot learning with implicit world modeling. *arXiv preprint arXiv:2505.15659*, 2025.

[44] Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

Appendix Overview

This appendix provides additional details, analyses, and supplementary experiments for DILL. Appendix A describes the method implementation and training objectives. Appendix B provides simulation benchmark details, including the LIBERO shortcut diagnostic, ablations, and the full LIBERO-Plus breakdown. Appendix C provides additional representation diagnostics.

Contents.

- Appendix A: Method details and training objectives.
- Appendix B: Simulation benchmark experiments.
- Appendix C: Representation diagnostics.

Appendix A
Additional Method Details

A. Domain Transformations and Positive Pair Mining

This subsection supports Sec. III-B by detailing how we construct the domain-transformed positive pairs to train the Task-Domain Encoder. A domain condition η specifies the full transformation condition, including the transformation family and its sampled parameters or random seed. Task-positive pairs vary the domain condition while preserving the same underlying observation chunk, whereas domain-positive pairs preserve the domain condition across different observation chunks.

a) Data sources.: Task-domain encoder pretraining uses a source trajectory collection composed of MimicGen [25], ManiSkill [34], and subsets of OXE [12]. The OXE subset includes Bridge and Fractal/RT-1 trajectories. Table III summarizes the pretraining data. In our main experiments, the Task-Domain Encoder is pretrained once on this source trajectory collection and then kept fixed during VLA policy pretraining and downstream policy learning. This design decouples Task-Domain Encoder pretraining from downstream policy learning: downstream users can train only the policy-side modules unless they choose to further adapt the Task-Domain Encoder with additional domain transformations.

TABLE III
Source trajectory collection used for Task-Domain Encoder pretraining. The MimicGen row aggregates 16 task datasets, while OXE aggregates Bridge and Fractal/RT-1 subsets.

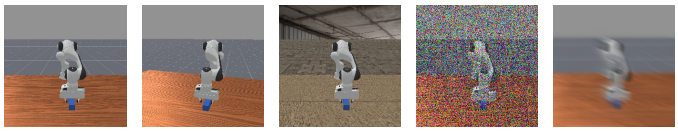
Source	Frames	Trajectories	Tasks
MimicGen, 16 datasets	1,010,618	3,200	16
ManiSkill, 2 datasets	1,632,291	14,057	17
OXE, Bridge + Fractal/RT-1	5,785,810	140,404	20,553
Total	8,428,719	157,661	20,586

For simulation data, controllable rendering enables viewpoint changes and, when segmentation masks are available, mask-based background replacement. We additionally apply image-level transformations directly to frames; these transformations are used for both simulation data and recorded trajectories such as the OXE subsets.

b) Transformation families.: We group domain transformations into four families that reflect common visual domain shifts: viewpoint changes, environment appearance variation, camera heterogeneity, and visual degradation. A sampled domain condition can include a single transformation or a combination of transformations from these families. Table IV summarizes the transformation types and the shared domain condition used for domain-positive pairs. Figure 5 illustrates simple single-family task-positive examples; the training procedure samples a broader range of transformation types, parameters, and combinations.

TABLE IV
Domain transformation families used for positive pair mining. A domain condition η consists of a transformation family and its sampled parameters or random seed. The last column lists the condition shared by domain-positive pairs.

Family	Transformations	Shared condition for domain positives
Viewpoint changes	Rendered camera view, random crop, intrinsics change, radial distortion, warping	Camera ID or camera parameters; crop seed; focal scale; distortion and warp parameters
Environment appearance	Background replacement, lighting changes, color changes	Background texture IDs; mask regions; replacement mode; lighting or color parameters
Camera heterogeneity	Camera unprocessing, sensor noise, camera color response changes	Camera-pipeline seed; color matrices; gamma; ISO level; shot/read noise; channel gains
Visual degradation	Blur, weather effects, compression, pixelation, image corruptions	Corruption type, severity, and random seed



Anchor | Viewpoint Changes | Environment Appearance | Camera Heterogeneity | Visual Degradation

Fig. 5. Representative single-family task-positive transformations. Each column after the anchor shows the same trajectory chunk under one family of domain-specific visual factors.

c) Background replacement.: For ManiSkill trajectories [34] with available segmentation masks, we perform mask-based background replacement for table, floor, and wall regions. Table and floor textures are sampled from Poly Haven [29], while wall and background assets are generated with Stable Diffusion XL using text-to-image prompts for robot-relevant environments such as factories, kitchens, laboratories, workspaces, loading docks, and server rooms [28, 32]. The resulting texture pool contains 92 table textures, 163 floor textures, and 604 wall/background textures, for a total of 859 region-specific assets. At training time, textures are sampled by region and composited into the corresponding segmentation

masks. For domain-positive pairs, the same texture IDs, mask regions, and replacement mode define the shared domain condition. Representative samples from the texture pool are shown in Fig. 6.

