TwinBrainVLA : Unleashing the Potential of Generalist VLMs for Embodied Tasks via Asymmetric Mixture-of-Transformers

Anonymous ACL submission

Abstract

Vision-Language-Action (VLA) models are built on the premise that pre-trained Vision-Language Models (VLMs) can supply general semantic understanding ability for embodied tasks. However, we find that robotic fine-tuning can severely erode this capability: VQA benchmarks and layer-wise CKA analysis reveal rapid representational drift, and co-training with general vision-language data does not prevent the collapse. We propose **TwinBrainVLA**, an asymmetric dual-stream VLA that keeps a pre-trained "Left Brain" strictly frozen as a semantic generalist while training a "Right Brain" as an embodied control specialist. Its **Asymmetric Mixture-of-Transformers (AsyMoT)** mechanism lets the Right Brain query frozen Left-Brain key-value representations at every transformer layer, integrating intact semantic knowledge without corrupting it. Across SimplerEnv, RoboCasa, LIBERO, and real-robot experiments, TwinBrainVLA outperforms strong baselines and large-scale embodied models, achieving a **14.3%** improvement on SimplerEnv and a **2.7**× gain in physical-robot instruction generalization.

1 Introduction

Vision-Language-Action (VLA) models graft robotic control heads onto pre-trained Vision-Language Models (VLMs), aiming to reuse their semantic reasoning and open-world generalization for embodied control (Kim et al., 2024; Intelligence et al., 2025; NVIDIA et al., 2025). This direction is attractive for both robotics and language research: a VLA is expected to understand instructions, resolve visual references, recognize objects and attributes, and then translate that language-grounded understanding into continuous actions. In principle, a stronger VLM backbone should therefore bring stronger instruction following and semantic generalization to robots. The appeal is not merely parameter reuse: VLMs contain hard-won priors for open-vocabulary recognition, attribute grounding, spatial-language reasoning, and instruction following that are difficult to reconstruct from robot trajectories alone.

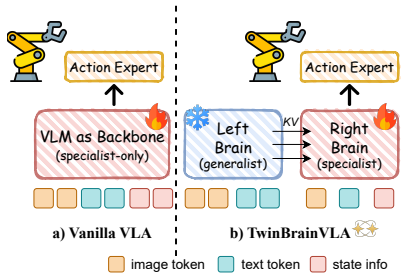


Figure 1: Architectural comparison between Vanilla VLA and TwinBrainVLA.

Yet this promise is fragile. Robotic fine-tuning optimizes the same backbone for dense low-level action prediction, and this objective can overwrite the high-level visual and linguistic features that made the VLM useful in the first place (Hancock et al., 2025; Zhang et al., 2026a; Zhou et al., 2025; Fang et al., 2025; Zhang et al., 2026b). This creates a *semantic tax* of action tuning: the model gains motor competence by spending the very multimodal knowledge that motivated VLA initialization. The resulting model may still imitate demonstrations, but it no longer behaves like a VLM-grounded embodied agent: language becomes a narrow command token, visual semantics become task-local cues, and open-world understanding is sacrificed for short-horizon control.

This failure mode is especially problematic for tasks that require generalization beyond the exact demonstrations. A robot asked to manipulate a new object color, follow a paraphrased instruction, or infer the relevant object from language must retain the semantic abstractions learned during VLM pre-training. If VLA fine-tuning collapses these

abstractions, scaling the backbone or adding more action data may improve imitation while weakening the very linguistic and perceptual capabilities that motivated using a VLM. Existing countermeasures, such as co-training with visual-dialogue data, still update the same parameters with conflicting objectives. They can slow forgetting but do not remove the underlying interference: a single representational stream is still asked to serve both language-grounded semantics and motor control.

We argue that a generalizable VLA should separate semantic preservation from motor specialization. **TwinBrainVLA** implements this principle with an asymmetric dual-stream architecture: a strictly frozen "Left Brain" preserves pre-trained VLM knowledge, while a trainable "Right Brain" learns proprioceptive control. The two streams communicate through Asymmetric Mixture-of-Transformers (AsyMoT), which lets the Right Brain query frozen Left-Brain semantic features at every transformer layer without updating the Left Brain. This design changes the role of the VLM backbone from a single set of parameters pulled between incompatible objectives into a persistent semantic resource that the control policy can consult during action generation.

The asymmetry is important. If both streams are trainable, the semantic stream can still drift toward the robotic objective; if the frozen stream is isolated, the policy cannot exploit its preserved knowledge. TwinBrainVLA therefore combines two constraints that are usually treated separately: strict protection of one VLM pathway and dense layer-wise access from the trainable control pathway. This gives the robot a stable semantic reference while still allowing the action policy to specialize aggressively on demonstrations.

Our contributions are summarized as follows:

- We diagnose catastrophic forgetting in VLA fine-tuning with VQA benchmarks and layer-wise CKA, showing that co-training does not prevent representational collapse.
- We propose **TwinBrainVLA**, an asymmetric dual-stream VLA that decouples semantic preservation from motor learning without extra data or supervision.
- We introduce **AsyMoT**, a layer-wise unidirectional joint-attention mechanism that fuses frozen semantic KV states with trainable control representations.
- Experiments on SimplerEnv, RoboCasa, LIBERO, and a real Franka robot show consistent gains, including a **14.3%** improvement on SimplerEnv, **2.7**$\times$ higher physical-robot instruction generalization, and a 15.1% iso-parameter gain over an 8B monolithic VLA.

2 Motivation: General Capabilities Erode

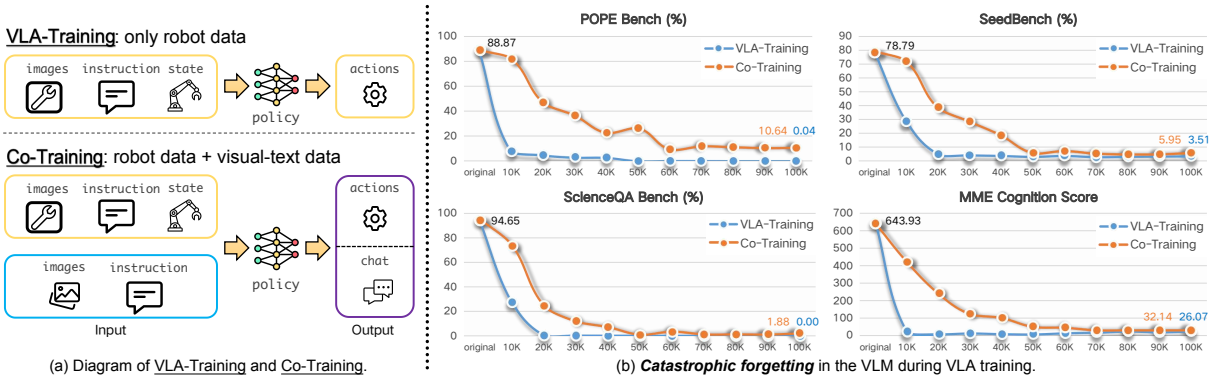
2.1 The Specialization-Generalization Conflict

Standard VLAs fine-tune pre-trained VLMs on robotic demonstrations ($\mathcal{D}_{robot}$), creating a direct conflict between action specialization and semantic generalization. From a gradient-conflict view (Lopez-Paz and Ranzato, 2017; Yu et al., 2020), dense action-prediction signals can push the backbone away from the representations needed for visual-language understanding. We interpret this as a *representational bottleneck*: a single VLM stream is forced to encode both language-grounded semantics and proprioceptive control features. The conflict is especially sharp because VLM pre-training emphasizes abstract semantic alignment, whereas robotic control requires spatially precise, task-specific, and embodiment-grounded representations.

2.2 Empirical Analysis: The Severity of Catastrophic Forgetting

We quantify this degradation with Qwen2.5-VL-3B-Instruct and Qwen3-VL-4B-Instruct under two training paradigms: **VLA Training**, which uses only robotic trajectory data, and **Co-Training**, which mixes robotic and visual-QA data at a 1:1 ratio following recent mitigation strategies (Mazzaglia et al., 2025; Zhou et al., 2025). As shown in Figure 2b, standard VLA training collapses general visual understanding: the POPE score of Qwen3-VL drops from 88.87% to 0.04%, and the model often loses the ability to produce coherent linguistic responses. Co-training slows but does not prevent this failure, suggesting that data mixing alone is not a structural solution.

We further examine internal representation drift with Centered Kernel Alignment (CKA), comparing layer-wise activations of each fine-tuned

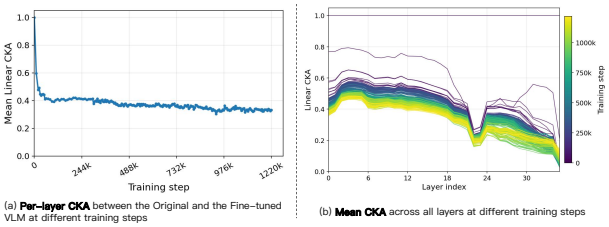


(a) Diagram of VLA-Training and Co-Training.

