PELICAN-VLA 0.5: INDUCING TRANSFERABLE MANIPULATION-CENTRIC ATTENTION VIA BOTTLENECK TOKENS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generalization across objects, scenes, tasks, and embodiments remains a central challenge for Vision-Language-Action (VLA) models, which still rely heavily on task- and environment-specific robot data. We introduce **Pelican-VLA 0.5**, a unified model for vision-language understanding, future-frame prediction, and action generation. At its core is a compact set of learnable **Bot**tleneck **Tokens** (BotTokens), through which information is routed to the action pathway. This bottlenecked perception-to-action routing gives rise to **attention-level generalization**: without object annotations, segmentation masks, or explicit attention supervision, the model develops *manipulation-centric attention* during pre-training. Even before task-specific fine-tuning, its action pathway can localize task-relevant objects and contact regions across unseen scenes and robot embodiments. After fine-tuning, Pelican-VLA 0.5 achieves **98.6%** success on LIBERO, **91.4%** on *RoboTwin Clean*, and **91.0%** on *RoboTwin Randomized*. Across three real-robot embodiments, it performs comparably to $\pi_{0.5}$ under the same evaluation setting. Moreover, we find that its attention patterns remain highly consistent before and after fine-tuning, providing further evidence that the manipulation-centric attention acquired during pre-training is transferable. At the same time, the *attention-to-action gap* between correctly attending to manipulation-relevant regions and successfully executing actions remains a key obstacle to developing broadly generalizable robot policies.

1 INTRODUCTION

Building robots that can follow language instructions and operate in open, unstructured environments is a central goal of embodied intelligence. Vision-Language-Action (VLA) models provide a promising route toward this goal by mapping visual observations and language instructions directly to robot actions Zitkovich et al. (2023); Kim et al. (2024); Octo Model Team et al. (2024); Black et al. (2026); Liu et al. (2024); Qu et al. (2025); Bjorck et al. (2025). Despite rapid progress, most VLA models still rely heavily on task-, environment-, and embodiment-specific robot data. Transferring them to unfamiliar objects, scenes, tasks, or robot embodiments typically requires additional demonstrations and fine-tuning Kim et al. (2024); Black et al. (2025); Ye et al. (2026); Zha et al. (2026). Generalization, rather than performance after extensive adaptation, therefore remains a defining challenge for VLA research.

Learning a transferable robot policy requires more than fitting low-level action trajectories. Before producing an action, the policy must identify which visual evidence is relevant to the instruction and route that information into the action-generation pathway Black et al. (2025); Zha et al. (2026); Chen et al. (2026). However, when we inspect representative open-source VLA models, we find that their action pathways often attend diffusely across the robot arm, background clutter, and task-irrelevant objects, rather than consistently concentrating on the target object and potential contact region. Such attention may reflect reliance on environment-specific correlations and visual shortcuts Jia et al. (2026). This observation motivates a central question: *can a VLA model learn a transferable, manipulation-centric perception-to-action interface during pre-training, before receiving task-specific action supervision?*

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

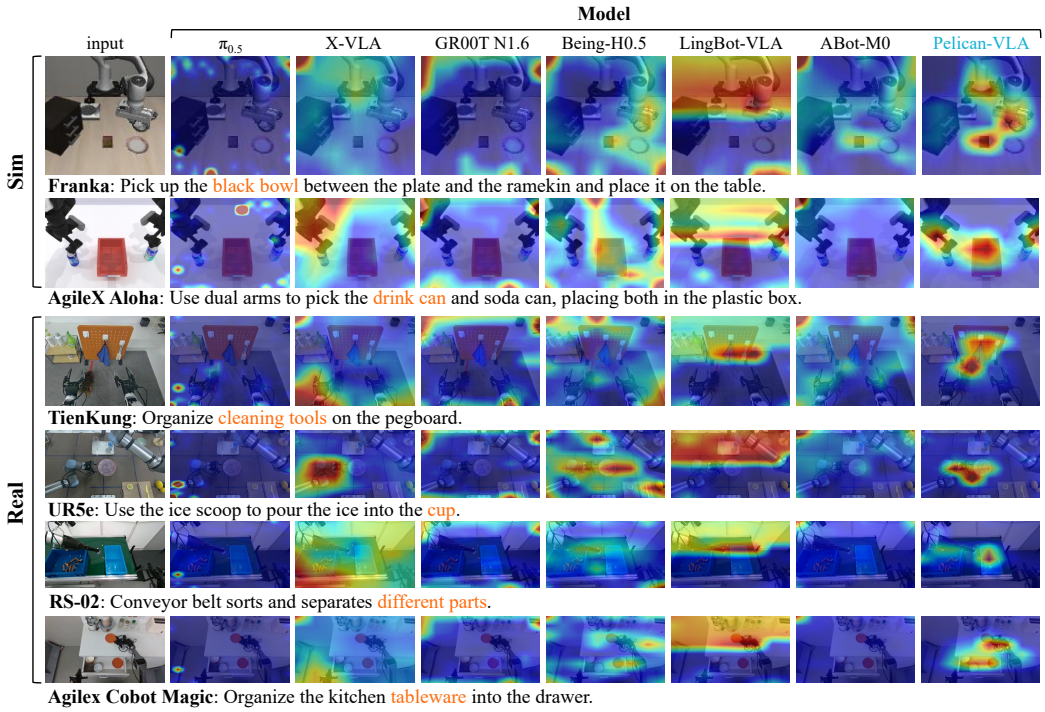


Figure 1: **Zero-shot attention comparison with open-source VLA baselines.** Before task-specific fine-tuning, Pelican-VLA 0.5 concentrates its action-pathway attention on the manipulation target and contact region, whereas the baselines exhibit more diffuse attention over the robot, surrounding objects, and background. The attention visualization settings in Appendix A.1.

We investigate this question through **Pelican-VLA 0.5**, a unified model for vision-language understanding, future-frame prediction, and action generation within a shared Transformer backbone. At its core is a compact set of learnable **Bot**tleneck **Tokens** (BotTokens), which form a constrained interface between perception and action. BotTokens selectively aggregate information from the upstream vision-language context, while the action pathway accesses perceptual information only through these bottleneck representations rather than directly attending to the dense visual tokens. By constraining the information flow from perception to action, this design encourages the model to organize manipulation-relevant cues—including the target object, potential contact regions, anticipated future changes, and current robot-state constraints—while reducing its reliance on task-irrelevant or redundant visual information.

This design gives rise to a striking pre-training behavior that we term **attention-level generalization**: the ability of perception-to-action attention to remain aligned with manipulation-relevant regions under shifts in objects, scenes, and robot embodiments, prior to task-specific fine-tuning. As shown in Fig. 1, Pelican-VLA 0.5 already concentrates its attention on target objects and potential contact regions in the zero-shot setting. This behavior emerges without object annotations, segmentation masks, or explicit attention supervision, and transfers to previously unseen visual environments and robot embodiments. Moreover, introducing the same routing constraint into an alternative policy architecture produces a similar attention pattern, suggesting that the phenomenon is not specific to a single policy structure.

The transferable attention learned during pre-training also persists through downstream adaptation. Pelican-VLA 0.5 achieves **98.6%** average success on LIBERO and **91.4%/91.0%** on RoboTwin Clean/Randomized, while performing comparably to $\pi_{0.5}$ across three real-robot embodiments. Its attention patterns remain highly similar before and after fine-tuning, suggesting that adaptation mainly converts a pre-formed manipulation-centric representation into executable actions. Nevertheless, strict zero-shot rollouts still exhibit failures in precise grasping and placement, revealing an **attention-to-action gap** between transferable target grounding and transferable control.

Our contributions are summarized as follows:

- We introduce **Pelican-VLA 0.5**, a unified architecture for vision-language understanding, future-frame generation, and action prediction, in which learnable BotTokens serving as a compact perception-to-action interface.
- We show that compressing the perception-to-action pathway (BotTokens) induces **attention-level generalization**: manipulation-centric attention emerges without spatial supervision and transfers across objects, scenes, embodiments, and policy architectures.
- Our model achieves strong simulation and real-world performance, including high success rates on RoboTwin and results comparable to $\pi_{0.5}$ across three robot embodiments. Further analysis reveals an *attention-to-action gap* between transferable perception and control.