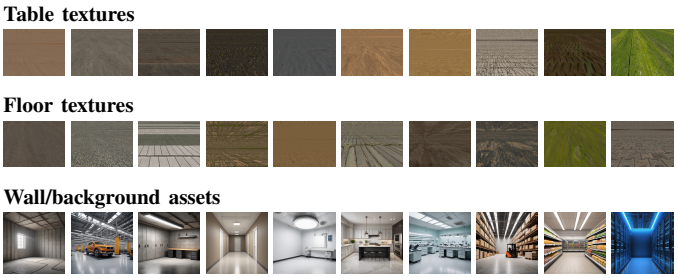


Fig. 6. Representative assets for mask-based background replacement. We sample table and floor textures from Poly Haven and generate wall and background assets with Stable Diffusion XL.

d) Pair mining details.: For the task encoder, each task-positive pair is constructed from the same episode and the same temporal window. Each training item samples two non-overlapping windows from the same episode, denoted by $\mathbf{o}^i_{t_a}$ and $\mathbf{o}^i_{t_b}$, and samples two domain conditions η_1 and η_2. We then form two task-positive pairs:

$$\big(\mathcal{T}_{\eta_1}(\mathbf{o}^i_{t_a}), \mathcal{T}_{\eta_2}(\mathbf{o}^i_{t_a})\big), \qquad \big(\mathcal{T}_{\eta_1}(\mathbf{o}^i_{t_b}), \mathcal{T}_{\eta_2}(\mathbf{o}^i_{t_b})\big). \tag{13}$$

Within each pair, the two clips share the same underlying temporal window but differ in domain condition. Across the two pairs, the temporal windows are distinct while the task identity and domain conditions are shared. After collation, the two task-positive pairs per item are flattened from $[B, 2, \cdots]$ to $[2B, \cdots]$ before applying InfoNCE. Thus, examples from the same episode but different temporal windows are included as ordinary in-batch negatives. This discourages the task encoder from solving the contrastive objective using task identity alone, while avoiding false negatives from the same underlying temporal segment.

For the domain encoder, a domain-positive pair is constructed by applying the same domain condition to two different clips. Each training item first samples a source pair $(\mathbf{o}_a, \mathbf{o}_b)$, either from two non-overlapping windows of the same episode or from cross-task clips within the same camera bucket. It then samples four distinct domain-transformation types without replacement, with corresponding domain conditions $\eta_1, \ldots, \eta_4$, and forms four domain-positive pairs:

$$\big(\mathcal{T}_{\eta_m}(\mathbf{o}_a), \mathcal{T}_{\eta_m}(\mathbf{o}_b)\big), \qquad m = 1, \ldots, 4. \tag{14}$$

Within each pair, the clips differ in trajectory content but share the same domain condition. Across the four pairs in the same item, the source clips are fixed while the domain-transformation type changes. After collation, the four domain-positive pairs per item are flattened from $[B, 4, \cdots]$ to $[4B, \cdots]$ before applying InfoNCE. Therefore, the domain encoder receives controlled in-batch comparisons: for an anchor under domain condition η_m, other clips from the same item but with $\eta_{m'} \neq \eta_m$ appear among the standard in-batch negatives. No separate hard-negative loss is used; the dataloader simply ensures that informative non-matching domain conditions are present in the minibatch.

All transformations are applied consistently across the frames of a chunk so that each video chunk corresponds to a coherent visual domain.

B. Task-Domain Latent Disentanglement Objectives

a) InfoNCE objective.: For a minibatch of positive pairs $\{(x_i, x_i^+)\}_{i=1}^B$, let z_i and z_i^+ denote the pooled latent vectors from the corresponding encoder. We use the InfoNCE loss

$$\mathcal{L}_{\text{NCE}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp(\text{sim}(z_i, z_i^+)/\tau)}{\sum_{k=1}^{B} \exp(\text{sim}(z_i, z_k^+)/\tau)}, \tag{15}$$

where $\text{sim}(\cdot,\cdot)$ denotes cosine similarity and τ is a temperature. For the task encoder, the positive pairs are task-positive pairs; for the domain encoder, the positive pairs are domain-positive pairs. The resulting losses are $\mathcal{L}_{\text{task}}$ and $\mathcal{L}_{\text{dom}}$.

b) Gaussian disentanglement loss.: The task and domain encoders are trained with different positive pairs, but without an additional constraint, the resulting task and domain latents can still encode overlapping information. We therefore define a Gaussian disentanglement loss that regularizes concatenated latents toward an isotropic Gaussian distribution.