(b) ***Catastrophic forgetting*** in the VLM during VLA training.

Figure 2: **Empirical Evidence of Catastrophic Forgetting.** **(a)** Overview of two prevailing training paradigms: standard VLA-Training (utilizing robot-only data) and Co-Training (mixing robot data with general vision-language data). **(b)** Evaluation of the VLM's general visual understanding capabilities across four benchmarks (POPE (Li et al., 2023b), SEED-Bench (Li et al., 2023a), ScienceQA (Lu et al., 2022), and MME (Fu et al., 2025)) during the VLA fine-tuning process. Both strategies exhibit a precipitous decline in performance as training steps increase, demonstrating that co-training with general data is insufficient to prevent the collapse of the VLM's pre-trained semantic reasoning when optimizing for robotic control.

checkpoint against the original VLM. Figure 3 shows that CKA drops from near-perfect alignment to below 0.3 in deeper layers after only several thousand steps. Thus, VLA fine-tuning does not merely change output format; it restructures the semantic feature space itself. Appendix C provides the CKA background, feature extraction protocol, and layer-wise analysis details.



(a) **Per-layer CKA** between the Original and the Fine-tuned VLM at different training steps

(b) **Mean CKA** across all layers at different training steps

Figure 3: **Mean and Per-layer CKA between fine-tuned VLM and original checkpoint.** As VLA training progresses, both mean (left) and per-layer (right) CKA values decrease sharply, indicating severe representational drift. Lower CKA signifies greater divergence from the original semantic representations.

These results motivate a structural question: ***how can a VLA learn robotic control while still using uncorrupted VLM knowledge?***

3 Method: TwinBrainVLA

TwinBrainVLA keeps pre-trained semantic knowledge intact and makes it accessible to a trainable control policy. Its asymmetric dual-stream design separates a frozen semantic pathway from a trainable sensorimotor pathway, then couples them with layer-wise joint attention.

3.1 Asymmetric Dual-VLM Backbone

The backbone contains two VLM pathways initialized from the same pre-trained checkpoint: the "Left Brain" ($\mathcal{M}_L$) and the "Right Brain" ($\mathcal{M}_R$).

Left Brain. The Left Brain is kept strictly frozen and processes the visual observation I and instruction T into semantic representations:

$$\mathbf{H}_L^0 = [\, \mathcal{V}(I) \,;\, \mathcal{T}(T) \,] \tag{1}$$

where $\mathcal{V}$ and $\mathcal{T}$ denote the vision encoder and text tokenizer. Freezing this stream turns it into a stable source of open-world VLM knowledge.

Right Brain. The Right Brain is trainable and receives the same visual and textual inputs plus a lightweight state encoding $\phi(s)$ for proprioception:

$$\mathbf{H}_R^0 = [\, \mathcal{V}(I) \,;\, \mathcal{T}(T) \,;\, \phi(s) \,] \tag{2}$$

Asymmetric Mixture-of-Transformers (AsyMoT). Inspired by Mixture-of-Transformers (Liang et al., 2025), AsyMoT creates a unidirectional bridge from the frozen Left Brain to the trainable Right Brain.

Let $\mathbf{H}_L^l$ and $\mathbf{H}_R^l$ denote the hidden state vectors of the Left and Right Brains at layer l, respectively.

The Left Brain performs ordinary frozen self-attention:

$$\mathbf{H}_L^{l+1} = \mathrm{Attn}(Q_L^l, K_L^l, V_L^l) + \mathrm{FFN}(\mathbf{H}_L^l), \tag{3}$$

where Q, K, V are derived from $\mathbf{H}_L^l$ using frozen projections.

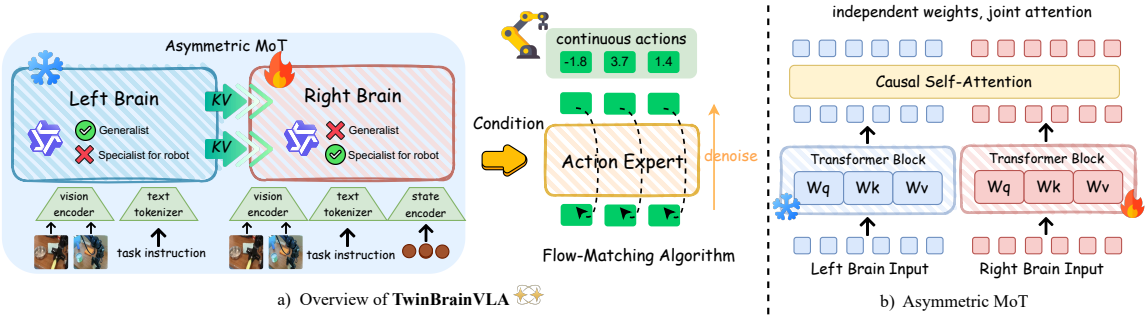


Figure 4: **The framework of TwinBrainVLA . (a) Overall Architecture.** The model features an Asymmetric Mixture-of-Transformers design composed of two distinct pathways: a frozen "Left Brain" (Generalist) for semantic reasoning and a trainable "Right Brain" (Specialist) for embodied motor control. The Right Brain fuses visual, textual, and proprioceptive state inputs to provide conditioning for the Action Expert, which utilizes a Flow-Matching algorithm to denoise continuous robotic actions. **(b) Asymmetric MoT Mechanism (AsyMoT).** Through causal self-attention, the trainable Right Brain attends to the frozen Key-Value (KV) pairs of the Left Brain, enabling the transfer of general semantic knowledge to the robotic control policy without catastrophic forgetting.

The Right Brain computes trainable queries and attends jointly to its own KV states and stop-gradient Left-Brain KV states:

$$K_{\text{joint}} = [\text{sg}(K_L^l)\ ;\ K_R^l], \tag{4}$$

$$V_{\text{joint}} = [\text{sg}(V_L^l)\ ;\ V_R^l], \tag{5}$$

$$\mathbf{H}_R^{l+1} = \text{Softmax}\left(\frac{Q_R^l(K_{\text{joint}})^T}{\sqrt{d_k}}\right)V_{\text{joint}} + \text{FFN}(\mathbf{H}_R^l), \tag{6}$$

where $[\,;\,]$ concatenates along the sequence dimension and $\text{sg}(\cdot)$ blocks gradients into the Left Brain.

AsyMoT vs. Cross-Attention. Unlike standard cross-attention, AsyMoT does not force the Right Brain to attend exclusively to the Left Brain. KV concatenation lets each layer select between self-context and frozen semantic context within one attention operation.

3.2 Flow-Matching Action Expert

Following recent VLAs (NVIDIA et al., 2025; Intelligence et al., 2025), we use a flow-matching action expert implemented as a Diffusion Transformer (Peebles and Xie, 2023). Conditioned on Right-Brain representations $\mathbf{H}_R$, it predicts continuous action trajectories with the standard vector-field regression loss:

$$\mathcal{L}_{\text{action}}(\psi) = \mathbb{E}_{t,\mathbf{a}_0,\mathbf{a}_1}\Big[\big\|v_\psi(\mathbf{a}_t, t, \mathbf{H}_R) - (\mathbf{a}_1 - \mathbf{a}_0)\big\|_2^2\Big], \tag{7}$$

where v_ψ denotes the DiT network, $\mathbf{a}_0$ is the Gaussian noise, and $\mathbf{a}_1$ represents the ground-truth action.

3.3 Training Strategy

Optimization Objective. We train only the Right Brain, Action Expert, and State Encoder with the robotic action objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{action}}(\theta_R, \psi, \phi; \mathcal{D}_{\text{robot}}), \tag{8}$$

where $\mathcal{D}_{\text{robot}}$ is the demonstration dataset and $\{\theta_R, \psi, \phi\}$ are the trainable parameters. The Left Brain is never updated, so action learning cannot corrupt its linguistic or visual priors.

4 Experiment

We evaluate TwinBrainVLA on SimplerEnv (Li et al., 2024c), RoboCasa (Nasiriany et al., 2024), LIBERO (Liu et al., 2023a), and a physical Franka robot. Training is based on the starVLA framework (starVLA, 2025) and runs on 8 × NVIDIA H100 GPUs with its default protocols unless otherwise stated.

4.1 Experiments on SimplerEnv (OOD)

Setup and Results. We instantiate TwinBrainVLA with Qwen2.5-VL-3B and Qwen3-VL-4B backbones and train on the `Bridge-V2` and `Fractal` OXE subsets. Evaluation uses four OOD SimplerEnv tasks with 480 trials per task. As shown in Table 1, TwinBrainVLA reaches **71.4%** average success with Qwen3-VL-4B, outperforming the strongest baseline (Isaac-GR00T-N1.6: 57.1%) by **14.3**

Table 1: **Results of evaluating the VLA models with the WidowX robot in the SimplerEnv simulation environment**. We highlight the best results in **bold** and the second-best results with underline. Here, **Vanilla VLA** refers to a VLA constructed with a single VLM backbone.