2 Related Work

2.1 Vision Language Action Model

VLMs pretrained on internet-scale data have demonstrated impressive world knowledge and cross-distribution generalization capabilities. A natural idea is to transfer this knowledge into VLA models for general-purpose embodied intelligence. Early works such as RT2 Zitkovich et al. (2023) and OpenVLA Kim et al. (2024) directly tokenize and autoregressively generate actions, but face challenges in continuous control precision, action generation efficiency, and OOD generalization. Pi0 Black et al. (2026), pi0.5 Black et al. (2025), GR00T Bjorck et al. (2025), etc. adopt MoT architectures and flow matching to generate action chunks through multi-step denoising, achieving higher action precision and control frequencies. Despite their capability to accomplish complex tasks, they still rely on task-specific data fine-tuning to achieve satisfactory success rates on novel tasks or new embodiment types, which has yet to meet the expectation of general-purpose embodied intelligence. Recent works Jiang et al. (2025b); Zheng et al. (2026); Li et al. (2026); Yang et al. (2025) have focused on the use of more scalable data sources such as Ego-Centric data, which provide more diverse visual and semantic knowledge. However, whether such diversity can be effectively translated into highly generalizable action capabilities remains unproven.

2.2 Perception-to-Action Representation in VLA

Existing VLA models rely on VLM knowledge to recognize and localize target objects, providing the foundation for the Action Expert to generate reasonable actions. However, how to efficiently extract action-relevant information from VLMs remains an open problem. GR00T Bjorck et al. (2025) uses only the last hidden state as the conditional representation for the Action Expert, while MoT architectures such as pi0 Black et al. (2026) and pi0.5 Black et al. (2025) pass layer-wise KV-Cache to the Action Expert. Both approaches expect the model to learn how to utilize such information through large-scale training. Wall-oss 0.5 Zhai et al. (2025) and G0.5 Galaxea Team (2026) additionally introduce visual grounding tasks, improving the VLM's ability to localize target objects. LingBot-VLA 2.0 Wu et al. (2026b) and Motus Bi et al. (2026) supervise future observation representations, encouraging understanding of the world and temporal information. These approaches often require carefully designed losses, architectures, or additional supervised labels, introducing extra training costs. In this work, we propose Bottleneck Tokens to constrain the VLM through an information bottleneck, encouraging it to focus on details critical for action generation. This enables self-supervised training without additional annotations or auxiliary losses, while yielding generalization at both the representation and attention levels.

3 Pelican-VLA 0.5

3.1 Overview

As shown in Fig. 2, Pelican-VLA 0.5 is a unified Vision-Language-Action model built on a single shared Qwen3-VL 4B backbone Bai et al. (2025). The model is designed to support three tightly coupled functions within one Transformer: visual-language understanding, future-frame prediction, and action generation. Unlike dual-system or Mixture-of-Transformers designs Cai et al. (2026); Liu et al. (2026b), Pelican-VLA 0.5 does not introduce separate reasoning and acting experts. Instead, all functions operate over a shared token stream and communicate through a common hidden state.

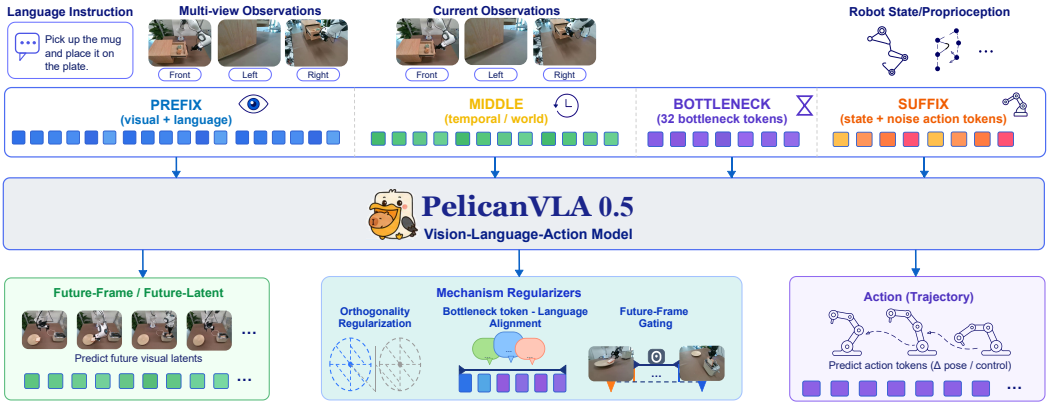


Figure 2: **Overview of Pelican-VLA 0.5.** Pelican-VLA 0.5 uses a shared Qwen3-VL 4B backbone to unify visual-language understanding, future-frame prediction, and action generation within a single Transformer. The input sequence contains four segments: vision-language *prefix* tokens, Cosmos latent *middle* tokens, K learnable *BotTokens*, and a *suffix* containing proprioceptive states and noisy action chunks. A single forward pass predicts future-frame latents from the middle, a compact task representation from the BotTokens, and action denoising velocities from the suffix.

The input sequence is organized into four contiguous segments:

$$\mathbf{Z} = \big[\ \underbrace{\text{Prefix}}_{\text{vision-language}}\ ;\ \underbrace{\text{Middle}}_{\text{Cosmos latents}}\ ;\ \underbrace{\text{Bottleneck}}_{K\text{ bottleneck tokens}}\ ;\ \underbrace{\text{Suffix}}_{\text{state and noisy action}}\ \big]. \tag{1}$$

A single forward pass produces three outputs in parallel: action denoising velocities from the suffix, future-frame latents from the middle, and a compact task representation from the BotTokens.

This section first describes the unified token sequence and attention geometry in §3.2. We then introduce the BotTokens and its auxiliary regularizers in §3.3, followed by the training objectives in §A.2 and the cached inference procedure in §A.3.

3.2 Unified Token Sequence

Prefix: vision-language tokens. The current camera observations and language instruction are encoded by the Qwen3-VL. This produces the prefix sequence $\mathbf{P} \in \mathbb{R}^{L_p \times D}$. The prefix uses bidirectional attention so that language tokens and visual tokens can mutually contextualize each other.

Middle: Cosmos latent tokens. In parallel, we encode a short visual history using a frozen Cosmos-Tokenizer Agarwal et al. (2025). For each view, two frames at time $t-15$ and t are mapped into continuous latent features. The features are flattened into the middle sequence $\mathbf{M} \in \mathbb{R}^{L_m \times D}$. The middle segment provides a pixel-level and dynamics-aware view of the scene, complementing the semantic patch tokens in the prefix.

Bottleneck: learnable BotTokens We introduce K learnable BotTokens, $\mathbf{S}_0 \in \mathbb{R}^{1 \times K \times D}$, initialized from $\mathcal{N}(0, 0.02^2)$ and broadcast across the batch. During training, BotTokens dropout with probability $p = 0.1$ is applied:

$$\mathbf{S} = \text{drop}_p(\mathbf{S}_0) \in \mathbb{R}^{B \times K \times D}. \tag{2}$$

Since K is much smaller than the number of upstream visual tokens, the BotTokens form a fixed-capacity interface between perception and action. Their attention to the upstream context provides a direct way to inspect what information is routed into the action pathway.

Suffix: proprioceptive state and noisy action. The suffix contains the robot state and the noisy action chunk used by the flow-matching. The proprioceptive state is projected into a single state token. The noisy action chunk $\mathbf{x}_\tau \in \mathbb{R}^{H \times d_a}$ is embedded by a linear layer. The suffix sequence is therefore $\mathbf{U} \in \mathbb{R}^{(1+H) \times D}$.

Attention geometry. The full sequence is $\mathbf{Z} = [\mathbf{P}; \mathbf{M}; \mathbf{S}; \mathbf{U}]$. We construct a two-dimensional attention mask from per-token block flags, following the prefix-LM and block-causal construction used in prior VLA architectures Black et al. (2026). The prefix, middle, and BotTokens segments are internally bidirectional. The suffix uses causal attention with respect to its visible context.

3.3 BOTTLENECK TOKENS

Pelican-VLA 0.5 enforces this bottleneck through three mechanisms: a curriculum attention mask, an orthogonality regularizer, and a token-gated generation.

Curriculum bottleneck mask. Let $\mathcal{C} = \{\text{prefix}\} \cup \{\text{middle}\}$ denote the upstream perception tokens, and let $\mathcal{S}$ denote the BotTokens. We define a binary attention-visibility mask A, where $A_{i \to j} = 1$ means that query token i is allowed to attend to key/value token j, and $A_{i \to j} = 0$ means that this attention edge is masked out. In the hard bottleneck regime, we remove all direct attention edges from suffix tokens to the upstream perception tokens:

$$A_{i \to j} = 0, \qquad \forall i \in \text{suffix},\ j \in \mathcal{C}. \tag{3}$$

The suffix can still attend to the BotTokens and to its permitted suffix context. Thus, the only path from perception to action passes through $\mathcal{S}$.