For a minibatch, we construct

$$u_i^{\text{td}} = [z_i^{\text{task}}; z_i^{\text{dom}}], \qquad i = 1, \ldots, B, \tag{16}$$

where $[\cdot;\cdot]$ denotes concatenation. Here $z_i^{\text{task}}, z_i^{\text{dom}} \in \mathbb{R}^{D_z}$, and therefore $u_i^{\text{td}} \in \mathbb{R}^D$ with $D = 2D_z$. We implement the disentanglement regularizer $\mathcal{R}_{\text{dis}}$ using SIGReg [4]. Given a batch of vectors $U = \{u_i\}_{i=1}^B \subset \mathbb{R}^D$, we sample M random unit directions $r_m \sim \text{Unif}(\mathbb{S}^{D-1})$, project $s_{i,m} = r_m^\top u_i$, and match each projected distribution to a standard Gaussian. Using a Gaussian kernel with bandwidth σ, the Epps–Pulley statistic for the projected samples $\{s_{i,m}\}_{i=1}^B$ along direction r_m is

$$\begin{aligned}\text{EP}_\sigma(\{s_{i,m}\}_{i=1}^B) = \frac{1}{B^2}\sum_{i=1}^{B}\sum_{j=1}^{B} \exp\left(-\frac{(s_{i,m}-s_{j,m})^2}{2\sigma^2}\right) \\ - \frac{2\sigma}{B\sqrt{\sigma^2+1}}\sum_{i=1}^{B}\exp\left(-\frac{s_{i,m}^2}{2(\sigma^2+1)}\right) + \frac{\sigma}{\sqrt{\sigma^2+2}}.\end{aligned} \tag{17}$$

The regularizer averages this statistic over random projection directions:

$$\mathcal{R}_{\text{dis}}(U) = \frac{1}{M}\sum_{m=1}^{M} \text{EP}_\sigma\left(\{r_m^\top u_i\}_{i=1}^B\right). \tag{18}$$

The task-domain disentanglement loss is

$$\mathcal{L}_{\text{dis}}^{\text{td}} = \mathcal{R}_{\text{dis}}\left(\{u_i^{\text{td}}\}_{i=1}^B\right). \tag{19}$$

c) Why Gaussian matching encourages disentanglement.: The Gaussian disentanglement loss does not guarantee exact independence for arbitrary distributions. Its motivation is clearest under a joint Gaussian approximation. Let

$$u = \begin{bmatrix} z^a \\ z^b \end{bmatrix}$$

be a jointly Gaussian random vector with covariance

$$\Sigma = \begin{bmatrix} \Sigma_a & C \\ C^\top & \Sigma_b \end{bmatrix}. \tag{20}$$

If the concatenated latent is isotropic Gaussian, then $\Sigma = I$, which implies

$$\Sigma_a = I, \qquad \Sigma_b = I, \qquad C = 0. \tag{21}$$

For jointly Gaussian variables, zero cross-covariance is sufficient for independence, so

$$p(z^a, z^b) = p(z^a)p(z^b). \tag{22}$$

Thus, driving the concatenated latent $[z^a, z^b]$ toward an isotropic Gaussian encourages the two components to be disentangled.

We use this argument twice. For task-domain latent disentanglement, $z^a = z^{\text{task}}$ and $z^b = z^{\text{dom}}$. For policy learning, $z^a = z^{\text{curr}}$ and $z^b = z^{\text{dom},\star}$. In practice, this should be interpreted as an approximate disentanglement regularizer rather than a guarantee of exact independence.

d) Total disentanglement objective.: The full task-domain disentanglement objective is

$$\mathcal{L}_{\text{TDD}} = \lambda_{\text{task}}\mathcal{L}_{\text{task}} + \lambda_{\text{dom}}\mathcal{L}_{\text{dom}} + \lambda_{\text{dis}}^{\text{td}}\mathcal{L}_{\text{dis}}^{\text{td}}. \tag{23}$$