Method	Put Spoon on Towel	Put Carrot on Plate	Stack Green Block on Yellow Block	Put Eggplant in Yellow Basket	Average
RT-1-X (ONeill et al., 2024)	0.0	4.2	0.0	0.0	1.1
Octo-Base (Team et al., 2024)	15.8	12.5	0.0	41.7	17.5
Octo-Small (Team et al., 2024)	41.7	8.2	0.0	56.7	26.7
OpenVLA (Kim et al., 2024)	4.2	0.0	0.0	12.5	4.2
RoboVLM (Li et al., 2024b)	50.0	37.5	0.0	83.3	42.7
TraceVLA (Zheng et al., 2025b)	12.5	16.6	16.6	65.0	27.7
SpatialVLA (Qu et al., 2025)	20.8	20.8	25.0	70.8	34.4
ThinkAct (Huang et al., 2025)	58.3	37.5	8.7	70.8	43.8
CogACT (Li et al., 2024a)	71.7	50.8	15.0	67.5	51.3
VideoVLA (Shen et al., 2025)	75.0	20.8	45.8	70.8	53.1
π_0 (Black et al., 2024)	29.1	0.0	16.6	62.5	27.1
$\pi_{0.5}$ (Intelligence et al., 2025)	49.3	64.7	44.7	69.7	57.1
Isaac-GR00T-N1.6-Bridge (Team et al., 2025b)	64.5	65.5	5.5	93.0	57.1
Vanilla VLA + Qwen2.5-VL-3B-Instruct	74.9	44.7	9.3	81.2	52.5
Vanilla VLA + Qwen2.5-VL-7B-Instruct	68.7	35.4	16.7	62.5	44.8
Vanilla VLA + Qwen3-VL-4B-Instruct	87.5	50.0	29.2	54.2	55.2
Vanilla VLA + Qwen3-VL-8B-Instruct	68.7	38.5	30.2	87.9	56.3
ours					
TwinBrainVLA + Qwen2.5-VL-3B-Instruct	83.3	54.2	31.5	89.6	64.7
TwinBrainVLA + Qwen3-VL-4B-Instruct	87.5	58.3	45.8	93.8	**71.4**

percentage points. The gain is not merely from parameter count: TwinBrainVLA (Qwen3-VL-4B, ∼8B total) exceeds a monolithic Qwen3-VL-8B VLA (56.3%) by 15.1 percentage points, while vanilla scaling from 3B to 7B degrades performance, consistent with larger backbones being more exposed to forgetting (Wang et al., 2025). Appendix B provides the full training configuration and implementation details.

4.2 Experiments on RoboCasa

Setup. We train on the Humanoid Robot Tabletop Manipulation subset from the PhysicalAI-Robotics-GR00T-X-Embodiment-Sim dataset (NVIDIA et al., 2025), using the same initialization and training protocol as Sec. 4.1.

Evaluation and Results. RoboCasa GR1 Tabletop contains 24 manipulation tasks with articulated objects and varied geometries; we average over 50 trials per task. Table 2 shows that TwinBrainVLA achieves the best average success rate, with Qwen3-VL reaching **54.6%**. This surpasses Isaac-GR00T-N1.6 by 7.0 points, QwenGR00T by 6.8 points, and QwenPI by 10.7 points.

Table 2: **RoboCasa GR1 Tabletop results.** Isaac-GR00T results are from the official repository (Team et al., 2025b); the first four baselines are from starVLA (starVLA, 2025). Appendix D summarizes the Qwen-based baseline architectures, and Table 9 reports all 24 task-level success rates.

Model	QwenFAST +Qwen3VL	QwenGR00T +Qwen3VL	QwenPI +Qwen3VL	QwenOFT +Qwen3VL
Average	39.0	47.8	43.9	48.8
Model	Isaac-GR00T N1.5	Isaac-GR00T N1.6	**TwinBrainVLA** + Qwen2.5VL	**TwinBrainVLA** + Qwen3VL
Average	48.2	47.6	53.5	**54.6**

4.3 Experiments on LIBERO Benchmark

LIBERO is largely saturated by current VLAs, but it tests whether the proposed architecture preserves in-domain competence. We train one model on the combined data from all four LIBERO suites and evaluate it across the same suites. TwinBrainVLA attains a 97.9% average success rate, matching or slightly exceeding strong baselines without massive pre-training. Appendix G describes the unified training data, training configuration, and evaluation protocol.

4.4 Real-robot Experiments

We evaluate real-world robustness on a Franka Research 3 (7-DOF), targeting two axes tied to VLM

Table 3: **LIBERO benchmark results.**

Method	Spatial	Object	Goal	Long	Avg
OpenVLA	84.7	88.4	79.2	53.7	76.5
π_0	96.8	98.8	95.8	85.2	94.1
$\pi_{0.5}$	98.8	98.2	98.0	92.4	96.9
Vanilla VLA (4B)	98.8	99.0	98.0	94.4	97.6
Vanilla VLA (8B)	99.2	99.0	97.6	93.4	97.3
TwinBrainVLA (4B $\times$ 2)	99.2	99.0	98.0	95.4	**97.9**

capabilities: *perceptual generalization* to unseen visual attributes and *semantic generalization* to diverse instructions.

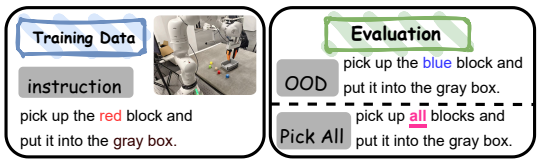


Figure 5: Real-robot Experiment Environment.

Setup and Results. After pre-training on 20 OXE subsets, we fine-tune with 300 teleoperated Franka trajectories for color-conditioned pick-and-place instructions over red, green, and blue blocks. Evaluation includes in-domain trials, altered object attributes, 50 lexically diverse paraphrases executed twice, and a long-horizon Pick-All task. Table 4 shows that TwinBrainVLA matches $\pi_{0.5}$ on in-domain performance while improving instruction generalization by **2.7**$\times$ (41/100 vs. 15/100) and task generalization from 40/100 to 52/100, supporting the link between preserved VLM semantics and real-world robustness. Appendix A details the OXE subset selection, real-robot fine-tuning setup, and evaluation protocol.

Table 4: **Real-robot results.** Pick-and-place success rates on the Franka Research 3 across four evaluation protocols.

Method	Success Rate			
	In-Domain	Task Gen.	Instr. Gen.	Pick-All
OpenVLA (Kim et al., 2024)	7/30	0/100	0/100	0/30
π_0 (Black et al., 2024)	23/30	27/100	9/100	0/30
$\pi_{0.5}$ (Intelligence et al., 2025)	**28/30**	40/100	15/100	2/30
Vanilla VLA (4B)	23/30	25/100	4/100	0/30
Vanilla VLA (8B)	22/30	28/100	5/100	0/30
TwinBrainVLA (4B $\times$ 2)	**28/30**	**52/100**	**41/100**	**3/30**

4.5 Ablation Experiments

Freezing strategy. Table 5 compares three Left-Brain settings: no freezing, random frozen initialization, and our frozen pre-trained Left Brain. Freezing pre-trained semantics improves average success by 15.7 points on the 3B backbone and 12.6 points on the 4B backbone over the no-freezing variant. The random-frozen control shows that the gain is not a generic regularization effect: removing pre-trained knowledge substantially reduces performance, especially for Qwen3-VL-4B.

Table 5: **Ablation on Freezing Strategy.** Success rates (%) on SimplerEnv. Three configurations of the dual-stream architecture are compared to disentangle the contribution of *pre-trained knowledge content* from the *structural regularization effect* of freezing.

Method	Put Spoon on Towel	Put Carrot on Plate	Stack Block	Eggplant in Basket	Avg
Qwen2.5-VL-3B-Instruct					
(1) No Freezing	64.6	45.8	20.8	64.6	49.0
(2) Random Frozen Left Brain	66.7	49.0	17.7	55.6	47.3
(3) **TwinBrainVLA (Ours)**	83.3	54.2	31.5	89.6	**64.7**
Qwen3-VL-4B-Instruct					
(1) No Freezing	85.4	50.0	29.1	70.8	58.8
(2) Random Frozen Left Brain	79.2	41.7	14.6	31.3	41.7
(3) **TwinBrainVLA (Ours)**	87.5	58.3	45.8	93.8	**71.4**

AsyMoT density. Table 6 sparsifies joint attention with intervals $k = \{0, 1, 2, 4, 8\}$. Dense or moderately sparse interaction works best, whereas large intervals degrade performance. Removing the Left Brain entirely reduces the model to Vanilla VLA and causes a 16.2-point drop for Qwen3-VL-4B (71.4% vs. 55.2%).