Applying this hard constraint from the beginning of training can make optimization unstable. We use a curriculum schedule. At training step t, the hard bottleneck mask is applied with probability

$$p_t = \min\left(1, \frac{t}{T_{\text{warm}}}\right), \qquad T_{\text{warm}} = 10\text{k}. \tag{4}$$

With probability $1 - p_t$, the suffix is allowed to directly attend to the upstream context. This schedule lets the model first learn with an easier dense channel and then progressively shifts perception-to-action communication through the BotTokens.

Orthogonality regularizer. A compact tokens set is only useful if different BotTokens capture complementary information. Let $\mathbf{Z}^{\text{tokens}} \in \mathbb{R}^{B \times K \times D}$ be the final BotTokens outputs of the Transformer. For each batch element b, let $\hat{\mathbf{Z}}_b^{\text{tokens}} \in \mathbb{R}^{K \times D}$ denote the BotTokens-output matrix after ℓ_2-normalizing each BotTokens vector along the hidden dimension. We encourage BotTokens diversity by penalizing deviations of the BotTokens Gram matrix from identity:

$$\mathcal{L}_{\text{reg}} = \frac{1}{B} \sum_{b=1}^{B} \frac{1}{K^2} \left\| \hat{\mathbf{Z}}_b^{\text{tokens}} \left(\hat{\mathbf{Z}}_b^{\text{tokens}}\right)^\top - \mathbf{I}_K \right\|_F^2. \tag{5}$$

Here $\mathbf{I}_K$ is the $K \times K$ identity matrix and $\| \cdot \|_F$ denotes the Frobenius norm. This prevents all BotTokens from collapsing to the same direction and encourages them to encode distinct factors of the scene, task, or action context.

Bottleneck tokens gating of generation. The BotTokens are also connected to the future-frame prediction branch. From the mean-pooled BotTokens output

$$\bar{\mathbf{z}} = \frac{1}{K} \sum_{k=1}^{K} \mathbf{z}_k^{\text{tokens}}, \tag{6}$$

we compute a channel-wise gate $\mathbf{g} = \sigma(\mathbf{W}_g \bar{\mathbf{z}})$, where $\mathbf{W}_g$ projects the pooled BotTokens feature to the channel dimension of the generation features. The gate modulates the generation features before they are decoded back into Cosmos latent space:

$$\tilde{\mathbf{F}}_{\text{gen}} = \mathbf{F}_{\text{gen}} \odot \mathbf{g}. \tag{7}$$

Here $\mathbf{g}$ is broadcast over the token or spatial dimensions of $\mathbf{F}_{\text{gen}}$.

3.4 TRAINING OBJECTIVES

Pelican-VLA 0.5 is trained with a multi-task objective that combines action generation, future-frame prediction, language-task alignment, and BotTokens regularization(The each loss details in A.2.) The final objective is

$$\mathcal{L} = \mathcal{L}_{\text{action}} + \lambda_{\text{gen}}\,\mathcal{L}_{\text{gen}} + \lambda_{\text{task}}\,\mathcal{L}_{\text{task}} + \lambda_{\text{reg}}\,\mathcal{L}_{\text{reg}}. \tag{8}$$

We set $\lambda_{\text{gen}} = 0.01$, $\lambda_{\text{task}} = 0.1$, $\lambda_{\text{reg}} = 0.01$. The bottleneck schedule is advanced by the global training step, so the perception-to-action pathway is gradually constrained as optimization proceeds.

4 EXPERIMENTS

4.1 TRAINING DATA

Pelican-VLA 0.5 is pre-trained on a heterogeneous, cross-embodiment mixture of large-scale manipulation corpora. Dataset and experimental details are provided in A.4.

4.2 SIMULATION RESULTS

Simulation benchmark. On the RoboTwin benchmark Chen et al. (2025), Pelican-VLA attains an average success rate of $\mathbf{91.4}\%$ under the *clean* setting and $\mathbf{91.0}\%$ under the *randomized* setting, as summarized in Table 1. On the Libero benchmark, Pelican-VLA attains an average success rate of $\mathbf{98.6}\%$ (See Appendix Table 5 for details).

Table 1: Benchmark results on RoboTwin. Compared with recent VLA baselines, Pelican-VLA 0.5 achieves the best performance.

Methods	Clean	Randomized	Average
π_0 Black et al. (2026)	80.0	79.5	79.8
$\pi_{0.5}$ Black et al. (2025)	86.8	87.0	86.9
ABot-M0 Yang et al. (2026)	86.1	85.1	85.6
LingBot-VLA Wu et al. (2026a)	88.6	86.7	87.7
Qwen-VLA Wang et al. (2026)	86.1	87.2	86.7
JoyAI-RA Zhang et al. (2026b)	90.5	89.3	89.9
Hy-VLA Zhang et al. (2026a)	<u>90.9</u>	<u>90.1</u>	<u>90.5</u>
Pelican-VLA 0.5	**91.4**	**91.0**	**91.2**

4.3 REAL-WORLD ROBOT EVALUATION

To evaluate real-world performance, we conduct experiments on nine tasks across three robot embodiments: Tien Kung, AgileX, and UR5e. The tasks cover precise manipulation, multi-step organization, assembly, cleaning, and bimanual coordination. We evaluate each task in its original setting and under changes in object placement, background color, and illumination. Pelican-VLA 0.5 and $\pi_{0.5}$ use the same task-specific data, initial configurations, and success criteria. Each method is evaluated over 25 trials per task and setting, with full-task success rate as the primary metric. Detailed task instructions and experimental settings are provided in Appendix A.9 and Fig. 11.

As shown in Fig. 3, Pelican-VLA 0.5 achieves an average success rate of 63.6% in the original setting, compared with 61.8% for $\pi_{0.5}$, indicating similar overall performance. In the perturbed setting, the average success rates are 52.0% and 40.9%, respectively. Both methods degrade under scene perturbations, with an average decrease of 11.6 percentage points for Pelican-VLA 0.5 and 20.9 points for $\pi_{0.5}$. These results show a smaller average performance drop for Pelican-VLA 0.5 under the tested perturbations, although performance varies across tasks.

4.4 ZERO-SHOT GENERALIZATION.

Beyond the in-distribution benchmark above, we evaluate whether Pelican-VLA can act *without task-specific fine-tuning*. We directly deploy the pre-trained policy on RoboTwin 2.0, which is en-

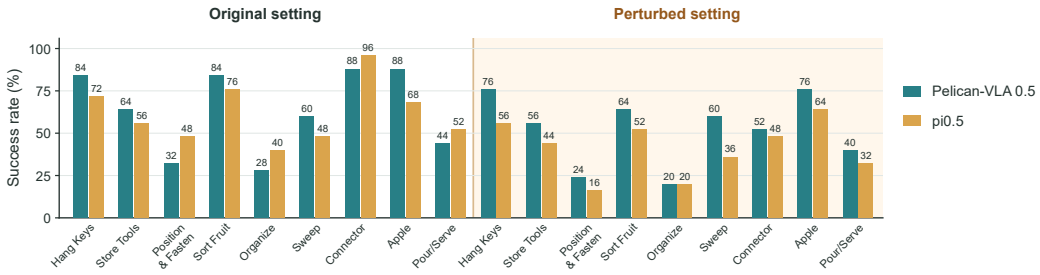


Figure 3: Real-world success rates across nine tasks.

tirely excluded from pre-training and introduces unseen objects, scene layouts, and robot embodiments (details in Appendix A.6).

As shown in Appendix Fig. 8, Pelican-VLA exhibits zero-shot manipulation behavior across tasks. The policy consistently reaches toward manipulation-relevant objects and produces coherent, goal-directed motions rather than random or degenerate behavior. These results suggest that the manipulation-centric attention acquired during pre-training transfers across unseen objects, scenes, and embodiments. Table 2 summarizes the corresponding success rates on five RoboTwin tasks.

Table 2: Zero-shot results on five RoboTwin tasks.

Method	bottle horizontally	shake bottle	click bell	open microwave	click alarmclock
VLA w/o BotToken	31.0	27.0	27.0	18.0	20.0
Pelican-VLA 0.5	49.0	45.0	42.0	37.0	25.0

5 Analysis

5.1 Disentangling Data from Architecture

A first hypothesis is that the representation is already latent in the demonstrations, such that *any* sufficiently capable policy trained on them would acquire it. We hold the data, optimizer, schedule, and parameter budget fixed, and we vary only the architecture, training a dense Mixture-of-Transformers (MoT) policy Cai et al. (2026) whose action expert attends *densely* to all upstream perception.