C. Policy Objectives

a) Lookahead contrastive alignment.: For a minibatch, the Lookahead Predictor outputs $\{\hat{z}_i^{\text{look}}\}_{i=1}^B$, and the pretrained task encoder provides future task targets $\{z_i^{\text{task},\star}\}_{i=1}^B$. We train the Lookahead Predictor with the in-batch InfoNCE loss

$$\mathcal{L}_{\text{look}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp\left(\text{sim}\left(\hat{z}_i^{\text{look}}, \text{sg}(z_i^{\text{task},\star})\right)/\tau_{\text{look}}\right)}{\sum_{k=1}^{B}\exp\left(\text{sim}\left(\hat{z}_i^{\text{look}}, \text{sg}(z_k^{\text{task},\star})\right)/\tau_{\text{look}}\right)}. \tag{24}$$

The positive pair is the predicted lookahead and its corresponding future task target; for each anchor, all other future task targets in the batch serve as negatives. The stop gradient $\text{sg}(\cdot)$ prevents this loss from updating the pretrained task encoder.

b) Current Representation Head domain disentanglement.: For a minibatch, let $\{z_i^{\text{curr}}\}_{i=1}^B$ denote current representations produced by the Current Representation Head, and let $\{z_i^{\text{dom},\star}\}_{i=1}^B$ denote the corresponding domain latents extracted by the pretrained domain encoder. We construct

$$u_i^{\text{curr}} = [z_i^{\text{curr}}; \text{sg}(z_i^{\text{dom},\star})], \qquad i = 1, \ldots, B. \tag{25}$$

The current-domain disentanglement loss is

$$\mathcal{L}_{\text{dis}}^{\text{curr}} = \mathcal{R}_{\text{dis}}\left(\{u_i^{\text{curr}}\}_{i=1}^B\right). \tag{26}$$

Under the Gaussian approximation described above, this encourages the current representation to be disentangled from the domain component while retaining information needed for action prediction.

c) Action loss and policy objective.: The action head receives both policy latents:

$$\hat{a}_{t:t+K_a-1} = A_\phi\left([\hat{z}_t^{\text{look}}; z_t^{\text{curr}}]\right). \tag{27}$$

The behavior cloning loss is

$$\mathcal{L}_{\text{act}} = \frac{1}{K_a}\sum_{k=0}^{K_a-1} \|\hat{a}_{t+k} - a_{t+k}\|_1. \tag{28}$$

The full policy objective is

$$\mathcal{L}_{\text{policy}} = \lambda_{\text{act}}\mathcal{L}_{\text{act}} + \lambda_{\text{look}}\mathcal{L}_{\text{look}} + \lambda_{\text{dis}}^{\text{curr}}\mathcal{L}_{\text{dis}}^{\text{curr}}. \tag{29}$$

D. Architecture and Implementation Details

a) Task and domain encoders.: In the main text, E_ψ^{task} and E_ξ^{dom} denote the full task and domain encoder branches that output pooled latent vectors. Each branch consists of a VJEPA2 video backbone [2], branch-specific LoRA adapters [17], and an attention-pooling projection head.

b) Multi-query attention pooling.: Let $H \in \mathbb{R}^{N\times D_v}$ be a sequence of input tokens and let $Q \in \mathbb{R}^{R\times D_q}$ be R learned query vectors. We project queries, keys, and values using $W_q \in \mathbb{R}^{D_q\times d}$, $W_k \in \mathbb{R}^{D_v\times d}$, and $W_v \in \mathbb{R}^{D_v\times D_p}$. A multi-query attention pooler computes

$$A = \text{softmax}\left(\frac{(QW_q)(HW_k)^\top}{\sqrt{d}}\right), \tag{30}$$

$$U = AHW_v \in \mathbb{R}^{R\times D_p}. \tag{31}$$

The pooled tokens U are flattened and further projected to the output latent dimension D_z:

$$\text{Head}(H) = W_o\,\text{vec}(U) + b_o, \qquad W_o \in \mathbb{R}^{D_z\times(RD_p)}, \quad b_o \in \mathbb{R}^{D_z}. \tag{32}$$

The projection heads for the task encoder, domain encoder, Lookahead Predictor, and Current Representation Head all use this attention-pooling-plus-linear architecture, with separate parameters.

c) Policy-side heads.: The policy uses a Qwen2.5-VL [3] backbone with LoRA adaptation. On top of the VLM tokens, we train two AttentiveLatentHead modules and one ResNetActionHead. The Lookahead Predictor maps VLM tokens to $\hat{z}_t^{\text{look}}$, and the Current Representation Head maps the same tokens to z_t^{curr}. Each latent head uses 8 learned query tokens, a two-layer attentive pooler with 16 attention heads, and a linear projection to a 4096-dimensional latent.

d) Action head.: The action head is an MLP-ResNet that maps the concatenated policy latents to an action chunk. In the dual-head setting, the action head input dimension is $4096 + 4096 = 8192$. We use hidden dimension 2048, two residual MLP blocks, and output a chunk size of $K_a = 50$.