5 Analysis and Discussion

5.1 From Forgetting Mitigation to Capability Utilization

TwinBrainVLA reframes forgetting mitigation as capability utilization: the frozen Left Brain is not only protected from action fine-tuning, but also made available to the Right Brain through AsyMoT. The attention visualization in Figure 6 provides qualitative evidence that the Right Brain queries Left-Brain semantic tokens at runtime.

This distinction matters because simply preventing weight updates is not enough: the preserved knowledge must also be accessible to the policy at the moment of decision making. AsyMoT provides this access densely, allowing the Right Brain to select between its own embodied state and the frozen semantic context at each layer. The freezing ablation supports this interpretation. A random frozen Left Brain preserves the architecture but removes semantic content, and it performs substantially worse than the pre-trained frozen Left Brain. Thus, the gain comes from the content of

Table 6: **Ablation on AsyMoT Interaction Frequency.** We investigate the impact of sparsifying the joint attention connections by varying the interaction interval k on SimplerEnv benchmark.

Interval (k)	**0** (ours)	**1**	**2**	**4**	**8**
Qwen2.5-VL-3B-Instruct based	64.7	**66.7**	57.4	56.5	55.9
Qwen3-VL-4B-Instruct based	71.4	70.1	**73.5**	62.5	61.9

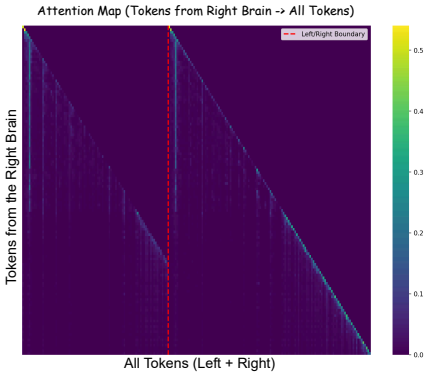


Figure 6: **Runtime attention visualization.** The Right Brain attends to semantic tokens from the frozen Left Brain through AsyMoT.

pre-trained VLM representations rather than from parameter count or regularization alone.

5.2 When General Semantics Help Most

The experimental pattern is consistent with the paper's central hypothesis: general VLM semantics matter most when evaluation requires language or perceptual generalization rather than only in-domain motor replay. On LIBERO, where several recent VLA systems already approach or exceed 95% success and the test tasks remain close to the training distribution, TwinBrainVLA yields only a modest gain.

The benefit becomes clearer on SimplerEnv and the real robot. SimplerEnv contains OOD object and scene configurations, where the policy must ground instructions in visual semantics that are not identical to the demonstrations. The real-robot instruction-generalization protocol is even more diagnostic: the physical task is the same, but the language distribution changes through lexically diverse paraphrases. Vanilla VLAs can overfit to the training command, whereas TwinBrainVLA can still draw on a frozen language-vision representation that understands the paraphrases. This explains why the largest physical-robot gain appears on instruction generalization rather than on the in-domain setting.

5.3 Inference Cost and Deployment

The main practical cost of TwinBrainVLA is that it runs two VLM streams. However, the frozen Left Brain also creates deployment opportunities that are not available in a fully trainable monolithic VLA. First, both brains process the same visual observation, so the vision encoder can be shared. Second, the Left Brain's KV states depend only on the observation and instruction and can be cached across multiple action denoising steps. Third, because the Left Brain is never updated at deployment time, it is a natural target for quantization and pipeline execution. These optimizations reduce the overhead of preserving semantic knowledge while keeping the architectural separation intact.

Table 7: **Deployment cost.** Runtime for 2,880 inference calls.

Model	**Total (s)**	**Per-step (ms)**	**Overhead**
Vanilla (4B)	320.7	111.4	—
Vanilla (8B)	334.8	116.3	—
TwinBrainVLA (2×4B)	402.3	139.7	**+25%**

Table 7 reports the optimized inference cost on SimplerEnv. TwinBrainVLA incurs a 25% wall-clock overhead relative to the 4B Vanilla VLA, despite using two 4B streams, and remains within a practical range for closed-loop manipulation. This suggests a useful tradeoff: preserving semantic generalization does not require paying the full cost of naively doubling the backbone. Appendix I provides implementation details for the shared vision encoder, KV caching, quantization, and pipeline parallelism.

5.4 Twin-to-One Distillation

The dual-stream model is primarily designed to test whether intact VLM semantics can improve embodied control, but it also suggests a practical compression path. After training, the Right Brain has interacted with the frozen Left Brain at every layer through AsyMoT. Its hidden states therefore contain not only task-specific control features, but also semantic information repeatedly queried from the frozen VLM stream. We can use these enriched Right-Brain representations as supervision for a compact single-stream VLA, a procedure we refer to as *Twin-to-One* distillation.

This distillation view is useful for two reasons. First, it provides a route to deployment settings

where even the optimized dual-stream cost in Table 7 is undesirable: the dual-stream teacher can be used during training, while the distilled student keeps the runtime footprint of a standard VLA. Second, it gives an additional diagnostic for AsyMoT. If the Right Brain were only using the Left Brain as an external lookup table at inference time, a single-stream student would have little useful signal to inherit. By contrast, a student trained on the Right Brain's enriched states can inherit semantic structure only if repeated layer-wise interaction has reshaped the Right Brain's internal representation. Twin-to-One distillation therefore turns deployment compression into a probe of whether semantic knowledge has actually been absorbed by the control stream, reinforcing the same conclusion as the freezing and attention analyses. Appendix H provides the distillation objective, setup, and SimplerEnv results.

6 Related Work

Vision-language models and embodied semantics. VLMs (Liu et al., 2023b; Bai et al., 2025b,a) integrate visual encoders with LLMs to support visual question answering, instruction following, and semantic grounding. These capabilities are directly relevant to embodied agents: a robot must parse linguistic goals, bind object names and attributes to visual observations, and maintain enough semantic abstraction to handle unseen instructions. However, general-purpose VLMs still struggle with fine-grained spatial perception and embodied reasoning (Xiao et al., 2026; Dang et al., 2025; Zhang et al., 2025; Chen et al., 2025; Yang et al., 2025b). Recent work therefore improves spatial and embodied reasoning through post-training or specialized embodied foundation models (Yang et al., 2025a; Tan et al., 2025; Zhou et al., 2026; Zhu et al., 2025; Li et al., 2025; Team et al., 2025a; Hao et al., 2025; Lin et al., 2025). TwinBrainVLA is complementary to these efforts: rather than only improving the pre-trained model, it asks how such capabilities can survive and be used after robotic fine-tuning begins.

Vision-Language-Action models. VLA models fine-tune pre-trained VLMs to map visual-language instructions to actions (Pertsch et al., 2025; Zheng et al., 2025a; Kim et al., 2024). Recent systems differ in action representations, data scale, and policy heads, ranging from action tokenization to flow-matching control (Black et al., 2024; Intelligence et al., 2025; NVIDIA et al., 2025). Despite these differences, most share a monolithic adaptation pattern: the backbone that contains general VLM knowledge is also optimized for robot actions. This creates the specialization-generalization conflict studied in this paper.

Forgetting mitigation in VLAs. Several works mitigate forgetting by mixing dialogue or VQA data during training (Zhou et al., 2025; Hancock et al., 2025), adding reasoning traces (Huang et al., 2025), or blocking gradient flow during large-scale pre-training (Driess et al., 2025). These approaches reduce interference but either require additional data/annotations or depend on a particular training stage. TwinBrainVLA instead makes semantic preservation an architectural invariant: the Left Brain remains frozen regardless of downstream training, and the Right Brain accesses it through AsyMoT. This separates the question of preserving VLM knowledge from the question of learning a task-specific controller.

Multi-stream transformer interaction. Our AsyMoT mechanism is inspired by Mixture-of-Transformers (Liang et al., 2025), which connects transformer streams through joint attention. The key difference is the asymmetry required by VLA training: only the control stream is updated, while the semantic stream remains immutable. Compared with standard cross-attention, AsyMoT concatenates self and semantic KV states in one attention operation, so the Right Brain can preserve its embodied context while selectively consulting the frozen Left Brain at every layer.

7 Conclusion

We identified a central paradox in current VLA training: robotic fine-tuning can erode the VLM capabilities it aims to exploit. **TwinBrainVLA** addresses this structurally by coupling a frozen semantic generalist with a trainable embodied specialist through AsyMoT. Across SimplerEnv, RoboCasa, LIBERO, and real-robot settings, preserving general VLM capabilities yields stronger manipulation performance and substantially better instruction generalization. We hope this encourages future VLA architectures to treat semantic preservation as a first-class design criterion.

Limitations

TwinBrainVLA improves the use of pre-trained VLM semantics in VLA training, but several limitations remain. First, the dual-stream design increases parameter count and inference cost relative to a single-backbone VLA, although the frozen stream allows caching, quantization, and pipeline execution. Second, our real-robot study focuses on one Franka manipulation setup; broader validation across more embodiments, sensors, and long-horizon tasks is needed. Finally, our forgetting analysis uses representative VLM backbones and VQA benchmarks, which may not cover every form of semantic or spatial reasoning relevant to embodied agents.