The two models differ substantially, as shown in Fig. 4. The dense MoT policy reaches a comparable training action loss, indicating that it can fit the demonstrations under the same data and compute budget. However, its attention remains diffuse and only weakly aligned with the target object. This suggests that the representation is induced by the architecture rather than training data.

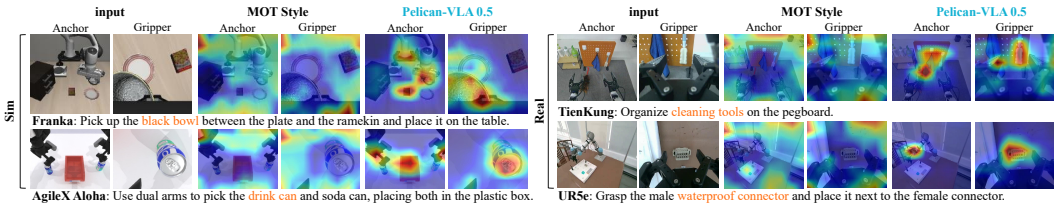


Figure 4: Comparison of attention patterns between Pelican-VLA 0.5 and a dense MoT baseline.

5.2 Architecture Ablation

In this section, we construct an ablation ladder that introduces one mechanism at a time while keeping the backbone, data, and training budget fixed. The ladder contains five configurations: 1. action loss only; 2. action loss + future-frame generation loss $\mathcal{L}_{\text{gen}}$; 3. action loss + language contrastive

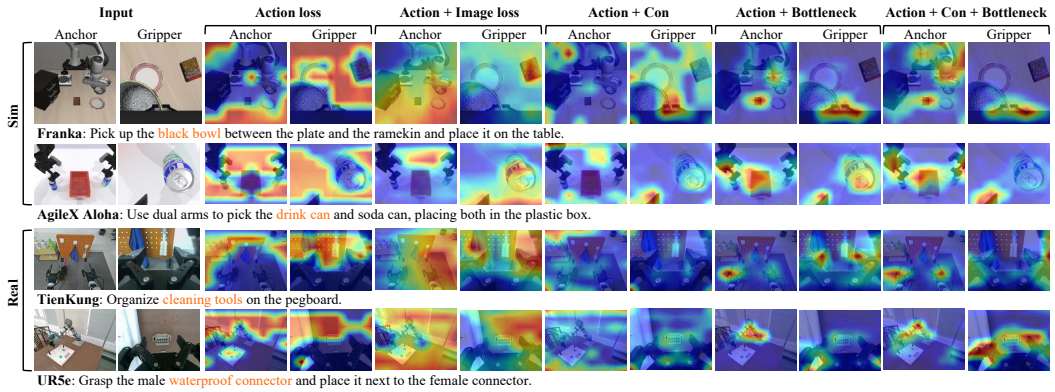


Figure 5: Component-wise ablation. Manipulation-relevant attention emerges only when perception is routed through the BotTokens.

loss $\mathcal{L}_{\text{task}}$, applied to a dense model without BotTokens;4. action loss + the BotTokens; 5. action loss + the BotTokens + contrastive loss.

The ablation yields a clear result. The generation loss leaves the representation unchanged, and the contrastive loss imposed on a dense model likewise has no effect—both configurations leave the attention near chance. The representation emerges *only* at row 4, precisely when perception is routed through the BotTokens, at which point it becomes pronounced. That a single mechanism accounts for the entire effect is notable; the evidence nonetheless is unambiguous: the auxiliary losses are not the cause, whereas the BotTokens is. Together with §5.1, this completes the attribution: the representation is induced specifically by routing perception through the narrow BotTokens.

Language-conditioned attention. In an early checkpoint, as shown in Fig. 6, the BotTokens-only model tends to localize *an* object, often the visually dominant or frequently manipulated one, regardless of the instruction. After introducing the contrastive task loss, the same scene exhibits language-conditioned behavior: changing the instruction from *Pick up the black bowl between the plate and the ramekin and place it on the table."* to *Open the drawer of the black cabinet."* shifts the attention from the bowl to the cabinet, following the object specified by the language. This suggests that the contrastive loss does not create attentional focus by itself, but can make the BotTokens content more conditional on language rather than only on visual salience.

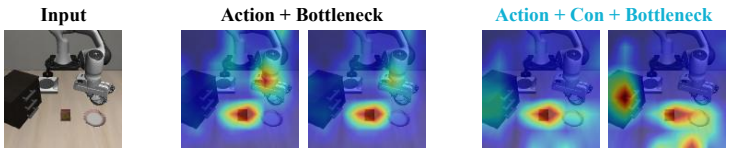


Figure 6: Changing the instruction shifts attention to the corresponding target.

Bottleneck tokens transferability across architectures. To further examine whether the effect of BotTokens is tied to the specific Pelican-VLA architecture, we also insert the same BotTokens interface into a MoT-style architecture, as illustrated in Fig. 7. The resulting model exhibits a similar attention pattern: despite the change in backbone organization and action-prediction pathway, the BotTokens consistently concentrate on regions related to the manipulated object and its contact area. This suggests that the BotTokens are not merely an artifact of our particular architecture, but can serve as a more general interface for routing task-relevant visual information toward action generation. While the strength and language selectivity of this behavior may still depend on the training objective and data quality, the MoT-based result indicates that BotTokens provide a transferable mechanism for inducing manipulation-centric attention across different VLA designs.

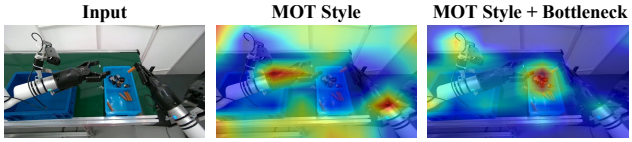


Figure 7: Bottleneck tokens transferability across architectures.

5.3 Attention dynamics over pre-training

To characterize how representation emerges during pre-training, we track attention through the BotTokens across checkpoints of the full model, as shown in Appendix Fig.9, A.7 and Tab.3. Tab. 3 provides quantitative evidence for this trend. Task-related regions remain a strong attention recipient throughout pre-training, with an IoU around 0.35 and no significant monotonic trend ($\rho = -0.43$, $p = 0.21$). In contrast, attention alignment with the target object increases significantly, with IoU rising from 0.054 to 0.124 and showing a strong positive monotonic trend across checkpoints ($\rho = 0.76$, $p = 0.01$). Manipulation-centric attention can emerge through pre-training under a BotTokens-mediated architecture. In addition, removing the BotTokens after training does not fully eliminate the model's ability to attend to the manipulation-relevant object, indicating that the BotTokens help internalize manipulation-centric attention into the shared backbone and allows the learned representation to persist as a property of the trained model itself.

Table 3: Attention dynamics across pre-training checkpoints.

Region	**IoU (50k)**	**IoU (500k)**	**ΔIoU**	**Spearman ρ**
Manipulator (arm + gripper)	0.360	0.350	-0.010	-0.43 (p = 0.21)
Target object (bottle)	0.054	0.124	+0.070	0.76 (p = 0.01)

5.4 Zero-shot vs fine-tuned representations

In this section, we compare the zero-shot model and its fine-tuned counterpart on the same held-out task, using attention visualizations, representation maps, and representation-similarity measures. As shown in Appendix Fig. 10, fine-tuning changes the representation pattern only marginally. The zero-shot and fine-tuned models attend to highly similar object regions, and their BotTokens representations remain close under representation-similarity analysis.

To quantify this observation, we measure target attention IoU, attention similarity, and BotTokens representation similarity between the zero-shot model and fine-tuned checkpoints, as summarized in Table 4. The target attention IoU remains stable throughout fine-tuning, while the attention similarity to the zero-shot model remains above 0.9, indicating that task adaptation preserves the manipulation-relevant attention pattern and does not substantially alter where the model attends. This result supports the attention-to-action gap introduced earlier. The manipulation-centric attention needed to identify *what* to act upon is largely present before fine-tuning.

Table 4: Target Attention Alignment and Attention Similarity Before and After Fine-Tuning

Stage	**Target Attention IoU**	**Attention Similarity**
Before FT	0.124	1.000
During FT	0.134	0.932
After FT	0.127	0.928

6 Conclusion

In this paper, we introduce Pelican-VLA 0.5, which compact BotTokens interface induces manipulation-centric attention without explicit spatial supervision and transfers across unseen scenes and robot embodiments. Pelican-VLA 0.5 achieves strong results on LIBERO and RoboTwin and performs comparably to $\pi_{0.5}$ across three real-robot embodiments.