TABLE V
POLICY-SIDE HEAD BUDGET FOR THE VLA IMPLEMENTATION.

Module	Parameters
Lookahead Predictor, AttentiveLatentHead	167.85M
Current Representation Head, AttentiveLatentHead	167.85M
ResNetActionHead, dual-head input	28.48M
Total policy-side heads	364.18M

E. Training Stage Summary

Table VI summarizes which modules are updated in each training stage. In the main experiments, the Task-Domain Encoder is pretrained once on the source trajectory collection and then kept fixed. This decouples reusable Task-Domain Encoder training from downstream VLA policy training.

TABLE VI
TRAINING STAGE SUMMARY FOR DOMAIN-INVARIANT LATENT LOOKAHEAD.

Stage	Data	Updated modules	Objective
Task-domain encoder pretraining	Source trajectory collection	Task/domain encoder LoRA adapters and projection heads	$\mathcal{L}_{\text{TDD}}$
VLA policy pretraining	Source trajectory collection	VLM LoRA, Lookahead Predictor, Current Representation Head, action head	$\mathcal{L}_{\text{policy}}$
Downstream policy tuning	Downstream action-labeled demonstrations (e.g., LIBERO)	VLM LoRA, Lookahead Predictor, Current Representation Head, action head	$\mathcal{L}_{\text{policy}}$

a) Optional Task-Domain Encoder adaptation.: Although the main experiments keep the pretrained Task-Domain Encoder fixed for downstream policy tuning, the same task-domain disentanglement objective could be used to adapt the task and domain encoders on additional downstream demonstration datasets. When controllable rendering or segmentation masks are unavailable, such optional adaptation would rely on image-level transformations such as cropping, warping, lighting and color changes, camera-pipeline perturbations, and visual corruptions.

APPENDIX B
SIMULATION BENCHMARK EXPERIMENTS

A. LIBERO Shortcut Diagnostic

We use the LIBERO shortcut diagnostic of [38] to test whether a policy follows the commanded task or instead executes the task spuriously associated with the observed view. Unlike standard robustness evaluation, this diagnostic explicitly separates shortcut-driven task substitution from general execution failure.

a) Task–view confounding.: The benchmark constructs two confounded task–view islands. We refer to the left-view task group as Task-L and the right-view task group as Task-R. Task-L contains LIBERO task IDs $\{0, 1, 3, 5, 8\}$ and is observed only in the left-view range (10°–25°) during training. Task-R contains task IDs $\{2, 4, 6, 7, 9\}$ and is observed only in the right-view range (55°–70°). At test time, we evaluate counterfactual task–view compositions by swapping these associations: Task-R is evaluated at the left viewpoint (10°), and Task-L is evaluated at the right viewpoint (70°). A task-faithful policy should follow the language instruction under these swapped views, whereas a shortcut-prone policy will execute the task spuriously associated with the observed view.

b) Metrics.: We report two complementary metrics. *OOD success rate* measures whether the commanded task is completed under the counterfactual view. *Shortcut degree* measures whether the policy instead executes the task spuriously associated with the observed view during training, thereby isolating view-induced task substitution.

B. Ablation Study

We ablate DILL to identify which design choices are responsible for shortcut mitigation. The central question is not whether a model can fit the confounded training distribution, but whether it can avoid using viewpoint as a proxy for task identity when the task–view association is broken. DILL is designed to address this by reshaping the information routed to the action head: a predicted lookahead latent aligned with the future task latent, and a current representation disentangled from the corresponding domain latent. The ablation study therefore asks whether each of these ingredients is necessary, and whether simpler alternatives—more augmented data or generic future prediction—are sufficient.

All variants are evaluated on the LIBERO shortcut diagnostic in Appendix B-A. This diagnostic separates three levels of generalization. *In-dist. SR* measures success on the original task–view training compositions. *Center OOD SR* evaluates each task group at an unseen midpoint viewpoint without swapping task–view association, measuring interpolation to an unseen visual domain. *Counter OOD SR* evaluates the counterfactual task–view swaps, where the model must follow the commanded task rather than the task associated with the observed view. Finally, *shortcut degree* measures how often the model follows the task spuriously associated with the observed view; lower is better.

Compared methods. We compare DILL with several ablative models:

(1) Base VLA. Base VLA is the plain behavior-cloning baseline. It uses the same VLA architecture as DILL, but directly maps the current observation-conditioned representation to actions. It does not use source augmentation, task–domain latent targets, the disentangled current head, or latent lookahead alignment. This is the minimal baseline for testing whether standard VLA behavior cloning is sufficient under task–view confounding.