Ethics Statement

This work studies VLA training for robotic manipulation. The main ethical risks come from physical deployment rather than from the benchmark evaluation itself: robots controlled by learned policies may fail under distribution shift, collide with surrounding objects, or execute instructions in unintended ways. Our real-robot experiments are conducted in a constrained tabletop setting with human supervision, and the method is not intended for unsupervised deployment in safety-critical environments. The real-robot demonstrations contain task-level teleoperation trajectories rather than personally identifying user data. If the models or data are released, operator-identifying information should be removed, and downstream users should evaluate safety, bias, and failure modes for their own embodiments and deployment contexts.

References

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, and 45 others. 2025a. Qwen3-vl technical report.

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, and 8 others. 2025b. Qwen2.5-vl technical report.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, and 1 others. 2024. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*.

Shiqi Chen, Tongyao Zhu, Ruochen Zhou, Jinghan Zhang, Siyang Gao, Juan Carlos Niebles, Mor Geva, Junxian He, Jiajun Wu, and Manling Li. 2025. Why is spatial reasoning hard for VLMs? an attention mechanism perspective on focus areas. In *International conference on machine learning (ICML)*.

Ronghao Dang, Yuqian Yuan, Yunxuan Mao, Kehan Li, Jiangpin Liu, Zhikai Wang, Xin Li, Fan Wang, and Deli Zhao. 2025. Rynnec: Bringing mllms into embodied world. *Preprint*, arXiv:2508.14160.

Danny Driess, Jost Tobias Springenberg, Brian Ichter, Lili Yu, Adrian Li-Bell, Karl Pertsch, Allen Z. Ren, Homer Walke, Quan Vuong, Lucy Xiaoyang Shi, and Sergey Levine. 2025. Knowledge insulating vision-language-action models: Train fast, run fast, generalize better. *Preprint*, arXiv:2505.23705.

Zhen Fang, Zhuoyang Liu, Jiaming Liu, Hao Chen, Yu Zeng, Shiting Huang, Zehui Chen, Lin Chen, Shanghang Zhang, and Feng Zhao. 2025. Dualvla: Building a generalizable embodied agent via partial decoupling of reasoning and action. *Preprint*, arXiv:2511.22134.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. 2025. MME: A comprehensive evaluation benchmark for multimodal large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Asher J. Hancock, Xindi Wu, Lihan Zha, Olga Russakovsky, and Anirudha Majumdar. 2025. Actions as language: Fine-tuning vlms into vlas without catastrophic forgetting. *Preprint*, arXiv:2509.22195.

Xiaoshuai Hao, Lei Zhou, Zhijian Huang, Zhiwen Hou, Yingbo Tang, Lingfeng Zhang, Guang Li, Zheng Lu, Shuhuai Ren, Xianhui Meng, Yuchen Zhang, Jing Wu, Jinghui Lu, Chenxu Dang, Jiayi Guan, Jianhua Wu, Zhiyi Hou, Hanbing Li, Shumeng Xia, and 25 others. 2025. Mimo-embodied: X-embodied foundation model technical report. *Preprint*, arXiv:2511.16518.

Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. 2025. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. *Preprint*, arXiv:2507.16815.

Physical Intelligence, 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, and 17 others. 2025. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *Preprint*, arXiv:2504.16054.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. 2024. Openvla: An open-source vision-language-action model. In *Annual Conference on Robot Learning (CoRL)*.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. *Preprint*, arXiv:1905.00414.

Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. 2023a. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. 2025. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. *Preprint*, arXiv:2510.12276.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. 2024a. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *Preprint*, arXiv:2411.19650.

Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, Hanbo Zhang, and Huaping Liu. 2024b. Towards generalist robot policies: What matters in building vision-language-action models. *Preprint*, arXiv:2412.14058.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. 2024c. Evaluating real-world robot manipulation policies in simulation. In *Annual Conference on Robot Learning (CoRL)*.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023b. Evaluating object hallucination in large vision-language models. *Preprint*, arXiv:2305.10355.

Weixin Liang, LILI YU, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen tau Yih, Luke Zettlemoyer, and Xi Victoria Lin. 2025. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *Transactions on Machine Learning Research*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for llm compression and acceleration. In *MLSys*.

Xiaopeng Lin, Shijie Lian, Bin Yu, Ruoqi Yang, Changti Wu, Yuzhuo Miao, Yurun Jin, Yukun Shi, Cong Huang, Bojun Cheng, and Kai Chen. 2025. Physbrain: Human egocentric data as a bridge from vision language models to physical intelligence. *Preprint*, arXiv:2512.16793.

Yixin Lin, Austin S. Wang, Giovanni Sutanto, Akshara Rai, and Franziska Meier. 2021. Polymetis. `https://facebookresearch.github.io/fairo/polymetis/`.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. 2023a. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023b. Visual instruction tuning. In *Advances in neural information processing systems (NeurIPS)*.

David Lopez-Paz and Marc' Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *Preprint*, arXiv:1711.05101.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*.

Pietro Mazzaglia, Cansu Sancaktar, Markus Peschl, and Daniel Dijkman. 2025. Hybrid training for vision-language-action models. *Preprint*, arXiv:2510.00600.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. 2024. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems*.

NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, and 24 others. 2025. Gr00t n1: An open foundation model for generalist humanoid robots. *Preprint*, arXiv:2503.14734.

Abby ONeill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, and 1 others. 2024. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE.

William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4195–4205.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. 2025. Fast: Efficient action tokenization for vision-language-action models. *Preprint*, arXiv:2501.09747.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, and Xuelong Li. 2025. Spatialvla: Exploring spatial representations for visual-language-action model. *Preprint*, arXiv:2501.15830.

Yichao Shen, Fangyun Wei, Zhiying Du, Yaobo Liang, Yan Lu, Jiaolong Yang, Nanning Zheng, and Baining Guo. 2025. Videovla: Video generators can be generalizable robot manipulators. *Preprint*, arXiv:2512.06963.

starVLA. 2025. Starvla: A lego-like codebase for vision-language-action model developing. GitHub repository.

Huajie Tan, Yuheng Ji, Xiaoshuai Hao, Xiansheng Chen, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. 2025. Reason-rft: Reinforcement fine-tuning for visual reasoning of vision language models. *Preprint*, arXiv:2503.20752.

BAAI RoboBrain Team, Mingyu Cao, Huajie Tan, Yuheng Ji, Xiansheng Chen, Minglan Lin, Zhiyu Li, Zhou Cao, Pengwei Wang, Enshen Zhou, Yi Han, Yingbo Tang, Xiangqi Xu, Wei Guo, Yaoxu Lyu, Yijie Xu, Jiayu Shi, Mengfei Du, Cheng Chi, and 34 others. 2025a. Robobrain 2.0 technical report. *Preprint*, arXiv:2507.02029.

GEAR Team, Allison Azzolini, Johan Bjorck, Valts Blukis, Fernando Castañeda, Rahul Chand, and 1 others. 2025b. Gr00t n1.6: An improved open foundation model for generalist humanoid robots. `https://research.nvidia.com/labs/gear/gr00t-n1_6/`.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. 2024. Octo: An open-source generalist robot policy. *Preprint*, arXiv:2405.12213.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, Siteng Huang, Yifan Tang, Wenhui Wang, Ru Zhang, Jianyi Liu, and Donglin Wang. 2025. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model. *arXiv preprint arXiv:2509.09372*.

Yuxi Xiao, Longfei Li, Shen Yan, Xinhang Liu, Sida Peng, Yunchao Wei, Xiaowei Zhou, and Bingyi Kang. 2026. Spatialtree: How spatial abilities branch out in mllms. *Preprint*, arXiv:2512.20617.

Rui Yang, Ziyu Zhu, Yanwei Li, Jingjia Huang, Shen Yan, Siyuan Zhou, Zhe Liu, Xiangtai Li, Shuangye Li, Wenqian Wang, Yi Lin, and Hengshuang Zhao. 2025a. Visual spatial tuning. *Preprint*, arXiv:2511.05491.

Shusheng Yang, Jihan Yang, Pinzhi Huang, Ellis Brown, Zihao Yang, Yue Yu, Shengbang Tong, Zihan Zheng, Yifan Xu, Muhan Wang, Daohan Lu, Rob Fergus, Yann LeCun, Li Fei-Fei, and Saining Xie. 2025b. Cambrian-s: Towards spatial supersensing in video. *Preprint*, arXiv:2511.04670.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836. Curran Associates, Inc.

Jianke Zhang, Xiaoyu Chen, Qiuyue Wang, Mingsheng Li, Yanjiang Guo, Yucheng Hu, Jiajun Zhang, Shuai Bai, Junyang Lin, and Jianyu Chen. 2026a. Vlm4vla: Revisiting vision-language-models in vision-language-action models. *Preprint*, arXiv:2601.03309.