AI USE STATEMENT

In this work, we used generative AI tools for assisting with translation, supporting qualitative and thematic data analysis, editing software code, drafting parts of a research paper, summarizing or analyzing existing literature, identifying gaps, brainstorming, sourcing and searching for information, editing a research paper to improve readability, identifying relevant literature, proposing a title or keywords for a research paper, and interpreting results. We have not used generative AI tools for generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses.

We have reviewed all AI-assisted work. AI-assisted analyses, interpretations, and other research-related outputs were independently checked by the authors, and the final decisions, claims, and conclusions were made and verified by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

This paper presents a novel VLA framework. We provide pelican vla 0.5 code in the supplementary materials, and the important hyper-parameters and training recipe are offered in Section A.4. Detailed experimental settings can be found in Section A.6 and Section A.9. We will open-source the pretrained model weights in the future. We will do our utmost to ensure that our experimental results and conclusions are reproducible, thereby advancing the development of the embodied intelligence community.

REFERENCES

Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical ai, 2025. URL https://arxiv.org/abs/2501.03575.

AgiBot-World-Contributors, Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, Shu Jiang, Yuxin Jiang, Cheng Jing, Hongyang Li, Jialu Li, Chiming Liu, Yi Liu, Yuxiang Lu, Jianlan Luo, Ping Luo, Yao Mu, Yuehan Niu, Yixuan Pan, Jiangmiao Pang, Yu Qiao, Guanghui Ren, Cheng Ruan, Jiaqi Shan, Yongjian Shen, Chengshi Shi, Mingkang Shi, Modi Shi, Chonghao Sima, Jianheng Song, Huijie Wang, Wenhao Wang, Dafeng Wei, Chengen Xie, Guo Xu, Junchi Yan, Cunbiao Yang, Lei Yang, Shukai Yang, Maoqing Yao, Jia Zeng, Chi Zhang, Qinglin Zhang, Bin Zhao, Chengyue Zhao, Jiaqi Zhao, and Jianchao Zhu. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems, 2025. URL https://arxiv.org/abs/2503.06669.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun

Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025. URL https://arxiv.org/abs/2511.21631.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, et al. Motus: A unified latent action world model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 35101–35113, 2026.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

Junhao Cai, Zetao Cai, Jiafei Cao, Yilun Chen, Zeyu He, Lei Jiang, Hang Li, Hengjie Li, Yang Li, Yufei Liu, Yanan Lu, Qi Lv, Haoxiang Ma, Jiangmiao Pang, Yu Qiao, Zherui Qiu, Yanqing Shen, Xu Shi, Yang Tian, Bolun Wang, Hanqing Wang, Jiaheng Wang, Tai Wang, Xueyuan Wei, Chao Wu, Yiman Xie, Boyang Xing, Yuqiang Yang, Yuyin Yang, Qiaojun Yu, Feng Yuan, Jia Zeng, Jingjing Zhang, Shenghan Zhang, Shi Zhang, Zhuoma Zhaxi, Bowen Zhou, Yuanzhen Zhou, Yunsong Zhou, Hongrui Zhu, Yangkun Zhu, and Yuchen Zhu. Internvla-a1: Unifying understanding, generation and action for robotic manipulation, 2026. URL https://arxiv.org/abs/2601.02456.

Haonan Chen, Jingxiang Guo, Bangjun Wang, Tianrui Zhang, Xuchuan Huang, Boren Zheng, Yiwen Hou, Chenrui Tie, Jiajun Deng, and Lin Shao. Goal-vla: Image-generative vlms as object-centric world models empowering zero-shot robot manipulation, 2026. URL https://arxiv.org/abs/2506.23919.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation, 2025. URL https://arxiv.org/abs/2506.18088.

Galaxea Team. Galaxea g0.5: One autoregressive stream for robot reasoning and action. *arXiv preprint arXiv:2608.11739*, 2026. URL https://arxiv.org/abs/2608.11739.

Xiaosong Jia, Bowen Yang, Zuhao Ge, Xian Nie, Yuchen Zhou, Cunxin Fan, Yufeng Li, Yilin Chai, Chao Jing, Zijian Liang, Qingwen Bu, Haidong Cao, Chao Wu, Qifeng Li, Zhenjie Yang, Chenhe Zhang, Hongyang Li, Zuxuan Wu, Junchi Yan, and Yu-Gang Jiang. Guidedvla:

Specifying task-relevant factors via plug-and-play action attention specialization, 2026. URL https://arxiv.org/abs/2605.12369.

Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, Jianning Cui, Xiao Liu, Shuiqi Cheng, Jiyang Gao, Huazhe Xu, and Hang Zhao. Galaxea open-world dataset and g0 dual-system vla model, 2025a. URL https://arxiv.org/abs/2509.00576.

Yuming Jiang, Siteng Huang, Shengke Xue, Yaxi Zhao, Jun Cen, Sicong Leng, Kehan Li, Jiayan Guo, Kexiang Wang, Mingxiu Chen, Fan Wang, Deli Zhao, and Xin Li. Rynnvla-001: Using human demonstrations to improve robot manipulation. *arXiv preprint arXiv:2509.15212*, 2025b.

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.

Hao Li, Ganlong Zhao, Yufei Liu, Haotian Hou, Guoquan Ye, Tongyan Fang, Chunxiao Liu, Siyuan Huang, Jianbo Liu, Xiaogang Wang, et al. Ace-ego-0: Unifying egocentric human and robotic data for vla pretraining. *arXiv preprint arXiv:2606.17200*, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Mengya Liu, Baoxiong Jia, Jiangyong Huang, Jingze Zhang, and Siyuan Huang. Lara: Latent action representation alignment for vision-language-action models, 2026a. URL https://arxiv.org/abs/2606.07100.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: A diffusion foundation model for bimanual manipulation, 2024. URL https://arxiv.org/abs/2410.07864.

Zhuoyang Liu, Jiaming Liu, Hao Chen, Jiale Yu, Ziyu Guo, Chengkai Hou, Chenyang Gu, Xiangju Mi, Renrui Zhang, Kun Wu, Zhengping Che, Jian Tang, Pheng-Ann Heng, and Shanghang Zhang. Last_0: Latent spatio-temporal chain-of-thought for robotic vision-language-action model, 2026b. URL https://arxiv.org/abs/2601.05248.

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*, 2024. doi: 10.15607/RSS.2024.XX.090.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model, 2025. URL https://arxiv.org/abs/2501.15830.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021.

Yang Tian, Yuyin Yang, Yiman Xie, Zetao Cai, Xu Shi, Ning Gao, Hangxu Liu, Xuekun Jiang, Zherui Qiu, Feng Yuan, Yaping Li, Ping Wang, Junhao Cai, Jia Zeng, Hao Dong, and Jiangmiao Pang. Interndata-a1: Pioneering high-fidelity synthetic data for pre-training generalist policy, 2025. URL https://arxiv.org/abs/2511.16651.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Qiuyue Wang, Mingsheng Li, Jian Guan, Jinhui Ye, Sicheng Xie, Yitao Liu, Junhao Chen, Zhixuan Liang, Jie Zhang, Xintong Hu, Xuhong Huang, Pei Lin, Junyang Lin, Dayiheng Liu, Shuai Bai, Jingren Zhou, Jiazhao Zhang, Haoqi Yuan, Gengze Zhou, Hang Yin, Ye Wang, Yiyang Huang, Zixing Lei, Wujian Peng, Delin Chen, Yingming Zheng, Jingyang Fan, Xianwei Zhuang, Xin Zhou, Haoyang Li, Anzhe Chen, Tong Zhang, Xuejing Liu, Yuchong Sun, Ruizhe Chen, Zhaohai Li, Chenxu Lü, Zhibo Yang, Tao Yu, and Xionghui Chen. Qwen-vla: Unifying vision-language-action modeling across tasks, environments, and robot embodiments, 2026. URL https://arxiv.org/abs/2605.30280.

Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, Shichao Fan, Xinhua Wang, Fei Liao, Zhen Zhao, Guangyu Li, Zhao Jin, Lecheng Wang, Jilei Mao, Ning Liu, Pei Ren, Qiang Zhang, Yaoxu Lyu, Mengzhen Liu, He Jingyang, Yulin Luo, Zeyu Gao, Chenxuan Li, Chenyang Gu, Yankai Fu, Di Wu, Xingyu Wang, Sixiang Chen, Zhenyu Wang, Pengju An, Siyuan Qian, Shanghang Zhang, and Jian Tang. Robomind: Benchmark on multi-embodiment intelligence normative data for robot manipulation. In *Robotics: Science and Systems XXI*, RSS2025. Robotics: Science and Systems Foundation, 2025. doi: 10.15607/rss.2025.xxi.152. URL http://dx.doi.org/10.15607/RSS.2025.XXI.152.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, Yiyu Ren, Kejia Zhang, Hui Yu, Jingmei Zhao, Shuai Zhou, Zhenqi Qiu, Houlong Xiong, Ziyu Wang, Zechen Wang, Ran Cheng, Yong-Lu Li, Yongtao Huang, Xing Zhu, Yujun Shen, and Kecheng Zheng. A pragmatic vla foundation model, 2026a. URL https://arxiv.org/abs/2601.18692.

Wei Wu, Fangjing Wang, Fan Lu, He Sun, Shi Liu, Yunnan Wang, Yibin Yan, Yong Wang, Shuailei Ma, Xinyang Wang, Yibin Liu, Shuai Yang, Tianxiang Zhou, Kejia Zhang, Lei Zhou, Cheng Su, Nan Xue, Bin Tan, Han Zhang, Youchao Zhang, Fei Liao, Xing Zhu, Yujun Shen, and Kecheng Zheng. From foundation to application: Improving vla models in practice. *arXiv preprint arXiv:2607.06403*, 2026b.

Ruihan Yang, Qinxi Yu, Yecheng Wu, Rui Yan, Borui Li, An-Chieh Cheng, Xueyan Zou, Yunhao Fang, Xuxin Cheng, Ri-Zhao Qiu, et al. Egovla: Learning vision-language-action models from egocentric human videos. *arXiv preprint arXiv:2507.12440*, 2025.

Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, Feng Xiong, Xing Wei, Zhiheng Ma, and Mu Xu. Abot-m0: Vla foundation model for robotic manipulation with action manifold learning, 2026. URL https://arxiv.org/abs/2602.11236.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, et al. World action models are zero-shot policies, 2026. URL https://arxiv.org/abs/2602.15922.

Lihan Zha, Asher J. Hancock, Mingtong Zhang, Tenny Yin, Yixuan Huang, Dhruv Shah, Allen Z. Ren, and Anirudha Majumdar. Lap: Language-action pre-training enables zero-shot cross-embodiment transfer, 2026. URL https://arxiv.org/abs/2602.10556.

Andy Zhai, Brae Liu, Bruno Fang, Chalse Cai, Ellie Ma, Ethan Yin, Hao Wang, Hugo Zhou, James Wang, Lights Shi, Lucy Liang, Make Wang, Qian Wang, Roy Gan, Ryan Yu, Shalfun Li, Starrick Liu, Sylas Chen, Vincent Chen, and Zach Xu. Igniting vlms toward the embodied space. *arXiv preprint arXiv:2509.11766*, 2025.

He Zhang, Lingzhu Xiang, Haitao Lin, Zeyu Huang, Minghui Wang, Dingyan Zhong, Yubo Dong, Yihao Wu, Yongming Rao, Dongsheng Zhang, Wanjia He, Ling Chen, Kai Huang, Jiahao Chen, Sichang Su, Xumin Yu, Ziyi Wang, Chengwei Zhu, Xiao Teng, Yuchun Guo, Yufeng Zhang, Yuandong Liu, Rui Wang, Zisheng Lu, Han Hu, and Zhengyou Zhang. Hy-embodied-0.5-vla: From vision-language-action models to a real-world robot learning stack, 2026a. URL https://arxiv.org/abs/2606.14409.

Tianle Zhang, Zhihao Yuan, Dafeng Chi, Peidong Liu, Dongwei Li, Kejun Hu, Likui Zhang, Junnan Nie, Ziming Wei, Zengjue Chen, Yili Tang, Jiayi Li, Zhiyuan Xiang, Mingyang Li, Tianci Luo, Hanwen Wan, Ao Li, Linbo Zhai, Zhihao Zhan, Xiaodong Bai, Jiakun Cai, Peng Cao, Kangliang

Chen, Siang Chen, Yixiang Dai, Shuai Di, Yicheng Gong, Chenguang Gui, Yucheng Guo, Peng Hao, Qingrong He, Haoyang Huang, Kunrui Huang, Zhixuan Huang, Shibo Jin, Yixiang Jin, Anson Li, Dongjiang Li, Jiawei Li, Ruodai Li, Yihang Li, Yuzhen Li, Jiaming Liang, Fangsheng Liu, Jing Long, Mingxi Luo, Xing Pan, Hui Shen, Xiaomeng Tian, Daming Wang, Song Wang, Junwu Xiong, Hang Xu, Wanting Xu, Zhengcheng Yu, He Zhang, Jiyao Zhang, Lin Zhao, Chen Zhou, Nan Duan, Yuzheng Zhuang, and Liang Lin. Joyai-ra 0.1: A foundation model for robotic autonomy, 2026b. URL https://arxiv.org/abs/2604.20100.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model, 2025. URL https://arxiv.org/abs/2510.10274.

Ruijie Zheng, Dantong Niu, Yuqi Xie, Jing Wang, Mengda Xu, Yunfan Jiang, Fernando Castañeda, Fengyuan Hu, You Liang Tan, Letian Fu, et al. Egoscale: Scaling dexterous manipulation with diverse egocentric human data. *arXiv preprint arXiv:2602.16710*, 2026.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Appendix

A.1 Unified attention measurement

To ensure a controlled comparison in Fig. 1, we apply a unified attention-extraction protocol to all models. Attention is taken from the final action-prediction step, averaged over all action-token queries and attention heads, and aggregated over the middle third of the corresponding Transformer layers. We use the same observations, camera views, instructions, spatial projection, and visualization scale for every model, without model-specific layer or head selection. For bottleneck-based models, action-to-image attention is obtained by composing the action-to-bottleneck and bottleneck-to-image attention matrices; for dense models, it is read directly from action queries to image tokens.

We further repeat this analysis on additional held-out manipulation tasks using the same fixed protocol. Across tasks, Pelican-VLA consistently concentrates attention on the instructed target and manipulation-relevant regions, whereas the dense and no-BotToken controls exhibit more diffuse spatial attention. The corresponding quantitative alignment results follow the same trend as the controlled checkpoint analysis, indicating that the observed effect is not specific to a single scene or selected visualization.

A.2 Training objectives

Pelican-VLA 0.5 is trained with a multi-task objective that combines action generation, future-frame prediction, language-task alignment, and BotTokens regularization.

Flow-matching action loss. We use conditional flow matching Lipman et al. (2023); Black et al. (2026) to model the action chunk. Let $\mathbf{a} \in \mathbb{R}^{H \times d_a}$ be the target delta-action chunk, where H is the action horizon and d_a is the action dimension. We sample Gaussian noise $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ with the same shape as $\mathbf{a}$, and sample the flow time $\tau \sim \mathcal{U}(0, 1)$. We construct the interpolated noisy action and the target velocity as

$$\mathbf{x}_\tau = \tau\, \boldsymbol{\varepsilon} + (1 - \tau)\, \mathbf{a}, \qquad \mathbf{u} = \boldsymbol{\varepsilon} - \mathbf{a}. \tag{9}$$

Here $\mathbf{x}_\tau$ moves from the clean action $\mathbf{a}$ at $\tau = 0$ to Gaussian noise $\boldsymbol{\varepsilon}$ at $\tau = 1$, and $\mathbf{u}$ is the constant velocity along this linear path. The noisy action $\mathbf{x}_\tau$ is embedded into the suffix, and the action-token outputs are passed through a LayerNorm-MLP head v_θ. The action loss is

$$\mathcal{L}_{\text{action}} = \mathbb{E}_{\mathbf{a}, \boldsymbol{\varepsilon}, \tau} \left\| v_\theta(\mathbf{Z})_{\text{act}} - \mathbf{u} \right\|_2^2. \tag{10}$$

Here $v_\theta(\mathbf{Z})_{\text{act}} \in \mathbb{R}^{H \times d_a}$ denotes the decoded outputs corresponding only to the H action tokens.