(2) Base VLA + SA. This variant keeps the same behavior-cloning objective and VLA architecture as Base VLA, but trains with the same augmented source data used by DILL for task–domain encoder pretraining. This controls for the data condition: if this variant were sufficient, the gain of DILL could be attributed mainly to additional visual diversity. If

not, then the improvement must come from how DILL uses the augmented data to shape the latent space, rather than from the data alone.

(3) Entangled LA. This variant adds latent lookahead prediction, but uses the representation from the original video encoder [2] as the future target instead of the task–domain latent targets. Thus, the predicted future latent is not explicitly encouraged to discard domain-specific information, and may still contain viewpoint, lighting, or background cues. This variant tests whether generic future prediction is sufficient, or whether the lookahead target must be domain-invariant.

(4) DILL w/o CH. This variant removes the disentangled current head. The model still uses task-domain latent targets and latent lookahead alignment, but the current action-conditioning representation is no longer explicitly regularized to be disentangled from the corresponding domain latent. This tests whether domain-invariant lookahead alone is sufficient, or whether the current representation used by the policy also needs to be explicitly shaped to suppress domain-specific visual cues.

(5) DILL w/o LA. This variant keeps the task–domain latent targets and the disentangled current head, but removes latent lookahead alignment. In other words, the current representation is still regularized through the task–domain latent structure, but the future lookahead latent is not trained to align with the domain-invariant target. This tests whether a disentangled current representation alone can mitigate shortcut learning, or whether robust control requires explicitly aligning the predicted future latent as well.

(6) DILL. DILL is the full model, combining source augmentation, task–domain latent targets, the disentangled current head, and domain-invariant latent lookahead alignment.

Results. Table VII shows that fitting the confounded training distribution is not sufficient for shortcut-free generalization. Base VLA reaches 68% in-distribution success, but obtains 0% Counter OOD success and a high shortcut degree of 73%. Adding source augmentation improves in-distribution success to 82% and slightly reduces shortcut degree to 69%, indicating that additional visual diversity and action pretraining are helpful for fitting the nominal tasks and can mildly reduce shortcut reliance. However, Counter OOD success remains only 2%, showing that data augmentation alone does not teach the policy which visual factors should be ignored when task–view correlations are counterfactually broken.

Entangled LA provides a second partial improvement. Compared to Base VLA + SA, it improves Center OOD success from 10% to 24% and further reduces shortcut degree from 69% to 65%, suggesting that future prediction can encourage representations that are somewhat more robust to unseen viewpoints. However, it still obtains 0% Counter OOD success. This failure is informative: predicting a future latent is not sufficient if the target representation remains entangled with viewpoint, background, or other domain-specific cues. The lookahead target must be tied to task-relevant structure rather than to the same visual correlations present in the training distribution.

The lower half of the table isolates the DILL-specific components. DILL w/o CH, which keeps task–domain latent targets and latent lookahead alignment but removes the disentangled current head, reaches 44% Center OOD and 44% Counter OOD success while reducing shortcut degree to 6%. This is a large jump over Entangled LA, indicating that aligning lookahead prediction with task–domain latent targets removes much of the view-induced task substitution. DILL w/o LA, which keeps the disentangled current head but removes latent lookahead alignment, achieves the strongest Counter OOD success (62%) and the lowest shortcut degree (3%), but its in-distribution success drops to 50%. This suggests that the disentangled current head is highly effective at suppressing shortcut behavior, while latent lookahead alignment helps recover a better balance between nominal task execution and counterfactual generalization.

The full DILL model provides the best overall tradeoff. It matches the best Center OOD success (62%), remains close to the best Counter OOD success (58%), keeps shortcut degree very low (5%), and achieves the highest average success across the three success metrics (62%). The key pattern is that source augmentation and generic lookahead improve some aspects of learning, but do not solve counterfactual task–

Variant	Components				Metrics				
	SA	TD	CH	LA	In-dist. SR ↑	Center OOD SR ↑	Counter OOD SR ↑	Avg. SR ↑	Shortcut degree ↓
Base VLA					68%	22%	0%	30%	73%
Base VLA + SA	√				**82%**	10%	2%	31%	69%
Entangled LA	√			√	76%	24%	0%	33%	65%
DILL w/o CH	√	√		√	60%	44%	44%	49%	6%
DILL w/o LA	√	√	√		50%	**62%**	**62%**	58%	**3%**
DILL	√	√	√	√	66%	**62%**	58%	**62%**	5%

TABLE VII
ABLATION STUDY.

view generalization. In contrast, the DILL-family variants that use task–domain latent targets sharply reduce shortcut degree, and the full model best preserves both task execution and shortcut resistance. These results support the design principle of DILL: shortcut mitigation requires not only additional visual diversity or future prediction, but a task-structured latent pathway that routes domain-invariant predictive information into action conditioning.