Jianke Zhang, Yuanfei Luo, Yucheng Hu, Xiaoyu Chen, Yanjiang Guo, Ziyang Liu, Hongbin Xu, Tian Lan, and Jianyu Chen. 2026b. Uam: A dual-stream perspective on forgetting in vla training. *Preprint*, arXiv:2605.15735.

Ziang Zhang, Zehan Wang, Guanghao Zhang, Weilong Dai, Yan Xia, Ziang Yan, Minjie Hong, and Zhou Zhao. 2025. Dsi-bench: A benchmark for dynamic spatial intelligence. *Preprint*, arXiv:2510.18873.

Jinliang Zheng, Jianxiong Li, Dongxiu Liu, Yinan Zheng, Zhihao Wang, Zhonghong Ou, Yu Liu, Jingjing Liu, Ya-Qin Zhang, and Xianyuan Zhan. 2025a. Universal actions for enhanced embodied foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22508–22519.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. 2025b. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *Preprint*, arXiv:2412.10345.

Enshen Zhou, Jingkun An, Cheng Chi, Yi Han, Shanyu Rong, Chi Zhang, Pengwei Wang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, and Shanghang Zhang. 2026. Roborefer: Towards spatial referring with reasoning in vision-language models for robotics. *Preprint*, arXiv:2506.04308.

Zhongyi Zhou, Yichen Zhu, Minjie Zhu, Junjie Wen, Ning Liu, Zhiyuan Xu, Weibin Meng, Ran Cheng, Yaxin Peng, Chaomin Shen, and Feifei Feng. 2025. Chatvla: Unified multimodal understanding and robot control with vision-language-action model. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Chenming Zhu, Tai Wang, Wenwei Zhang, Jiangmiao Pang, and Xihui Liu. 2025. Llava-3d: A simple yet effective pathway to empowering lmms with 3d-awareness. *Preprint*, arXiv:2409.18125.

A Real Robot Experiments Detail

This section describes the training implementation and evaluation setup for our real-robot experiments.

A.1 Real Robot Pre-training

Table 8 presents the OXE subsets used for our pre-training.

Table 8: **Selected OXE subsets for pre-training.** We curate 20 Open X-Embodiment subsets with end-effector position control.

Dataset Name	Robot	Episodes
Berkeley Bridge	WidowX	25,460
RT-1 Robot Action	Google Robot	79,499
DROID	Franka	92,233
BC-Z	Google Robot	39,350
Stanford HYDRA	Franka	550
Austin VIOLA	Franka	135
Berkeley Autolab UR5	UR5	896
NYU Franka Play	Franka	456
Stanford Kuka Multimodal	Kuka iiwa	3,000
Freiburg Franka Play	Franka	3,242
USC Jaco Play	Jaco 2	976
NYU VINN	Hello Stretch	435
Austin BUDS	Franka	50
UCSD Kitchen	xArm	150
CMU Franka Pick-Insert Data	Franka	520
Austin Mutex	Franka	1,500
Berkeley Fanuc Manipulation	Fanuc Mate	415
CMU Play Fusion	Franka	576
CMU Stretch	Hello Stretch	135
DobbE	Hello Stretch	5,208
Total		**254,786**

A.2 Data Collection via Teleoperation

We use Polymetis (Lin et al., 2021) for teleoperation and collect human expert data to support imitation learning for VLA models. For real-world experiments, we utilize a Franka Research 3 robot arm equipped with two Intel RealSense D435 cameras. We collected 300 expert human demonstrations, following the instruction format: *"pick up the {color} block and put it into the gray box"*. The dataset is balanced, containing 100 trajectories for each of the three colors: red, green, and blue. A schematic illustration of the demonstration samples is presented in Figure 7.

A.3 Fine Tuning on Human Expert Demonstration Samples

We perform fine-tuning on the checkpoints pre-trained on the OXE subsets, using the human expert demonstration data we collected. The training is conducted on 8 $\times$ H100 GPUs with a per-device batch size of 16 for 80K steps.

Figure 7: Diagram of human expert data acquired through teleoperation

A.4 Real Robot Evaluation

Our evaluation is divided into four settings: in-domain, task generalization, instruction generalization, and pick-all.

- **In-Domain**. The evaluation tasks use exactly the same objects and task instructions as those in the training set, with the positions of the blocks altered randomly.
- **Task Generalization**. The visual attributes of the target object are altered from those seen during training. Specifically, the model is tested with the instruction: *"pick up the yellow block and put it into the gray box"*, where yellow is a color not present in the training set.
- **Instruction Generalization**. To systematically evaluate language-driven generalization, we employ Gemini 3 Pro to generate 50 semantically equivalent but lexically diverse paraphrases of the original training instruction (*"pick up the red block and put it into the gray box"*). The paraphrases vary along multiple linguistic dimensions, including vocabulary choice (e.g., *"place"* vs. *"move"* vs. *"transfer"*), object description (e.g., *"red block"* vs. *"crimson cube"* vs. *"scarlet object"*), and sentence structure (e.g., active vs. passive voice, prepositional variations). Each paraphrased instruction is executed 2 times on the physical robot, yielding 100 independent trials in total. The average success rate across all 100 trials is reported. This protocol directly probes whether the model can leverage its preserved semantic understanding to generalize beyond the fixed linguistic form seen during training, without any additional fine-tuning.
- **Pick-All**. The task instruction requires the robotic arm to place all blocks into the box: *"pick up all blocks and put them into the gray box"*, with four blocks (red, blue, green, and yellow) present in the workspace. The model must comprehend the task semantics and achieve long-horizon generalization by composing atomic actions learned from the training dataset.

B Implementation Details

We instantiate the VLM backbones by initializing the weights from Qwen2.5-VL-3B-Instruct and Qwen3-VL-4B-Instruct. The model is fine-tuned for 40K steps on a cluster of 8 GPUs (batch size 16 per device). We employ the AdamW (Loshchilov and Hutter, 2019) optimizer initialized with a learning rate of 1e-5 and a cosine annealing schedule. System-level optimizations include DeepSpeed ZeRO-2 (Loshchilov and Hutter, 2019), gradient clipping at a norm of 1.0, and no gradient accumulation. Our training pipeline is built upon the starVLA (starVLA, 2025) framework.

For fine-tuning, we set the default training duration to 40K steps. This choice aligns with standard practices in related work and ensures that the training can be completed within one day on NVIDIA H100 GPUs.

Our model architecture incorporates a state encoder designed to process proprioceptive information, such as joint angles and gripper states. However, since mainstream simulation benchmarks (e.g., SimplerEnv) primarily evaluate vision-only policies and do not typically involve state observations in their standard protocols, we omitted the state encoder during all simulation experiments to ensure fair comparison. Conversely, in real-robot settings where precise proprioception is crucial for control, the state encoder was active and utilized throughout both the pre-training and fine-tuning phases.

Figure 8: Examples from the Bridge V2 dataset in the training set.

C Layer-wise Representation Analysis via CKA

C.1 Background: Centered Kernel Alignment (CKA)

Centered Kernel Alignment (CKA) (Kornblith et al., 2019) is a metric for measuring the similarity between representations learned by neural networks. Given two sets of representations $\mathbf{X} \in \mathbb{R}^{N \times D_1}$ and $\mathbf{Y} \in \mathbb{R}^{N \times D_2}$ (where N is the number of samples and D is the feature dimension), CKA is defined as:

$$\mathrm{CKA}(\mathbf{X}, \mathbf{Y}) = \frac{\mathrm{HSIC}(\mathbf{K}, \mathbf{L})}{\sqrt{\mathrm{HSIC}(\mathbf{K}, \mathbf{K}) \cdot \mathrm{HSIC}(\mathbf{L}, \mathbf{L})}}, \tag{9}$$

where $\mathbf{K}$ and $\mathbf{L}$ are the Gram matrices computed from $\mathbf{X}$ and $\mathbf{Y}$ respectively, and HSIC (Hilbert-Schmidt Independence Criterion) measures the statistical dependence between two distributions. The CKA value ranges from 0 to 1:

- **CKA = 1**: The two representations are identical up to a linear transformation.
- **CKA = 0**: The two representations are completely uncorrelated.

C.2 Experimental Setup

Model and Checkpoints. We use Qwen2.5-VL-3B-Instruct as the VLM backbone. The checkpoints are obtained by training a standard VLA model following the GR00T architecture (NVIDIA et al., 2025) under the standard VLA imitation learning paradigm on the OXE dataset, with optimizer and data-processing settings matched to the main experiments in this paper. For this diagnostic run, we save checkpoints at intervals of 1,000 steps up to 50,000 steps and select 8 representative checkpoints for CKA analysis: step 0 (pre-trained), 1k, 2k, 5k, 10k, 20k, 30k, and 50k.