Future-frame generation loss. The middle outputs are decoded by the BotTokens-gated generation head and supervised in the frozen Cosmos latent space. Given the future frame $\mathbf{I}_{t+15}$, we first encode it into a target Cosmos latent $\mathbf{c}_{t+15} = \mathrm{Cosmos}(\mathbf{I}_{t+15})$. We compute

$$\mathcal{L}_{\mathrm{gen}} = \left\| \mathrm{Dec}\big(\tilde{\mathbf{F}}_{\mathrm{gen}}\big) - \mathbf{c}_{t+15} \right\|_2^2. \tag{11}$$

Here $\mathrm{Dec}(\cdot)$ denotes the latent prediction head that maps the gated generation features to the Cosmos latent space.

BotTokens-language contrastive alignment. We further align the trajectory representation with the language instruction using a symmetric InfoNCE objective van den Oord et al. (2018); Radford et al. (2021); Liu et al. (2026a). For a batch size B, let $\mathbf{Z}_{\mathrm{traj}} \in \mathbb{R}^{B\times d_c}$ denote the trajectory embeddings, and let $\mathbf{Z}_{\mathrm{text}} \in \mathbb{R}^{B\times d_c}$ denote the text embeddings.

$$\mathbf{L} = \exp(s)\, \mathbf{Z}_{\mathrm{traj}} \mathbf{Z}_{\mathrm{text}}^{\top}. \tag{12}$$

The contrastive loss is

$$\mathcal{L}_{\mathrm{task}} = \tfrac{1}{2}\big(\mathrm{CE}(\mathbf{L}, \mathbf{y}) + \mathrm{CE}(\mathbf{L}^{\top}, \mathbf{y})\big), \qquad y_i = i,\ i = 1, \ldots, B. \tag{13}$$

Here $\mathbf{y}$ is the identity matching target within the mini-batch: the i-th trajectory is paired with the i-th instruction.

A.3 Cached inference

At inference time, the BotTokens provide a compact cacheable interface for action denoising. The prefix and middle tokens depend only on the current visual-language context, not on the flow time. Therefore, they are computed once and cached. Inference proceeds in three stages.

First, the prefix is processed by the language model with key-value caching enabled. Second, the middle tokens and BotTokens extend the cache. After this stage, we keep only the K BotTokens key-value pairs as the visual context for the denoising loop and discard the much larger prefix and middle cache. Third, we initialize the action chunk from Gaussian noise, $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and integrate the flow ODE backward from $\tau = 1$ to $\tau = 0$ using $N = 10$ Euler steps. At each step, only the suffix tokens are re-embedded and passed through the model using a copy of the BotTokens cache:

$$\mathbf{x}_{\tau+\Delta\tau} = \mathbf{x}_{\tau} + \Delta\tau \cdot v_{\theta}(\mathbf{Z})_{\mathrm{act}}, \qquad \Delta\tau = -\frac{1}{N}. \tag{14}$$

Here $v_{\theta}(\mathbf{Z})_{\mathrm{act}} \in \mathbb{R}^{H\times d_a}$ denotes the predicted velocity for the H action tokens. After the final step reaches $\tau = 0$, the predicted delta actions are de-normalized and added to the current proprioceptive state. This procedure makes the repeated denoising cost depend on the suffix length and the number of BotTokens, rather than on the full visual context length. As a result, the model can preserve a BotTokens-mediated perception-to-action interface while supporting efficient high-frequency control.

A.4 Implementation and pre-training recipe

Pelican-VLA 0.5 is pre-trained on a heterogeneous, cross-embodiment mixture of large-scale manipulation corpora, including AgiBot World Alpha AgiBot-World-Contributors et al. (2025), InternData-A1 Tian et al. (2025), the Galaxea Open-World Dataset Jiang et al. (2025a), and roughly 1000 hours of self-collected teleoperation Tienkung and UR data whose collection format is comparable to RoboMIND Wu et al. (2025). The pre-training mixture contains more than 6000 hours of manipulation data. All datasets are unified through the LeRobot data interface. Since the corpora cover different robot embodiments with different degrees of freedom, we pad all state and action vectors to a common 32-dimensional format.

We instantiate the backbone as the Qwen3-VL 4B and train it end-to-end in `bfloat16` with gradient checkpointing. The action chunk length is H=50 with delta-action targets, and the state and action vectors are padded to 32 dimensions for cross-embodiment training. Each observation comprises three camera views resized to 224×224, with image deltas of $[-15, 0, 15]$. Optimization uses AdamW with a peak learning rate of 5×10^{-5}, 2k warmup steps, and cosine decay to 5×10^{-6}, together with a per-GPU batch size of 6. The reasoning-BotTokens module uses K=32 BotTokens

with dropout 0.1 and a curriculum warmup of 10k steps, and the InfoNCE term is enabled with a task-embedding dimension of 256. The current model is pre-trained for only 0.4 epoch over this mixture, so it has effectively seen about 2400 hours of data; the results reported here are therefore obtained under a substantially under-trained regime.

A.5 Results on Libero benchmark.

Table 5: Benchmark results on LIBERO. Compared with recent VLA baselines, Pelican-VLA 0.5 achieves the best performance. Bold and underlined values denote the best and second-best results, respectively.

Methods	Spatial	Object	Goal	Long	Average
π_0 Black et al. (2026)	97.6	96.8	96.0	88.6	94.8
$\pi_{0.5}$ Black et al. (2025)	97.8	<u>98.6</u>	<u>96.8</u>	91.7	96.2
GR00T-N1 Bjorck et al. (2025)	94.4	97.6	93.0	90.6	93.9
X-VLA Zheng et al. (2025)	<u>98.2</u>	<u>98.6</u>	**97.8**	**97.6**	<u>98.1</u>
Pelican-VLA 0.5	**99.4**	**99.6**	**97.8**	<u>97.4</u>	**98.6**

A.6 Zero-shot setting.

To avoid inflating zero-shot performance through task-level leakage, we first audit the overlap between InternData-A1 and RoboTwin 2.0. We compute TF-IDF cosine similarity between task prompts from InternData-A1 and RoboTwin 2.0, manually re-examine all candidate pairs by reading the task descriptions and, when necessary, the corresponding demonstration videos, and exclude all tasks judged to be highly or moderately suspicious overlaps. The five tasks in Table 2 are referred to by their RoboTwin abbreviations; Table 6 gives the corresponding official descriptions.

Table 6: Task abbreviations used in Table 2 and their official RoboTwin descriptions.

Task	Description
shake bottle horizontally	Shake the bottle horizontally with the appropriate arm.
shake bottle	Shake the bottle with the appropriate arm.
click bell	Click the top center of the bell on the table.
open microwave	Use one arm to open the microwave.
click alarmclock	Click the center of the top-side button of the alarm clock on the table.

A.7 Attention dynamics over pre-training

To characterize how representation emerges during pre-training, we track attention through the Bot-Tokens across checkpoints of the full model, as shown in Fig.9. Using the action-to-image attention maps defined above, we retain pixels at or above the per-frame 90th percentile, yielding a binary region containing approximately the top 10% most-attended pixels. IoU is computed from the raw attention maps before visualization normalization, without smoothing or morphological post-processing. The evaluation masks are generated post hoc by using OWLv2 detections to initialize SAM 2.1 mask propagation; they are used exclusively for evaluation and are never provided during training. IoU is computed independently for each valid frame-view pair and then averaged within each checkpoint, with missing or empty masks excluded. The controlled analysis uses 30 fixed frames from one RoboTwin scene and ten checkpoints from 50k to 500k steps. Each checkpoint contains 57 valid target-object frame-view pairs and 30 manipulator pairs. We additionally compare against an area-matched random baseline that selects the same fraction of image pixels uniformly at random.