C. LIBERO-Plus evaluation details

We evaluate methods on LIBERO-Plus [14], which tests robustness under controlled perturbations of the original LIBERO benchmark. In the main paper, we report the visual perturbation categories because they are most directly aligned with our claim about visual shortcut mitigation. Here, we provide the full seven-category breakdown. All models are evaluated zero-shot on LIBERO-Plus after training on the original LIBERO setting.

a) Full perturbation breakdown.: Table VIII reports success rates on all LIBERO-Plus perturbation categories. *Total* denotes the average over Camera, Robot, Language, Light, BG, Noise, and Layout. The second row for each model reports the absolute drop relative to the original LIBERO score.

b) Results.: The full breakdown clarifies the scope of DILL's robustness. DILL is strongest on the perturbations that directly change visual appearance or viewpoint, achieving the best performance on Camera (68.4%) and Noise (68.7%), and competitive performance on Light (69.4%) and BG (70.0%). This pattern matches the design of the method: domain-invariant latent lookahead is intended to suppress shortcuts tied to visual-domain cues, such as viewpoint, background texture, and image degradation. In contrast, DILL is not designed to directly solve non-visual shifts. The low Language score reflects a failure mode related to instruction generalization and language grounding, while the Robot score reflects sensitivity to robot initial-state variation. These require different invariances than the visual-domain invariance targeted by our objective.

This distinction is important for interpreting the Total score. DILL obtains a Total score of 50.8%, improving over OpenVLA, WorldVLA, UniVLA, and NORA, but remaining below OpenVLA-OFT due primarily to the Language and Robot categories. Rather than indicating a uniform robustness improvement across all axes, the result shows a more specific effect: DILL substantially reduces brittleness under visual distribution shifts, while leaving language and robot-state robustness as separate failure modes.

Appendix C
Representation Diagnostics

A. Pairwise latent similarity protocol

We use pairwise latent similarity diagnostics to test whether the action-conditioning representation is organized around task content rather than domain-specific visual factors. For each representation, we compute cosine similarity between normalized latent vectors for two types of unseen task–domain pairs.

Task pairs. A task pair consists of two observations that share the same underlying trajectory content but differ in visual domain condition. A task-centric latent should assign high similarity to these pairs because the task-relevant trajectory content is preserved across the domain change.

Domain pairs. A domain pair consists of two observations that share the same visual domain condition but differ in trajectory content. A task-centric latent should assign lower similarity to these pairs because the underlying behavior is different. Conversely, a domain-centric latent would assign high similarity to domain pairs, even when the trajectory content changes.

Metric. For a task-centric action representation representation, we summarize the diagnostic using the task–domain separation gap,

$$\begin{aligned}\Delta_{\text{task}} = \mathbb{E}_{(i,j)\in\mathcal{P}_{\text{task}}}[\text{sim}(z_i, z_j)] \\ - \mathbb{E}_{(i,j)\in\mathcal{P}_{\text{domain}}}[\text{sim}(z_i, z_j)].\end{aligned} \tag{33}$$

where $\mathcal{P}_{\text{task}}$ denotes task pairs and $\mathcal{P}_{\text{domain}}$ denotes domain pairs. A larger positive gap indicates that the representation is

Model	Original	LIBERO-Plus perturbations							Total
		Camera	Robot	Language	Light	BG	Noise	Layout	
OpenVLA [21]	76.5	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
		↓75.7	↓73.0	↓53.5	↓68.4	↓41.7	↓61.3	↓48.0	↓60.9
WorldVLA [10]	79.1	0.1	27.9	41.6	43.7	17.1	10.9	38.0	25.0
		↓79.0	↓51.2	↓37.5	↓35.4	↓62.0	↓68.2	↓41.1	↓54.1
UniVLA [9]	95.5	1.8	46.2	69.6	69.0	81.0	21.2	31.9	43.9
		↓93.7	↓49.3	↓25.9	↓26.5	↓14.5	↓74.3	↓63.6	↓51.6
NORA [19]	87.9	2.2	37.0	65.1	45.7	58.6	12.8	62.1	39.0
		↓85.7	↓50.9	↓22.8	↓42.2	↓29.3	↓75.1	↓25.8	↓48.9
OpenVLA-OFT [22]	95.3	10.4	38.7	70.5	76.8	93.6	49.9	69.9	55.8
		↓84.9	↓56.6	↓24.8	↓18.5	↓1.7	↓45.4	↓25.4	↓39.5
DILL (Ours)	81.6	68.4	21.4	2.9	69.4	70.0	68.7	55.1	50.8
		↓13.2	↓60.2	↓78.7	↓12.2	↓11.6	↓12.9	↓26.5	↓30.8

TABLE VIII
Full LIBERO-Plus robustness breakdown.