Dataset. We construct an evaluation set of 500 image-instruction pairs sampled from diverse vision-language tasks (including VQA, image captioning, and visual reasoning) that are distinct from the robotic training data. This ensures we measure the drift of general semantic representations rather than task-specific adaptations.

Feature Extraction. For each checkpoint, we extract hidden states from all transformer layers (24 layers for Qwen2.5-VL-3B) given the same input batch. We use the output of each layer before the residual connection as the layer representation. The features are mean-pooled across the sequence dimension to obtain a single vector per layer per sample.

CKA Computation. We compute the linear CKA between each layer of the fine-tuned checkpoint and the corresponding layer of the original pre-trained model (step 0). For each layer l, we calculate:

$$\mathrm{CKA}_l = \mathrm{CKA}\left(\mathbf{H}_l^{(t)}, \mathbf{H}_l^{(0)}\right), \tag{10}$$

where $\mathbf{H}_l^{(t)}$ denotes the hidden states from layer l at training step t, and $\mathbf{H}_l^{(0)}$ denotes the original pre-trained model.

C.3 Results and Analysis

Global Trend. As shown in Figure 3, CKA values exhibit a consistent downward trend across all layers as training progresses. The average CKA across all layers drops from 1.0 (step 0) to approximately 0.25 (step 50k), indicating a *fundamental and irreversible* transformation of the internal representation space.

Layer-wise Patterns. We observe distinct patterns across different layers:

1. **Early layers (1-6):** CKA decreases moderately, maintaining values around 0.4-0.5 by the end of training. These layers primarily encode low-level visual features, which are partially preserved as they serve as shared visual encoders for both semantic understanding and robotic control.

2. **Middle layers (7-18):** CKA drops sharply, reaching values below 0.3. These layers are responsible for mid-level feature abstraction and appear to be most affected by the adaptation to robotic action prediction.

3. **Deep layers (19-24):** CKA decreases most dramatically, with final values often below 0.2. These high-level semantic representations, crucial for visual reasoning and language understanding, are severely disrupted during VLA fine-tuning.

Temporal Dynamics. The representational shift occurs rapidly in the early stages of training:

- Within the first 1,000 steps, average CKA drops to 0.7.
- By step 5,000, CKA falls below 0.5 for most layers.
- After step 20,000, the decrease slows but continues to drift downward.

Implications. The severe representational drift observed in deep layers (CKA < 0.2) explains why standard VLA training leads to catastrophic forgetting of semantic capabilities. The model is not merely adjusting its output layer; it is fundamentally rewiring its internal semantic representations to prioritize proprioceptive control over visual understanding. This validates our design choice of **structural decoupling**: by freezing the Left Brain, we ensure that these critical semantic representations remain intact (CKA $\equiv 1.0$), while the Right Brain is free to develop task-specific representations for robotic control.

C.4 Comparison with Co-Training

We also conduct CKA analysis for the Co-Training paradigm (mixing VQA data with robotic data at 1:1 ratio). Interestingly, while Co-Training partially preserves downstream VQA performance (as shown in Figure 2b), the CKA values still exhibit significant drift (final average CKA ≈ 0.35). This suggests that Co-Training mitigates the *symptoms* of forgetting (output performance) but does not prevent the *root cause* (internal representational drift). The semantic features are still being altered, just at a slower pace or with partial recovery from the auxiliary VQA loss.

C.5 Computational Cost

All experiments are conducted on a single NVIDIA A100 GPU. Computing CKA for 500 samples across 24 layers and 8 checkpoints takes approximately 15 minutes.

D QwenGR00T, QwenPI and QwenOFT

For certain simulation experiments, we adopted the built-in baselines provided by StarVLA, re-implementing, training, and evaluating them under our uniform experimental setup. The architectures of these baselines are detailed in this section. As illustrated in Figure 9, these models differ primarily in their action modeling and VLM-policy interaction mechanisms:

- **QwenGR00T (Figure 9a):** Incorporating a Flow Matching-based Action Expert, this model follows a *late-fusion* strategy (NVIDIA et al., 2025). The VLM functions as a high-level semantic encoder, passing only its last hidden states to the Action Expert. This design treats the VLM representation as a fixed holistic condition for the flow generation process.

- **QwenPI (Figure 9b):** Also leveraging Flow Matching for continuous control, QwenPI adopts a *deep-fusion* architecture (Black et al., 2024). Unlike GR00T, it establishes dense connections between the VLM and the Action Expert, allowing for layer-wise attention interactions. This enables the policy to

access and leverage multi-scale semantic features from the VLM's intermediate layers during action generation.

- **QwenOFT (Figure 9c):** Representing the discrete control paradigm (Kim et al., 2024), this model discretizes the action space into token bins and predicts actions autoregressively, treating manipulation as a standard next-token prediction task.

The implementations of the aforementioned architectures are all adopted from the built-in modules of the starVLA (starVLA, 2025) framework and are **not** claimed as our original contributions.

E RoboCasa Tabletop Benchmark

The RoboCasa Tabletop benchmark comprises 24 distinct sub-tasks. We present the detailed success rates for all these tasks in Table 9.

F VLM Prompt Template

This section presents the prompt templates used for VLM input during the VLA training process.

System Prompt – Left Brain

```
You are a helpful robot brain that can
understand images and texts.
```

System Prompt – Right Brain

```
You are a helpful robot brain that can
understand images, texts, and robot states.
You will be provided with an observation, an
instruction, and the robot state.
Take action to execute the instruction.
```

User Prompt – Left Brain

```
<image>
Instruction: {instruction}
```

User Prompt – Right Brain

```
<image>
Instruction: {instruction}
Robot Type: {robot_type}
Predict the next action for the robot based
on the observation, instruction, and robot
type.
```

G Implementation Details on the LIBERO Benchmark

Training Dataset To evaluate the model's multi-task generalization capabilities, we adopted a unified training strategy. We aggregated the training datasets from all four LIBERO task suites (LIBERO-Spatial, -Object, -Goal, and -Long) into a single combined dataset. Instead of training separate experts for each suite, we trained a single generalist model on this unified data.

Training Configuration. We utilized Qwen3-VL-4B as the visual-language backbone. The model was trained on a cluster of 8 NVIDIA H100 GPUs. We set the per-device batch size to 16, resulting in a global batch size of 128. The training process was conducted for a total of 100,000 steps.

Evaluation Protocol. For evaluation, we strictly followed the official LIBERO evaluation protocol. We utilized the standard evaluation scripts provided by the benchmark. Each suite is evaluated with 500 trials, consisting of 10 distinct tasks with 50 evaluation episodes per task. The reported average is computed across the four suite-level success rates.

H "Twin-to-One" via Knowledge Distillation

H.1 Motivation and Methodology

While TwinBrainVLA achieves superior performance by leveraging a dual-stream architecture to utilize general capabilities, the inclusion of a second VLM inevitably increases computational overhead during deployment. To bridge the gap between high-level capability utilization and deployment efficiency, we propose a **"Twin-to-One"** knowledge distillation framework.

Our core hypothesis is that the "Right Brain" of TwinBrainVLA, having dynamically fused semantic knowledge from the frozen "Left Brain" via AsyMoT, possesses a superior feature space compared to a standard single-stream VLA. Therefore, we treat the fully trained TwinBrainVLA as the **Teacher** and a standard Vanilla VLA as the **Student**. The goal is to force the Student to internalize the Teacher's fused general capabilities by mimicking its internal representations.

Distillation Objective. We utilize the last hidden states of the Teacher's Right Brain as the supervision signal. Formally, let $H_{teacher} \in \mathbb{R}^{L \times D}$ denote the final hidden states of the Right Brain in TwinBrainVLA, and $H_{student} \in \mathbb{R}^{L \times D}$ denote the final hidden states of the Student Vanilla VLA. In addition to the standard action prediction loss $\mathcal{L}_{action}$, we introduce a feature alignment loss $\mathcal{L}_{feat}$:

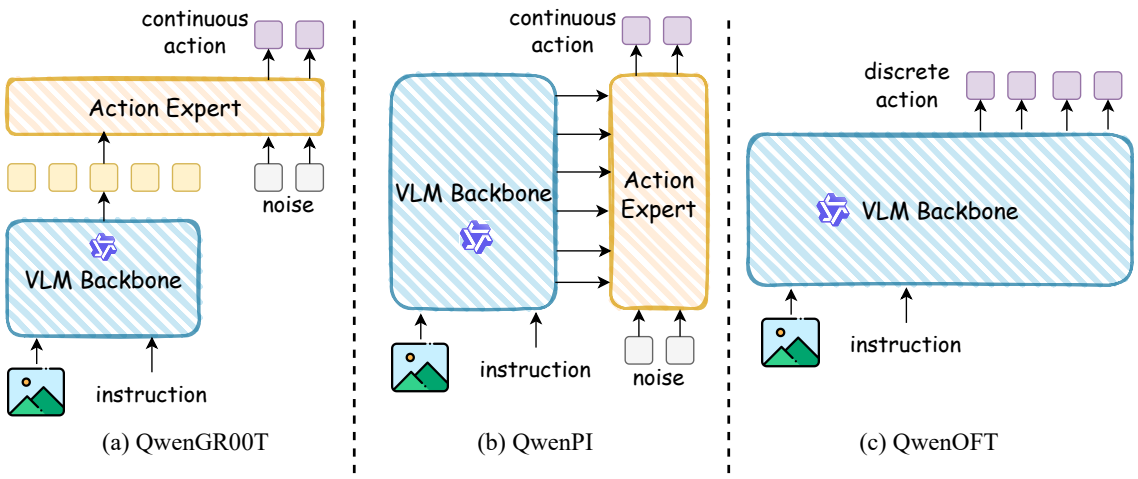


Figure 9: **Architectural overview of VLA baselines with distinct fusion strategies.** (a) **QwenGR00T** adopts a *late-fusion* design where a flow-matching Action Expert is conditioned solely on the last hidden states of the VLM. (b) **QwenPI** employs a *deep-fusion* strategy, enabling the flow-matching policy to interact with the VLM via layer-wise attention mechanisms. (c) **QwenOFT** follows the discrete token paradigm, predicting action bins autoregressively.

$$\mathcal{L}_{feat} = \frac{1}{L}\sum_{i=1}^{L} \|H^{(i)}_{student} - \mathrm{sg}(H^{(i)}_{teacher})\|_2^2 \quad (11)$$

where $\mathrm{sg}(\cdot)$ indicates the stop-gradient operation, ensuring the Teacher remains fixed. The total training objective for the Student is:

$$\mathcal{L}_{total} = \mathcal{L}_{action} + \lambda \mathcal{L}_{feat} \quad (12)$$

where λ is a balancing coefficient (set to 1.0 in our experiments).

H.2 Experiment Setup

We conducted the distillation experiment on the SimplerEnv benchmark. The training data (Bridge-V2 and Fractal subsets) and hyperparameters (e.g., learning rate, batch size, optimizer) are kept strictly identical to the main experiments described in Appendix B. The Student model architecture is identical to the Right Brain in TwinBrainVLA.

H.3 Results and Analysis

The quantitative results are presented in Table 10.

Performance Gains. As shown in Table 10, the Distilled Student achieves an average success rate of 58.3%, outperforming the standard Vanilla VLA (55.2%) by 3.1 points.

Conclusion. These results substantiate that the performance gains of TwinBrainVLA stem from the high-quality, semantics-rich features fused from the Left Brain. Furthermore, it demonstrates that TwinBrainVLA can serve as an effective supervisor, enabling single-stream models to internalize general capabilities and break the ceiling of standard fine-tuning paradigms.

I Inference Efficiency Optimizations

Although TwinBrainVLA employs two VLM streams, its asymmetric design admits a suite of complementary engineering optimizations that together bring the real-world inference overhead close to that of a single-stream Vanilla VLA. All latency measurements below are conducted on a single NVIDIA H100 GPU (single-GPU setting) or two NVIDIA H100 GPUs (multi-GPU pipeline setting), using the Qwen3-VL-4B-Instruct backbone with a batch size of 1 to reflect realistic robot deployment conditions.

(1) Shared Vision Encoder. Both the Left Brain and the Right Brain are initialized from the same pre-trained checkpoint and receive identical visual inputs at every timestep. The Vision Encoder (ViT) computation is therefore fully redundant in a naive dual-stream implementation. By sharing a single ViT forward pass and routing the resulting visual tokens to both streams, we eliminate half of the visual encoding cost with strictly zero impact on model outputs or accuracy.

(2) Left Brain KV Cache Pre-computation. The Left Brain is fully frozen and operates solely on

Figure 10: Schematic of RoboCasa Tabletop Evaluation.

the visual observation I and the task instruction T. In a typical robot manipulation episode, neither the task instruction nor the scene layout changes between control steps. Consequently, the Left Brain's Key-Value pairs $\{K_L^l, V_L^l\}_{l=1}^{L}$ can be computed once at the beginning of each episode and stored in GPU memory. All subsequent control steps reuse the cached KV pairs directly, reducing the Left Brain's amortized per-step cost to a single memory-read operation rather than a full transformer forward pass.

(3) Left Brain INT4 Quantization. As a frozen module that never participates in gradient computation, the Left Brain is an ideal candidate for post-training quantization. We apply weight-only INT4 quantization (using the AWQ algorithm (Lin et al., 2024)) to the Left Brain's linear layers. This reduces its parameter storage by $4\times$ and inference latency by approximately $2\times$ compared to BF16, with negligible degradation in the quality of the semantic Key-Value representations passed to the Right Brain. The Right Brain, which is responsible for precise continuous action prediction, is kept in BF16 to preserve numerical fidelity.

(4) Multi-GPU Pipeline Parallelism. When two or more GPUs are available a condition satisfied by the vast majority of real-world robot deployments the Left Brain and Right Brain can be placed on separate devices and executed asynchronously. Specifically, at timestep t, the Right Brain on GPU 1 generates the action a_t conditioned on the cached KV_L^t, while simultaneously the Left Brain on GPU 0 prefetches and caches KV_L^{t+1} for the next observation. This two-stage pipeline effectively hides the Left Brain's latency entirely behind the Right Brain's computation, as illustrated below:

Timestep	GPU 0 (Left Brain)	GPU 1 (Right Brain)
t	compute KV_L^{t+1}	generate a_t using KV_L^t
$t+1$	compute KV_L^{t+2}	generate a_{t+1} using KV_L^{t+1}

Table 9: **Results of evaluating the VLA models with the GR1 robot in the RoboCasa Tabletop simulation environment**. We highlight the best results in **bold** and the second-best results with underline.

Task	Isaac-GR00T N1.6	QwenGR00T + Qwen3VL	QwenPI + Qwen3VL	TwinBrainVLA + Qwen2.5VL	TwinBrainVLA + Qwen3VL
PnP Bottle To Cabinet Close	51.5	46.0	26.0	62.0	74.0
PnP Can To Drawer Close	13.0	80.0	62.0	66.0	72.0
PnP Cup To Drawer Close	8.5	54.0	42.0	46.0	52.0
PnP Milk To Microwave Close	14.0	48.0	50.0	52.0	60.0
PnP Potato To Microwave Close	41.5	28.0	42.0	56.0	36.0
PnP Wine To Cabinet Close	16.5	46.0	32.0	58.0	46.0
PnP Novel From Cuttingboard To Basket	58.0	48.0	40.0	50.0	62.0
PnP Novel From Cuttingboard To Cardboardbox	46.5	40.0	46.0	44.0	46.0
PnP Novel From Cuttingboard To Pan	68.5	68.0	60.0	64.0	70.0
PnP Novel From Cuttingboard To Pot	65.0	52.0	40.0	60.0	66.0
PnP Novel From Cuttingboard To Tieredbasket	46.5	56.0	44.0	52.0	52.0
PnP Novel From Placemat To Basket	58.5	42.0	44.0	56.0	30.0
PnP Novel From Placemat To Bowl	57.5	44.0	52.0	59.0	54.0
PnP Novel From Placemat To Plate	63.0	48.0	50.0	62.0	64.0
PnP Novel From Placemat To Tieredshelf	28.5	18.0	28.0	28.0	38.0
PnP Novel From Plate To Bowl	57.0	60.0	52.0	62.0	60.0
PnP Novel From Plate To Cardboardbox	43.5	50.0	40.0	52.0	58.0
PnP Novel From Plate To Pan	51.0	54.0	36.0	56.0	56.0
PnP Novel From Plate To Plate	78.7	70.0	48.0	68.0	66.0
PnP Novel From Tray To Cardboardbox	51.5	38.0	34.0	40.0	46.0
PnP Novel From Tray To Plate	71.0	56.0	64.0	56.0	72.0
PnP Novel From Tray To Pot	64.5	50.0	44.0	60.0	56.0
PnP Novel From Tray To Tieredbasket	57.0	36.0	50.0	42.0	46.0
PnP Novel From Tray To Tieredshelf	31.5	16.0	28.0	34.0	28.0
Average	47.6	47.8	43.9	53.5	**54.6**

Table 10: **"Twin-to-One" Distillation Results on SimplerEnv.** We compare the distilled single-stream model (Student) against the standard Vanilla VLA and the Teacher (TwinBrainVLA). All models use the Qwen3-VL-4B-Instruct backbone.

Method	Spoon on Towel	Carrot on Plate	Stack Block	Eggplant in Basket	Average
Baselines					
Vanilla VLA (Standard Training)	87.5	50.0	29.2	54.2	55.2
Teacher					
TwinBrainVLA (dual-stream)	87.5	58.3	45.8	93.8	**71.4**
"Twin-to-One"					
Distilled Student (single-stream)	83.3	54.2	29.2	66.7	58.3