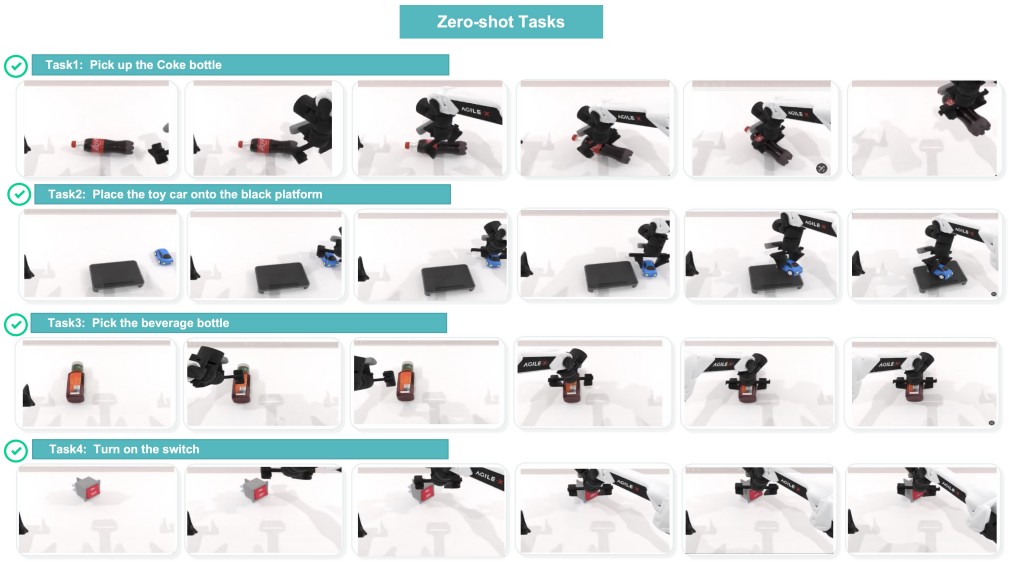


Figure 8: Zero-shot rollouts before task-specific fine-tuning.

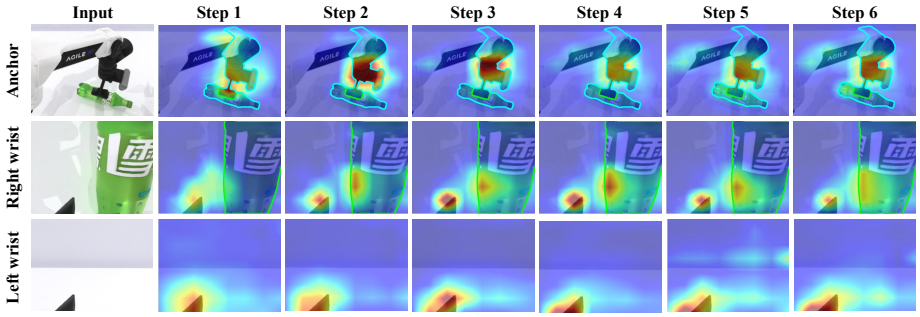


Figure 9: **The representation builds up over pre-training.** The BotTokens attention starts diffuse and migrates onto the manipulation-relevant object as training proceeds; the representation keeps rising after the action loss has saturated.

A.8 Zero-shot vs fine-tuned representations

This report shows that Pelican-VLA 0.5 already develops manipulation-centric attention before any task-specific adaptation. A natural question is whether fine-tuning creates a new representation pattern for the target task, or instead builds on the representation that is already present. To examine this, we compare the zero-shot model and its fine-tuned counterpart on the same held-out task, using attention visualizations, representation maps, and representation-similarity measures.

As shown in Fig. 10, fine-tuning changes the representation pattern only marginally. The zero-shot and fine-tuned models attend to highly similar object regions, and their BotTokens representations remain close under representation-similarity analysis. Their behavior, however, differs substantially. The zero-shot model often selects and approaches the correct object but fails at fine-grained control, whereas the fine-tuned model executes the task reliably.

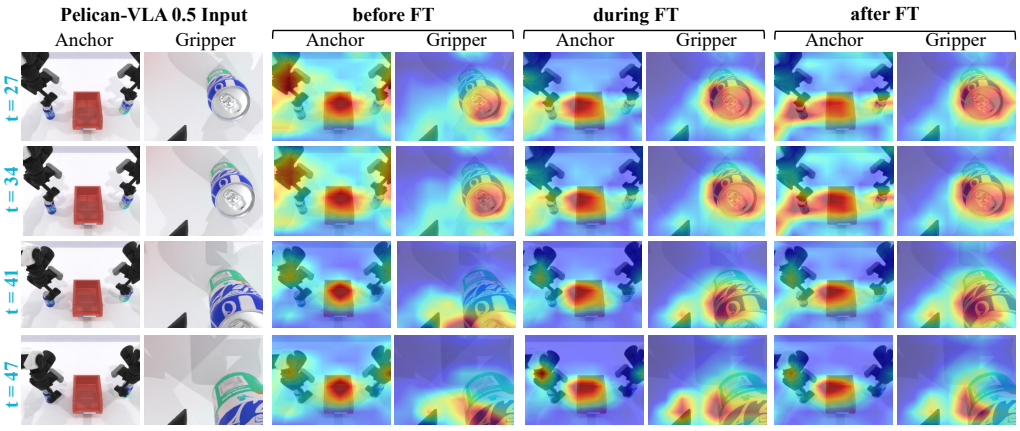


Figure 10: **Zero-shot and fine-tuned representations are highly similar.** Fine-tuning leaves the BotTokens representation and representation map largely unchanged, yet task success increases substantially. This suggests that the grounded representation is already present before adaptation, while fine-tuning mainly improves the mapping from representation to action.

A.9 REAL-WORLD ROBOT EVALUATION

We evaluate nine real-world manipulation tasks across Tien Kung, AgileX, and UR5e. Figure 11 shows the task scenes, and Table 7 lists the corresponding language instructions. The tasks include object placement, assembly, multi-step organization, cleaning, and bimanual coordination.

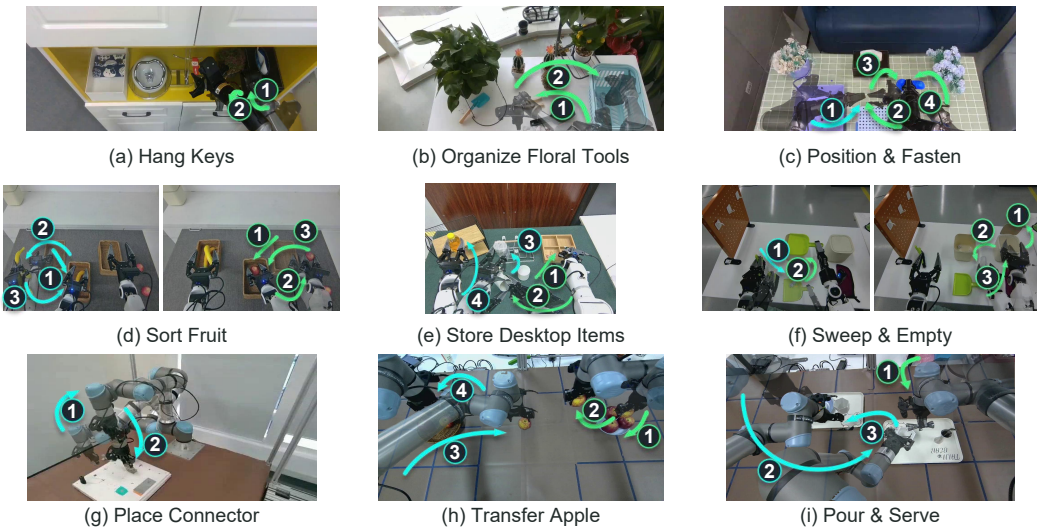


Figure 11: **Nine real-world manipulation tasks.** Panels (a)–(i) show the task scenes. Arrows and robot-arm overlays indicate task goals, not executed trajectories.

Table 7: Language instructions for the nine real-world tasks.

Task	Language instruction
(a) Hang Keys	Pick up the keyring and hang it on the key rack.
(b) Organize Floral Tools	Pick up the scissors and put them into the floral tool container.
(c) Position & Fasten	Position the parts and fasten the screws to assemble the number "1".
(d) Sort Fruit	Put the bananas in the left basket and the apples in the right basket, then return both baskets to their starting positions.
(e) Store Desktop Items	Put the snacks into the organizer, place the bottle on the shelf, and hang the cups on the rack.
(f) Sweep & Empty	Sweep the tabletop refuse into the dustpan, then empty the dustpan into the trash bin.
(g) Place Connector	Pick up the waterproof male connector and place it at the designated position.
(h) Transfer Apple	Pick up the apple from the plate and put it in the basket.
(i) Pour & Serve	Pour the wine with the right arm and place the cup on the tray with the left arm.

For each of the nine real-world tasks, we collect 150 teleoperated demonstrations in the original task setting and fine-tune Pelican-VLA 0.5 and $\pi_{0.5}$ on the same per-task dataset; these trajectories are used only for task adaptation. Each task is then evaluated with separate on-robot rollouts in the original setting and in a perturbed setting with changes in object placement, background color, and illumination, while the task instruction and success criterion remain unchanged. Both methods are evaluated from the same initial configurations within each setting over 25 trials per method, task, and setting, and a trial is successful only when all steps in the instruction are completed. We report full-task success rates and their unweighted average across the nine tasks; changes in average success rate are computed before rounding.