Task pair: same trajectory, different domain **Domain pair:** same domain, different trajectory

Fig. 7. **Pair construction for latent similarity diagnostics.** Task pairs test whether a latent remains stable across visual-domain changes when the underlying trajectory content is preserved. Domain pairs test whether a latent collapses observations that share visual-domain cues despite different trajectory content.

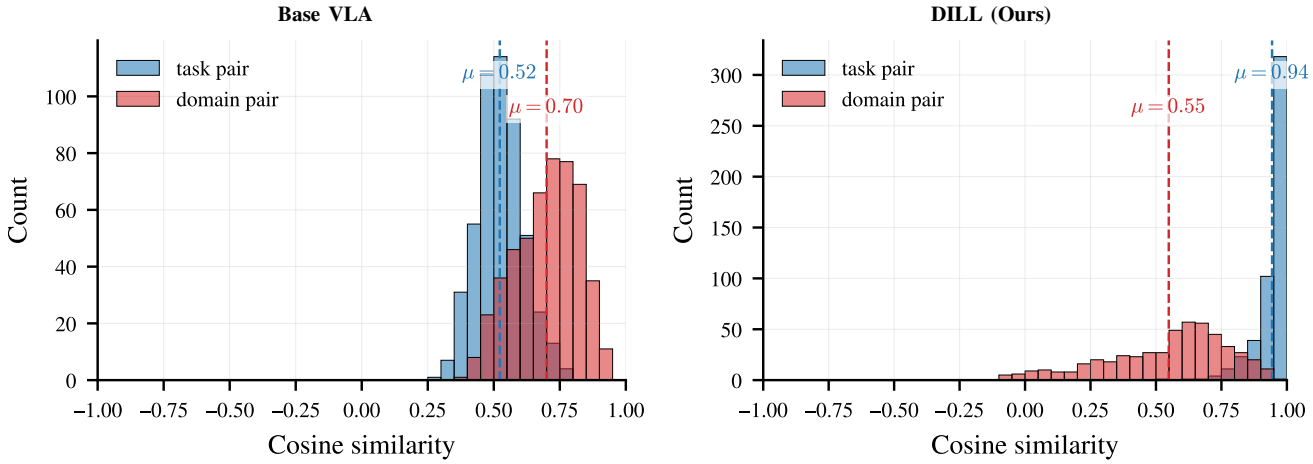


Fig. 8. **Final action-conditioning latent similarity diagnostics.**

more aligned with task content and less dominated by shared visual domain.

B. Final action-conditioning latent

The main paper analyzes the task encoder, domain encoder, and DILL's final action-conditioning latent (see Section IV-C). Here, we directly compare the final action-conditioning latent used by Base VLA and DILL. For Base VLA, this is the latent produced by the standard policy. For DILL, it is formed by concatenating the current representation and the predicted lookahead latent. A task-centric action-conditioning latent should assign higher similarity to task pairs than to domain pairs: observations with the same trajectory content should remain close across visual domains, while observations that merely share the domain should remain separated when their trajectory content differs.

Representation	Task-pair similarity ↑	Domain-pair similarity ↓	Gap Δ_{task} ↑
Base VLA final latent	0.52	0.70	−0.18
DILL final latent	0.94	0.55	0.39

TABLE IX
SUMMARY OF FINAL ACTION-CONDITIONING LATENT SIMILARITY.

a) Interpretation.: The final action-conditioning latent is directly tied to control, since it is the input used by the action head. Figure 8 and Table IX show that Base VLA has higher domain-pair similarity (0.70) than task-pair similarity (0.52), yielding a negative gap of −0.18 and indicating organization around shared visual domain. DILL reverses this ordering: task-pair similarity rises to 0.94, domain-pair similarity drops to 0.55, and the gap becomes 0.39. Thus, the action-conditioning latent becomes stable across domain changes when trajectory content is preserved, while remaining discriminative when trajectory content changes within the same visual domain. This supports the behavioral results by showing that DILL changes the geometry of the policy latent, not only its success rate under shifted views.